Consensus

Optimal Control and Learning – Lecture 8

Kevin Wu

Oct. 10, 2025

From Isolated Agents to Cooperative Control

- Consider a simple autonomous vehicle model: $\dot{x}_i = v_i,\ \dot{v}_i = u_i$

$$\min_{u_i} \int_0^T \|x_i - x_i^{\text{ref}}\|^2 + \|u_i\|^2 \, dt$$

- But what about in a platoon or formation task?
  - Objective depends on *relative* positions

Cooperative Control

$$\min_{u_i} \int_0^T \|x_i - x_i^{\text{ref}}\|^2 + \|u_i\|^2 + \alpha \sum_{j \in \mathcal{N}_i} \|x_i - x_j\|^2 \, dt$$

- Each agent benefits from knowing neighbors' states
- How to reconcile shared quantities they depend on?

Consensus $x_i = x_j \quad \forall (i,j) \in \mathcal{E}$

- “To reach an agreement regarding a certain quantity of interest that depends on the state of all agents”

- **Consensus Algorithm / Protocol** := ensuring agreement on shared variables through local communication (neighbors on the network)

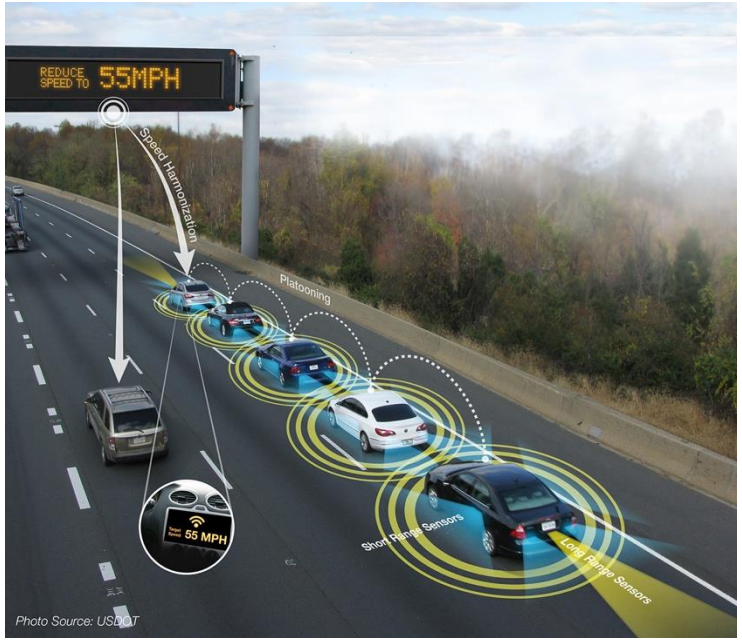
- **f-consensus problem** := a constrained consensus problem, i.e. state of all agents must become asymptotically equal, requires *cooperation*

Overview

- **Consensus: “distributed agreement”**
- ADMM: a general distributed optimization engine
- Consensus via ADMM
- Distributed MPC with ADMM consensus: a powerful tool for real control problems

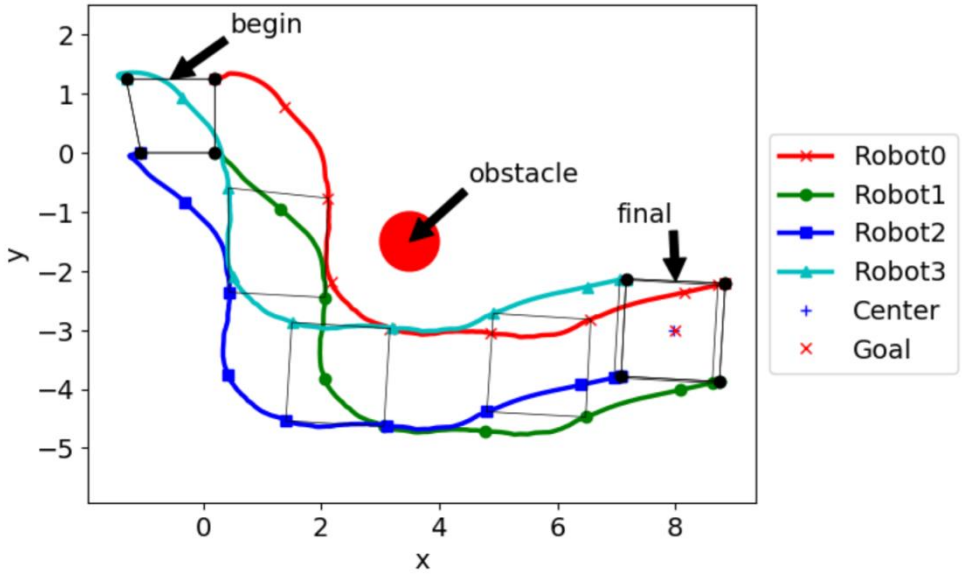
Olfati-Saber, Reza, J. Alex Fax, and Richard M. Murray. "Consensus and cooperation in networked multi-agent systems." *Proceedings of the IEEE* 95.1 (2007): 215-233.

Motivation



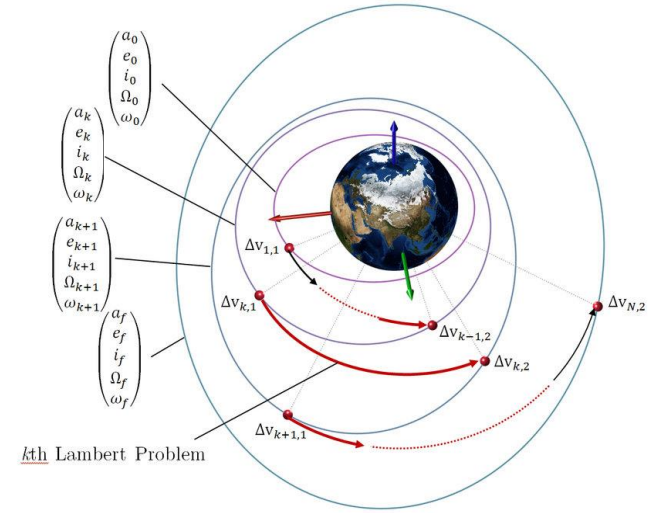
Autonomous vehicle platooning

Relative positions and velocities



Formation control

Relative positions and velocities

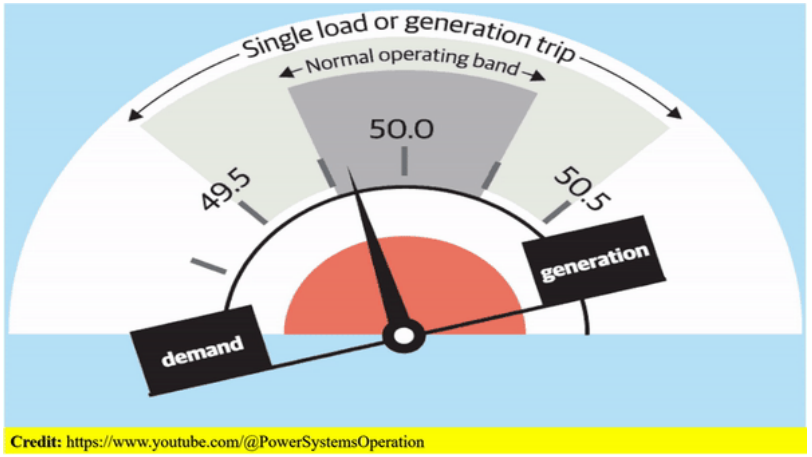


Space rendezvous

Position

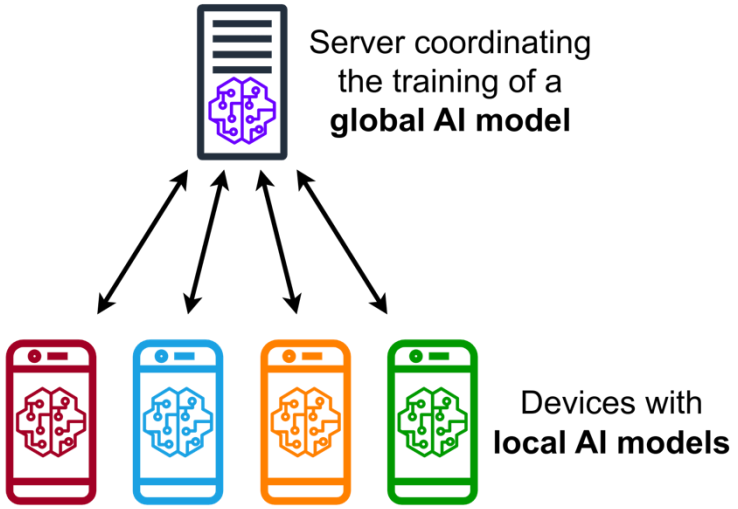
Flocking problem: achieving consensus in matching velocity

Motivation



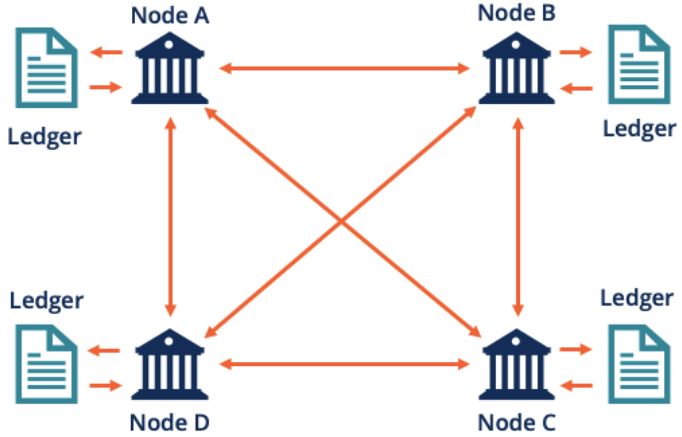
Power grid stability

Frequency/voltage



Federated learning

Common global model



Blockchain, distributed ledgers

Transaction history

Other applications: distributed sensor networks

Context sharing vs. Consensus

Mechanism	Enforces equality?	Primary purpose / example
Context sharing	No	Improving local predictions (e.g. Kalman filter)
Consensus	Yes	Enforcing a global consistent estimate
Distributed estimation	Mix of both	Combines fusion and agreement for multi-agent state estimation

Individual autonomous vehicles: shared information improves each others' local models

Platoons/formations: requires consensus constraints like velocity/inter-vehicle distance

Consensus for Control

Setting	Is Consensus Needed?	Why / Why not
Distributed estimation / filtering	YES, often	Share local measurements to estimate global state
Cooperative MPC (formation, coordination)	YES, sometimes	Agent costs depend on neighbors' states or shared variables
Independent MPC	No	Agent states/costs are separable
Centralized control	No	Global information already available

Why not centralized control for everything?

Consensus offers **scalability, privacy, robustness**

Consensus Notation 1

- Network of decision-making *agents*
  - With dynamics $\dot{x}_i = u_i$
  - Local communication with neighbors on graph $G = (V, E)$

- Achieve asymptotic convergence to an *agreement space,*

$$x_1 = x_2 = \ldots = x_n$$

- alternatively expressed as $x = \alpha \mathbf{1}$
- and $\alpha \in \mathbb{R}$ is referred to as the *collective decision*

Consensus Notation 2

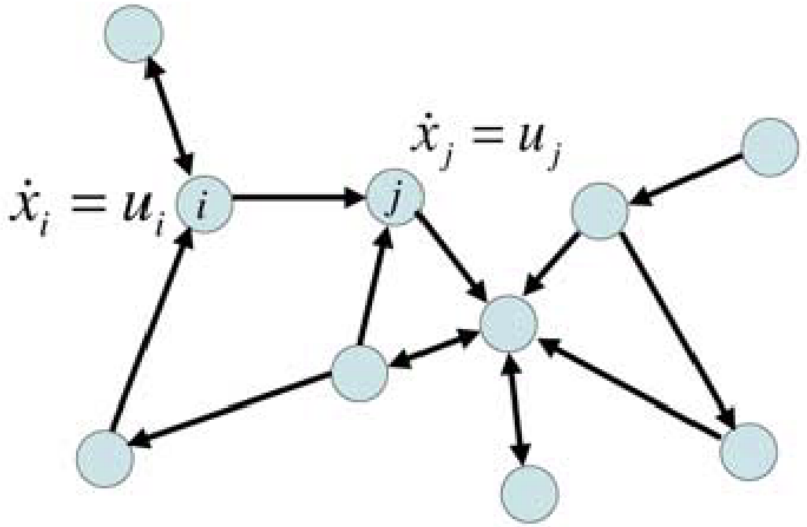
- Adjacency matrix: $A = [a_{ij}]$
- Set of neighbors of vertex *i*: $N_i = \{j \in V : a_{ij} \neq 0\}$
- Agent *i* communicates with agent *j* if $a_{ij} \neq 0$
- Furthermore, the graph can be made time-varying
  - Applications in mobile sensor networks, flocking

$G(t) = (V, E(t))$ $A(t)$ $N_i(t)$

Distributed Consensus Algorithm

$$\dot{x}_i(t) = \sum_{j \in N_i} \left(x_j(t) - x_i(t) \right) + b_i(t) \quad (1)$$

$$x_i(0) = z_i \in \mathbb{R}, b_i(t) = 0$$



Average-Consensus

$$\dot{x}_i(t) = \sum_{j \in N_i} a_{ij}\big(x_j(t) - x_i(t)\big) \quad (2)$$

- Suppose the graph is *connected* and *undirected:* $a_{ij} = a_{ji}$
- Then the special invariance property is satisfied: $\sum_i \dot{x}_i = 0$

- Final condition must satisfy: $\alpha = \frac{1}{n}\sum_i x_i(0)$

- Applications in sensor fusion in distributed sensor networks
- **Asymptotically solves an *average-consensus* problem for all initial states**

Graph Laplacian

$$\dot{x} = -Lx\,, \quad L = [l_{ij}]\,, \quad l_{ij} = \begin{cases} -1, & j \in N_i \\ |N_i|, & j = i. \end{cases}$$

- Since undirected, satisfies sum-of-squares (SOS)

$$x^T L x = \frac{1}{2} \sum_{(i,j)\in E} a_{ij}(x_j - x_i)^2$$

- Taking quadratic disagreement function: $\varphi(x) = \frac{1}{2} x^T L x$

- This is the **gradient-descent** algorithm: $\dot{x} = -\nabla \varphi(x)$

Algebraic Connectivity and Spectral Properties

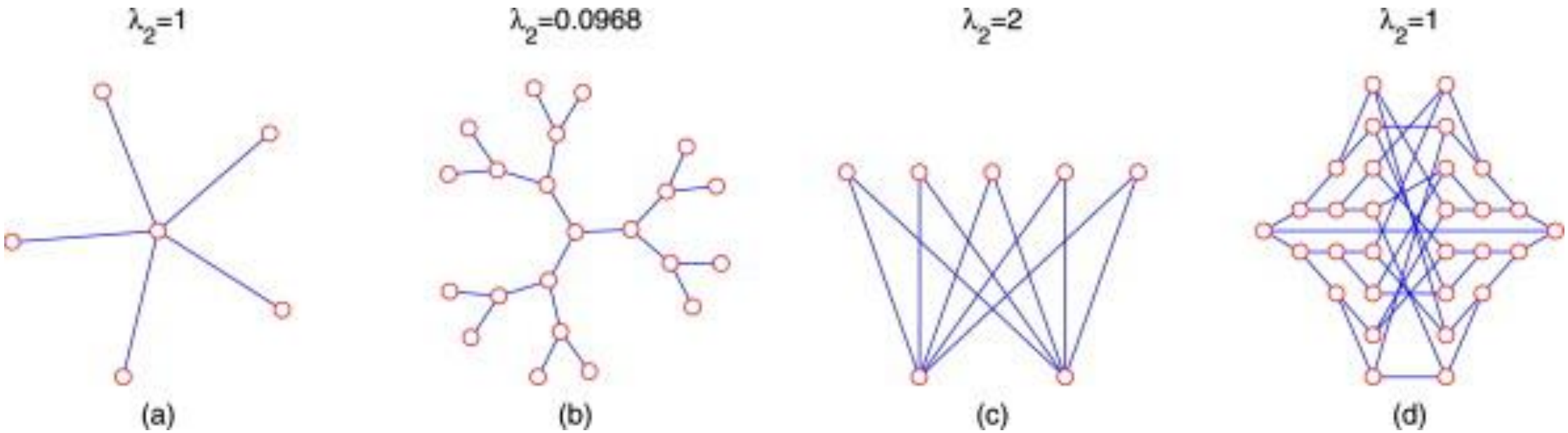
- Spectral properties of *L* affect the convergence of consensus algorithms
- For undirected graphs, *L* is symmetric and has real eigenvalues

$$0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n \leq 2\Delta$$

  - where Δ is the maximum degree of the graph

- λ_2 denotes the *algebraic connectivity*
  - Used as a measure of performance in consensus algorithms

Graph Connectivity and Consensus



- Algebraic connectivity quantifies how easily information flows through the network: larger λ_2 means faster convergence to consensus

Demo

Overview

- Consensus: “distributed agreement”
- **ADMM: a general distributed optimization engine**
- Consensus via ADMM
- Distributed MPC with ADMM consensus: a powerful tool for real control problems

Boyd, Stephen, et al. "Distributed optimization and statistical learning via the alternating direction method of multipliers." *Foundations and Trends® in Machine learning* 3.1 (2011): 1-122.

ADMM

- Alternating Direction Method of Multipliers
- Introduced in the mid-1970s

- Decomposition-coordination procedure, to parallelize small local subproblems in coordination to find a solution to a large global problem

ADMM Precursor: **Dual Ascent Method**

- Consider the convex optimization problem $\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & Ax = b \end{array}$
- Lagrangian: $L(x, y) = f(x) + y^T(Ax - b)$
- Dual function: $g(y) = \inf_x L(x, y) = -f^*(-A^T y) - b^T y$
- Dual problem: $\text{maximize} \quad g(y)$

- Iterative updates:

$$x^{k+1} := \operatorname*{argmin}_x L(x, y^k)$$

$$y^{k+1} := y^k + \alpha^k \underbrace{(Ax^{k+1} - b)}_{\text{residual}}$$

ADMM Precursor: **Dual Decomposition**

- Suppose now that the objective is separable: $f(x) = \sum_{i=1}^{N} f_i(x_i)$

$$L(x,y) = \sum_{i=1}^{N} L_i(x_i,y) = \sum_{i=1}^{N} \left(f_i(x_i) + y^T A_i x_i - (1/N) y^T b\right)$$

- The x-minimization step splits into *N* separate problems, solve in parallel:

$$x_i^{k+1} := \underset{x_i}{\operatorname{argmin}}\, L_i(x_i, y^k)$$
$$y^{k+1} := y^k + \alpha^k (A x^{k+1} - b)$$

ADMM Precursor: Augmented Lagrangians and Method of Multipliers

- Improve robustness of dual ascent, no longer needing assumptions of strict convexity or finiteness of *f*
- **Augmented Lagrangian**: $L_\rho(x, y) = f(x) + y^T(Ax - b) + (\rho/2)\|Ax - b\|_2^2$
- Associated with the problem:

$$\begin{array}{ll} \text{minimize} & f(x) + (\rho/2)\|Ax - b\|_2^2 \\ \text{subject to} & Ax = b. \end{array}$$

- **Method of multipliers:**

$$x^{k+1} := \operatorname*{argmin}_x L_\rho(x, y^k)$$

$$y^{k+1} := y^k + \rho(Ax^{k+1} - b)$$

- Downside: the penalty term is no longer separable

ADMM

- Blends dual ascent and the method of multipliers
- Given the problem: $\begin{array}{ll} \text{minimize} & f(x) + g(z) \\ \text{subject to} & Ax + Bz = c \end{array}$
- Form augmented Lagrangian:

$$L_\rho(x, z, y) = f(x) + g(z) + y^T(Ax + Bz - c) + (\rho/2)\|Ax + Bz - c\|_2^2$$

- Iterate:

$$\begin{aligned} x^{k+1} &:= \operatorname*{argmin}_x L_\rho(x, z^k, y^k) \\ z^{k+1} &:= \operatorname*{argmin}_z L_\rho(x^{k+1}, z, y^k) \\ y^{k+1} &:= y^k + \rho(Ax^{k+1} + Bz^{k+1} - c) \end{aligned}$$

Assumptions for Convergence

- (1) *f* must be closed, proper, and convex
  - Implies that the *x*-update is solvable

- (2) The un-augmented Lagrangian has a saddle point $(x^\star, z^\star, y^\star)$

$$L_0(x^\star, z^\star, y) \leq L_0(x^\star, z^\star, y^\star) \leq L_0(x, z, y^\star)$$

  - Implies that *y** is dual optimal and that strong duality holds

- Guarantees:
  - Residual convergence – iterates approach feasibility
  - Objective convergence – objective approaches optimal value
  - Dual variable convergence – iterates approach a dual optimal point

Optimality Conditions

- Primal Feasibility: $Ax^\star + Bz^\star - c = 0$
- Dual Feasibility:

$$0 \in \partial f(x^\star) + A^T y^\star$$

$$0 \in \partial g(z^\star) + B^T y^\star$$

- **Stopping Criterion:**
  - Computed as a bound on objective suboptimality of the current point
  - Chosen as feasibility tolerances of primal and dual residuals

Convergence in Practice

- Can be slow to converge to high accuracy
- However, converges to modest accuracy within few tens of iterations
  - For when modest accuracy is sufficient, e.g. large-scale problems
  - Statistical and ML problems (e.g. parameter estimation, diminishing returns in high accuracy)

Overview

- Consensus: “distributed agreement”
- ADMM: a general distributed optimization engine
- **Consensus via ADMM**
- Distributed MPC with ADMM consensus: a powerful tool for real control problems

Boyd, Stephen, et al. "Distributed optimization and statistical learning via the alternating direction method of multipliers." *Foundations and Trends® in Machine learning* 3.1 (2011): 1-122.

Consensus via ADMM

- Consider the *global consensus problem:* $\begin{array}{ll} \text{minimize} & \sum_{i=1}^{N} f_i(x_i) \\ \text{subject to} & x_i - z = 0, \quad i = 1, \ldots, N \end{array}$

- Augmented Lagrangian: $L_\rho(x_1, \ldots, x_N, z, y) = \sum_{i=1}^{N} \left(f_i(x_i) + y_i^T (x_i - z) + (\rho/2)\|x_i - z\|_2^2 \right)$
- ADMM algorithm:

$$x_i^{k+1} := \underset{x_i}{\operatorname{argmin}} \left(f_i(x_i) + y_i^{kT}(x_i - z^k) + (\rho/2)\|x_i - z^k\|_2^2 \right)$$

$$z^{k+1} := \frac{1}{N} \sum_{i=1}^{N} \left(x_i^{k+1} + (1/\rho) y_i^k \right)$$

$$y_i^{k+1} := y_i^k + \rho(x_i^{k+1} - z^{k+1}).$$

Global Consensus

- The algorithm can be simplified in notation:

$$z^{k+1} = \overline{x}^{k+1} + (1/\rho)\overline{y}^{k}$$

$$\overline{y}^{k+1} = \overline{y}^{k} + \rho(\overline{x}^{k+1} - z^{k+1})$$

- Hence, $\overline{y}^{k+1} = 0$, and we can write $z^k = \overline{x}^k$

$$x_i^{k+1} := \underset{x_i}{\operatorname{argmin}} \left(f_i(x_i) + y_i^{kT}(x_i - \overline{x}^k) + (\rho/2)\|x_i - \overline{x}^k\|_2^2 \right)$$

$$y_i^{k+1} := y_i^k + \rho(x_i^{k+1} - \overline{x}^{k+1}).$$

ADMM in Expansion Planning – Progressive Hedging

- First-stage investments *x* (e.g. generation/transmission/storage)
- Scenario-specific operational decisions z_s and costs f_s
- Non-anticipativity constraint

$$\min_{\{x_s, z_s\},\, x} \quad \sum_s p_s \left[c^\top x_s + f_s(z_s) \right]$$
$$\text{s.t.} \quad T x_s + W_s z_s = d_s,$$
$$x_s = x, \quad \forall s.$$

- Solve individual two-stage problems, relaxing anticipativity:

$$(x_s^{k+1}, z_s^{k+1}) = \arg\min_{x_s, z_s} \left[c^\top x_s + f_s(z_s) + y_s^{k\top}(x_s - x^k) + \tfrac{\rho}{2}\|x_s - x^k\|^2 \right]$$

- Update the consensus investment variable: $x^{k+1} = \sum_s p_s \left(x_s^{k+1} + \frac{1}{\rho} y_s^k \right)$
- Multiplier update: $y_s^{k+1} = y_s^k + \rho \left(x_s^{k+1} - x^{k+1} \right)$

General Form Consensus

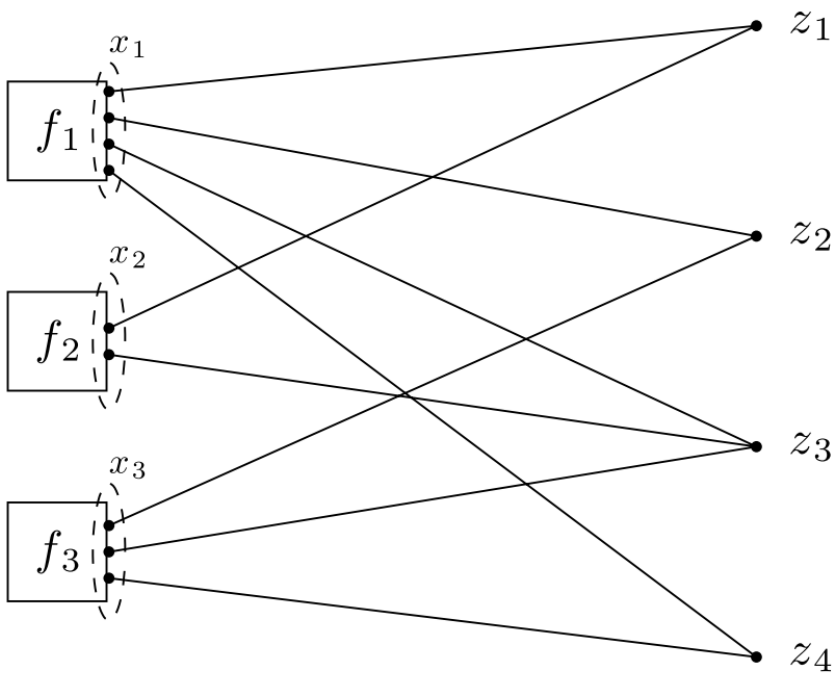
- Local variables *x* with separable objective $f_1(x_1) + \cdots + f_N(x_N)$
- Each local variable maps/corresponds to a global variable z_g with the mapping of $g = \mathcal{G}(i,j)$

$$(x_i)_j = z_{\mathcal{G}(i,j)}, \quad i = 1,\ldots,N, \quad j = 1,\ldots,n_i$$

- Application: model fitting
  - Global variable *z* is the full feature vector
  - Subsets of data distributed among *N* processors
  - x_i represents subvector of *z* that appears in block *i* of data
  - **datasets that are high-dimensional but sparse**

General Form Consensus

$$(\tilde{z}_i)_j = z_{\mathcal{G}(i,j)}$$



- Problem:

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^{N} f_i(x_i) \\ \text{subject to} & x_i - \tilde{z}_i = 0, \quad i = 1, \ldots, N \end{array}$$

- Iterate:

$$x_i^{k+1} := \underset{x_i}{\operatorname{argmin}} \left(f_i(x_i) + y_i^{kT} x_i + (\rho/2) \|x_i - \tilde{z}_i^k\|_2^2 \right)$$

$$z^{k+1} := \underset{z}{\operatorname{argmin}} \left(\sum_{i=1}^{m} \left(-y_i^{kT} \tilde{z}_i + (\rho/2) \|x_i^{k+1} - \tilde{z}_i\|_2^2 \right) \right)$$

$$y_i^{k+1} := y_i^k + \rho(x_i^{k+1} - \tilde{z}_i^{k+1}),$$

Overview

- Consensus: “distributed agreement”
- ADMM: a general distributed optimization engine
- Consensus via ADMM
- **Distributed MPC with ADMM consensus: a powerful tool for real control problems**

Summers, Tyler H., and John Lygeros. "Distributed model predictive consensus via the alternating direction method of multipliers." *2012 50th annual Allerton conference on communication, control, and computing (Allerton)*. IEEE, 2012.

Distributed Model Predictive Consensus via the Alternating Direction Method of Multipliers

- Consider the flocking problem in a network of double integrators
- Dynamics are discrete-time linear state:

$$x_i(t+1) = A_i x_i(t) + B_i u_i(t), \quad i = 1, ..., N$$

- Objective: infinite-horizon cost function

$$J = \sum_{t=0}^{\infty} \sum_{i=1}^{N} \ell_i(x_{\mathcal{N}_i}(t), u_{\mathcal{N}_i}(t))$$

where $x_{\mathcal{N}_i}(t)$, $u_{\mathcal{N}_i}(t)$ are the concatenations of states/inputs of neighbors of *i*

Distributed Model Predictive Consensus

- Goal: find a *distributed* optimal policy $\pi : \mathcal{X} \to \mathcal{U}$
- Each agent *i* should only depend on information from neighbors in *G*
- Finite-horizon:

$$\begin{aligned}
\text{minimize} \quad & J = \sum_{t=0}^{T-1} \sum_{i=1}^{N} \ell_i(x_{\mathcal{N}_i}(t), u_{\mathcal{N}_i}(t)) \\
& \qquad + \sum_{i=1}^{N} \ell_{if}(x_{\mathcal{N}_i}(T), u_{\mathcal{N}_i}(T)) \\
\text{subject to} \quad & x_i(t+1) = A_i x_i(t) + B_i u_i(t), \\
& x_{\mathcal{N}_i}(t) \in \mathcal{X}_i, \quad u_{\mathcal{N}_i}(t) \in \mathcal{U}_i, \\
& i = 1, ..., N, \quad t = 0, 1, ..., T-1 \\
& x_{\mathcal{N}_i}(T) \in \mathcal{X}_{if} \quad i = 1, ..., N,
\end{aligned}$$

Formulate into General Form Conensus

$$\begin{aligned} \underset{\mathbf{x}_i \in \mathbf{X}_i}{\text{minimize}} \quad & \sum_{i=1}^{N} f_i(\mathbf{x}_i) \\ \text{subject to} \quad & \mathbf{x}_i - \bar{\mathbf{E}}_i z = 0, \quad i = 1, ..., N. \end{aligned}$$

- Where $\boldsymbol{E_i}$ maps components of *x* that depend on shared variables

- Augmented Lagrangian:

$$\begin{aligned} L_\rho(\mathbf{x}, z, \lambda) = & \sum_{i=1}^{N} \Big[f_i(\mathbf{x}_i) \\ & + \lambda_i^T (\mathbf{x}_i - \bar{\mathbf{E}}_i z) + \frac{\rho}{2} \|\mathbf{x}_i - \bar{\mathbf{E}}_i z\|_2^2 \Big] \\ = & \sum_{i=1}^{N} L_{\rho i}(\mathbf{x}_i, z, \lambda) \end{aligned}$$

- ADMM:

$$\begin{aligned} \mathbf{x}_i^{k+1} &= \underset{\mathbf{x}_i \in \mathbf{X}_i}{\text{argmin}} \; L_{\rho i}(\mathbf{x}_i, z^k, \lambda^k) \\ z^{k+1} &= \underset{z}{\text{argmin}} \; L_\rho(\mathbf{x}^{k+1}, z, \lambda^k) \\ \lambda_i^{k+1} &= \lambda_i^k + \rho(\mathbf{x}_i^{k+1} - \bar{\mathbf{E}}_i z^{k+1}) \end{aligned}$$

Demo

Results

- *Kronecker product:* $P \otimes Q = [p_{ij}Q]$

- Euler discretization with step size T_s
- Agent mass m_i
- State space: position/velocity in 3D $\mathcal{X}_i = \mathbf{R}^6$
- Control: $\mathcal{U}_i = \{u \in \mathbf{R}^3 \mid \|u\|_\infty \leq u_{max}\}$
- Dynamics: $A_i = I_3 \otimes \begin{bmatrix} 1 & T_s \\ 0 & 1 \end{bmatrix}, \quad B_i = I_3 \otimes \begin{bmatrix} 0 \\ T_s/m_i \end{bmatrix}$

- Objective: all agents converge to common position and velocity **(flocking)**
- Noise: IID proves noise on acceleration with mean 0, variance 0.1
- Cost: penalize **neighbor disagreement** and **control effort**

$$\ell(x,u) = x^\top \big(I_3 \otimes (L \otimes I_2)\big)x \; + \; u^\top R u, \quad R = I_3 \otimes I_N$$

- Centralized solver (blue), ADMM (red)

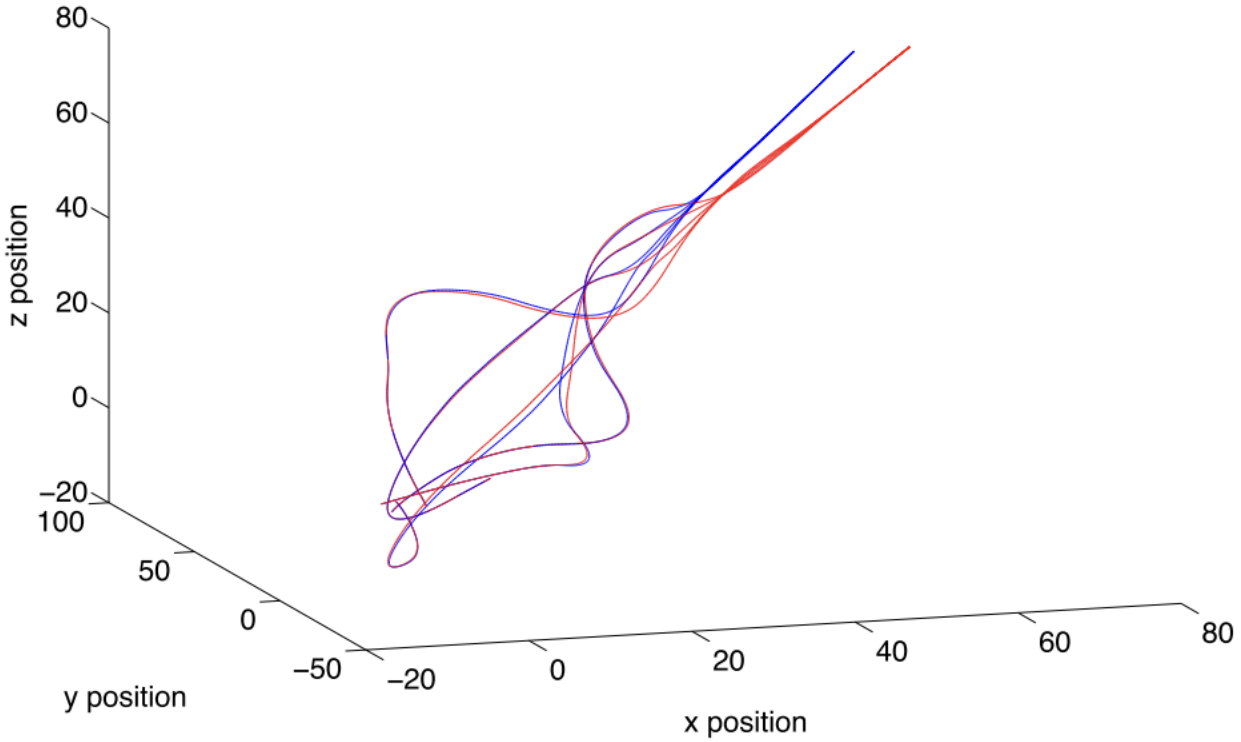
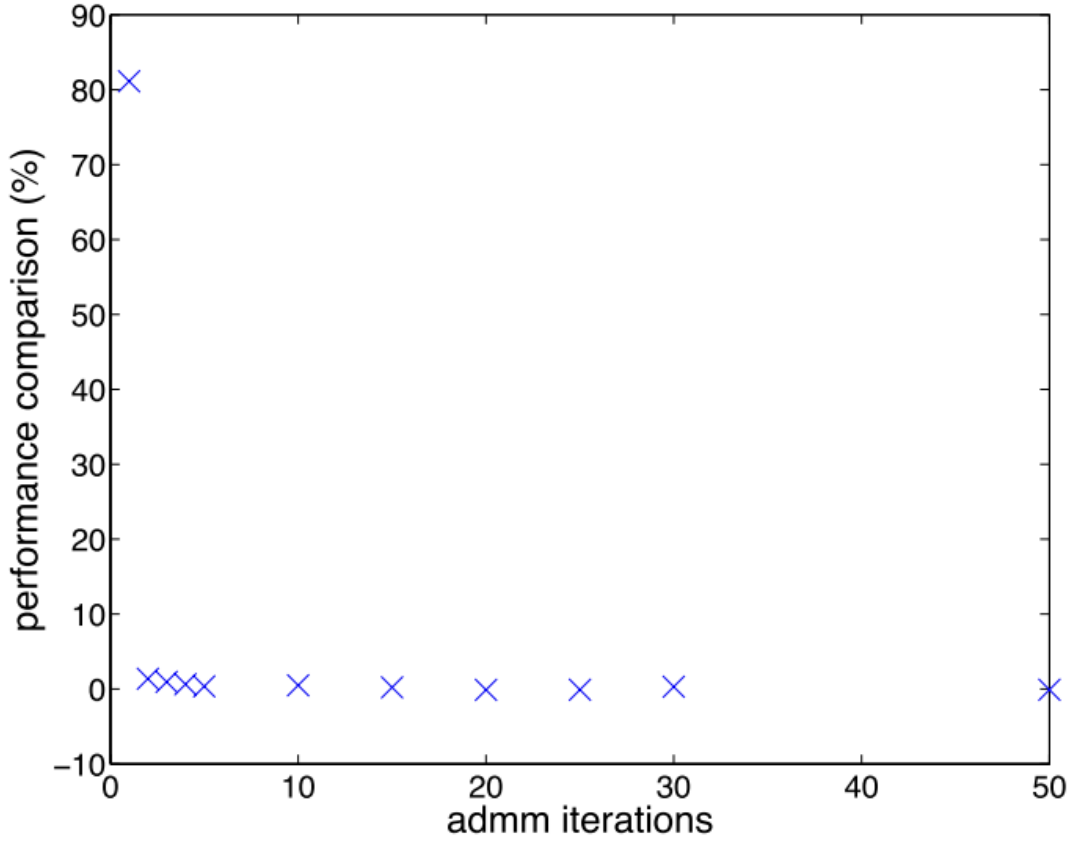


Fig. 4. Spatial plot of agent trajectories.

- Percent difference of ADMM cost with centralized solver



- N=5 agents, and T=10 time horizon, in 250-step closed loop simulation
- Local SPs solve in under 2ms (each ADMM iteration) -> **scalable!**

Conclusion

- **Consensus**
  - Mechanism for **agreement** among agents through local communication
  - Convergence guaranteed under **connectivity** and **convexity**
  - Enables coordination without centralized control (scalable, robust)
- **ADMM**
  - Augmented Lagrangian method that combines **local optimization** and **consensus enforcement**
  - Alternates between solving local subproblems and updating shared variables (dual updates)
  - Provides robust convergence and parallelizability for distributed optimization
- **Distributed MPC**
  - Enables agents to cooperatively solve MPC problems without centralized control, enabling scalability, privacy, and robustness