A Sheaf Theoretic Framework for Multi-Agent Model Predictive Control

Sam Cohen and Trevor Gross

Background

MPC

- MPC is a broadly useful control method

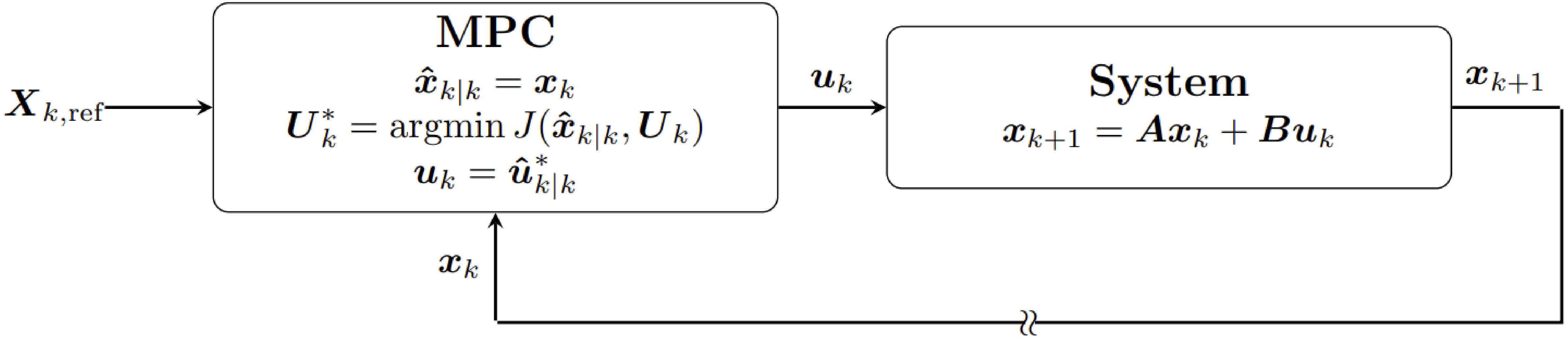
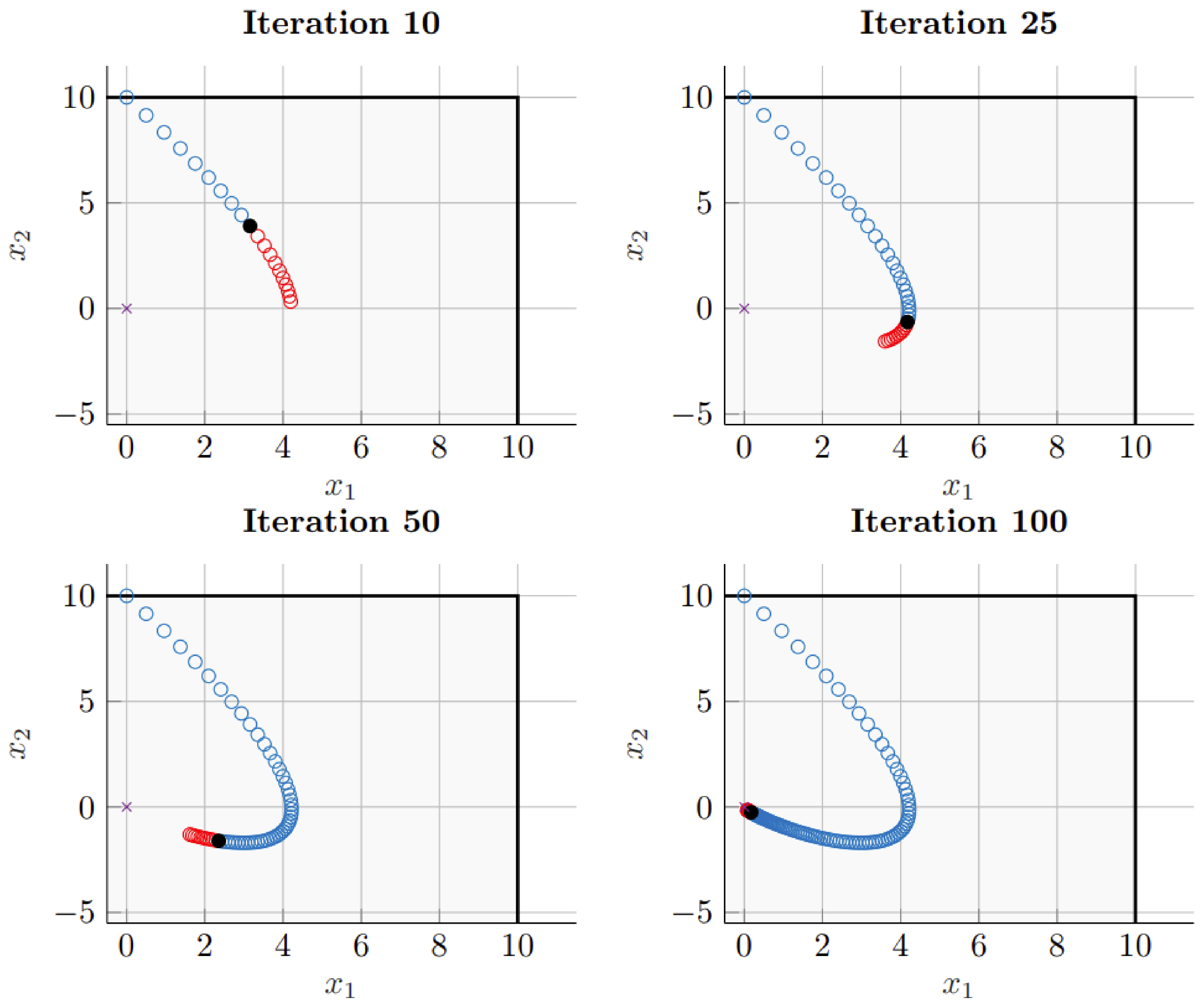


Figure 1: MPC scheme

MPC



Multi-agent MPC

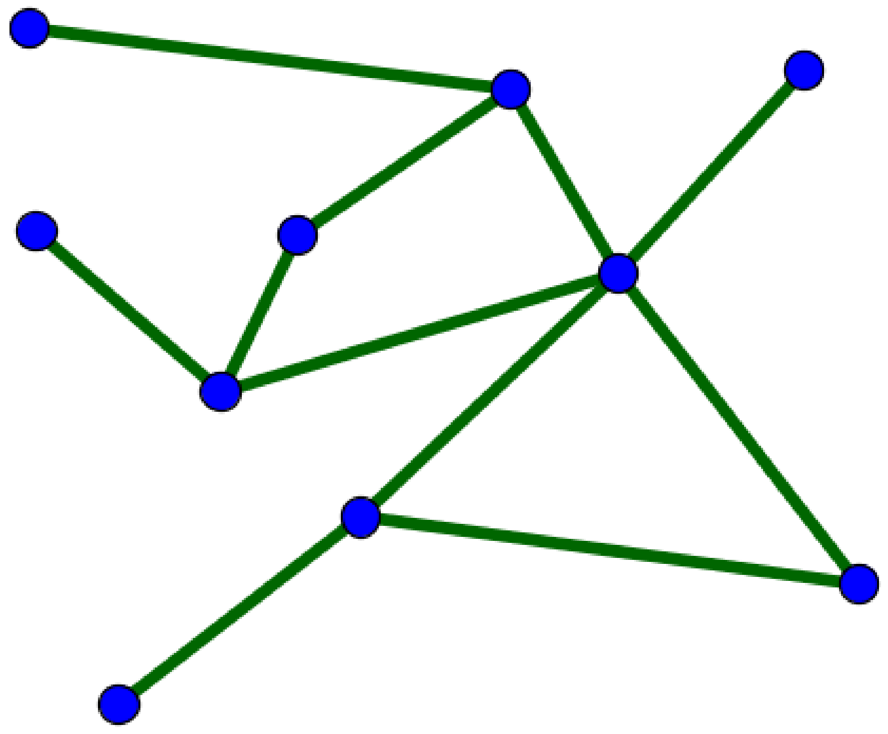
We consider a collection of $m \in \mathbb{N}$ agents with linear, discrete-time dynamics

$$
\begin{aligned}
x^i(t+1) &= A_i x^i(t) + B_i u^i(t), \quad x^i(0) = x_0^i, && \text{(1a)} \\
y^i(t) &= C_i x^i(t), && \text{(1b)}
\end{aligned}
$$

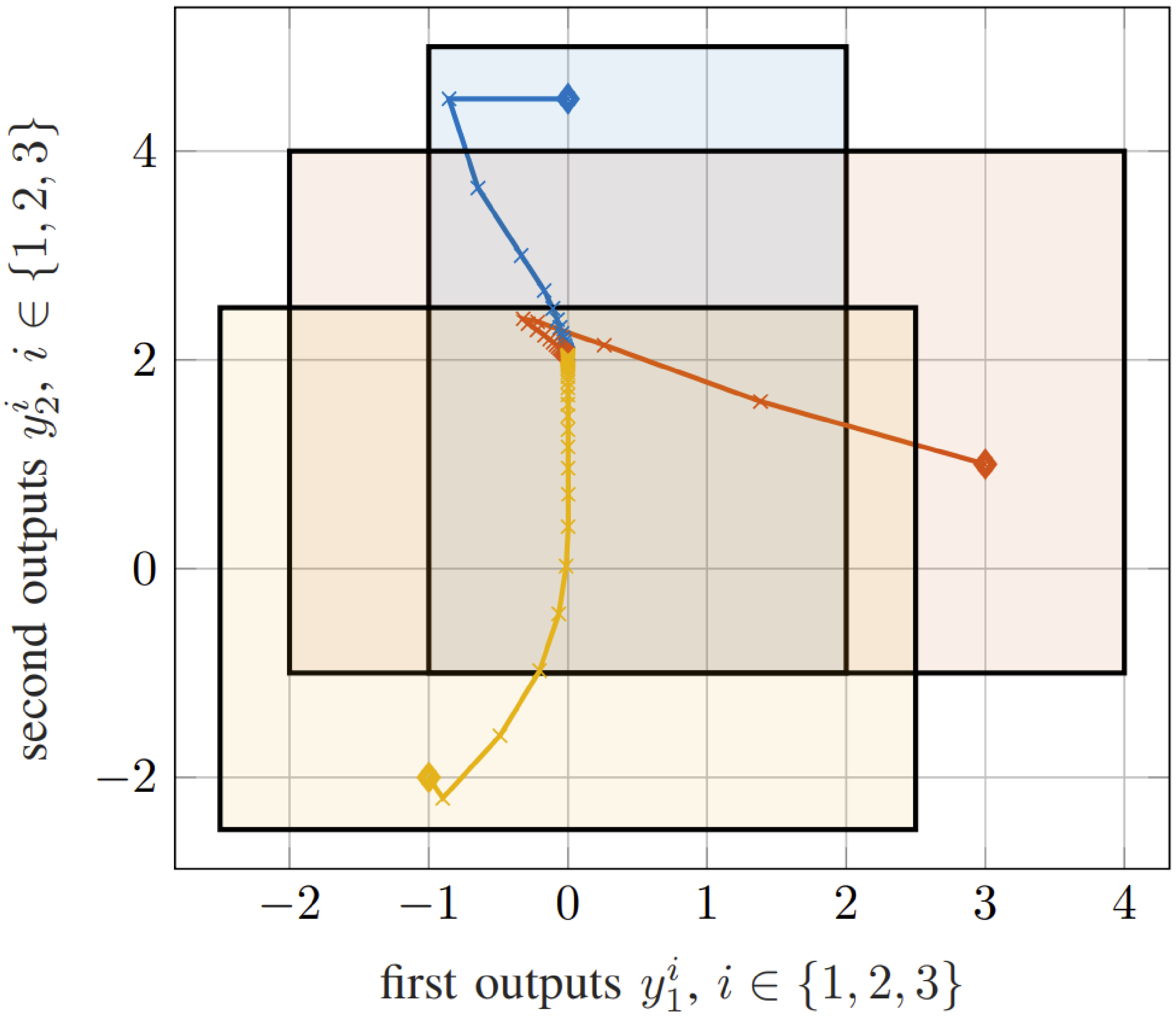
Multi-agent MPC

The communication topology of the multi-agent system is given by a connected, undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, with the set of nodes, i.e. agents, $\mathcal{V} = \{1, \dots, m\}$ and edges $\mathcal{E}$. Agents may bidirectionally share information with each other if they are connected by an edge. The induced set of neighbours of agent i is given as $\mathcal{N}_i = \{j \in \mathcal{V} \mid (i, j) \in \mathcal{E}\}$. The symmetric graph Laplacian is defined as $L = [l_{ij}] \in \mathbb{R}^{m \times m}$ where

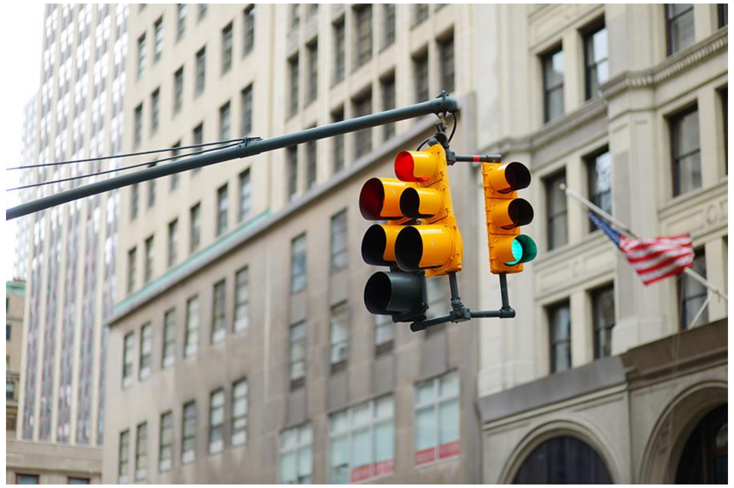
$$l_{ij} = \begin{cases} -1, & j \in \mathcal{N}_i, \\ |\mathcal{N}_i|, & j = i, \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

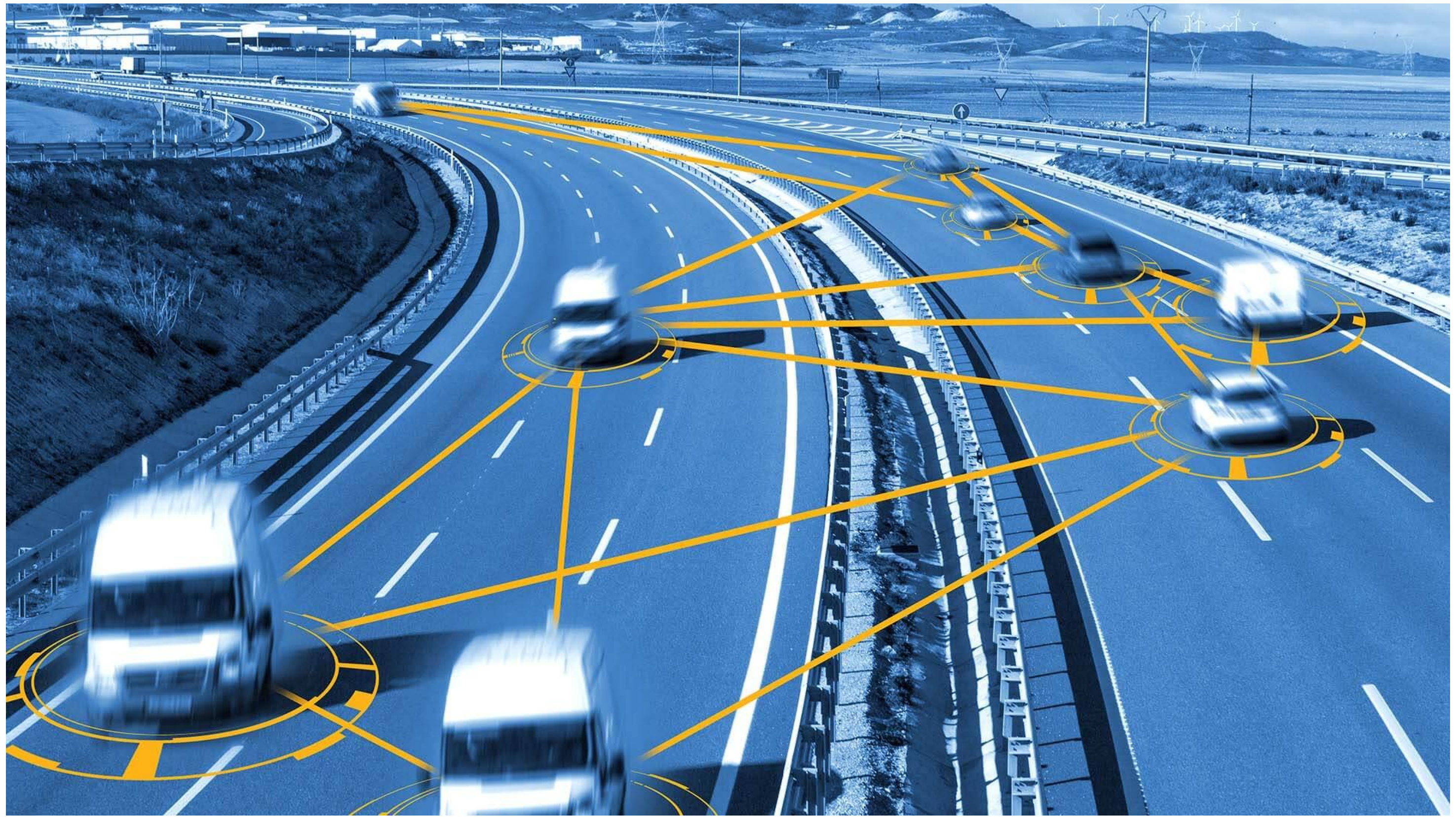


Multi-Agent MPC



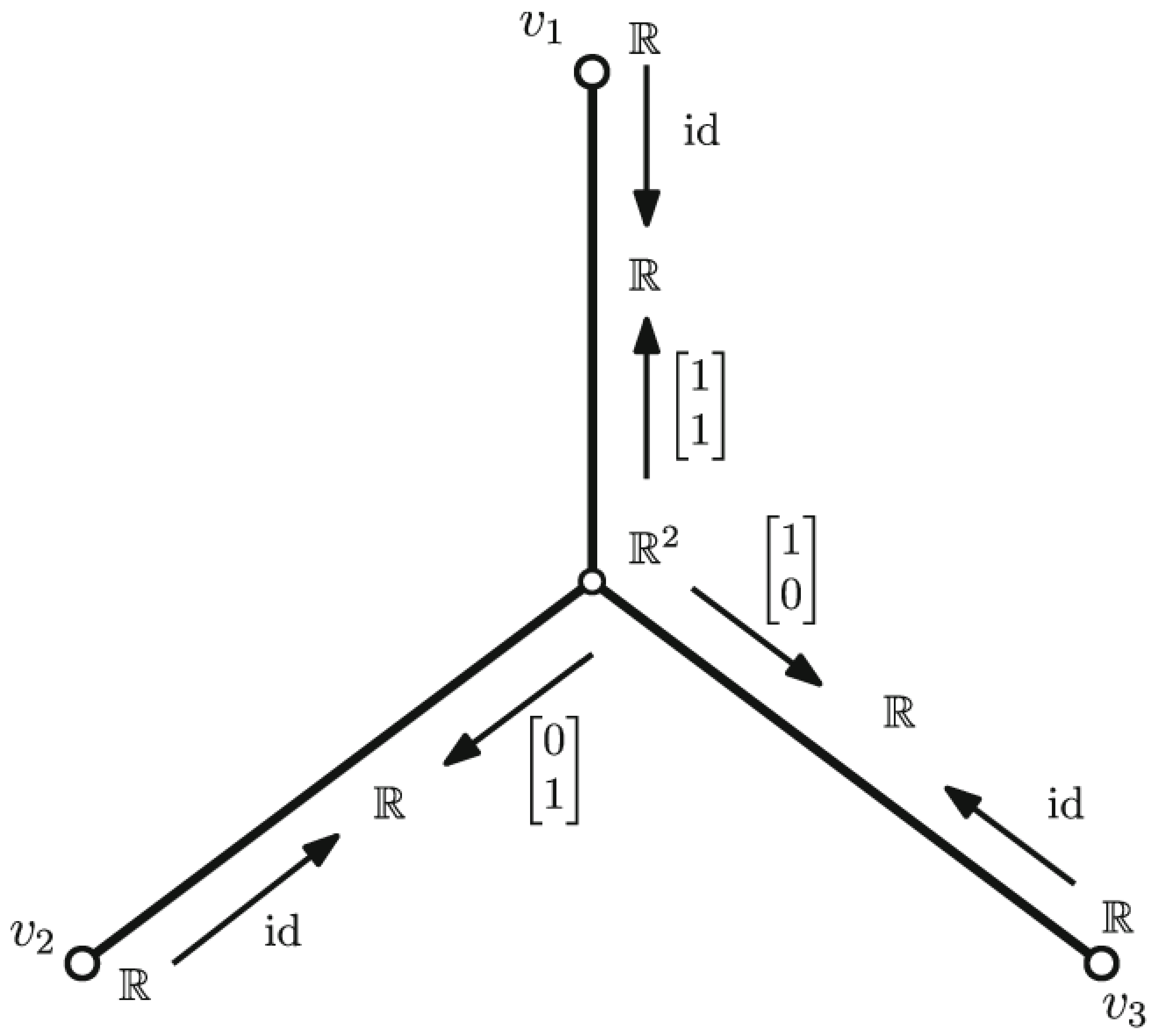
Multi-agent MPC





Cellular Sheaves

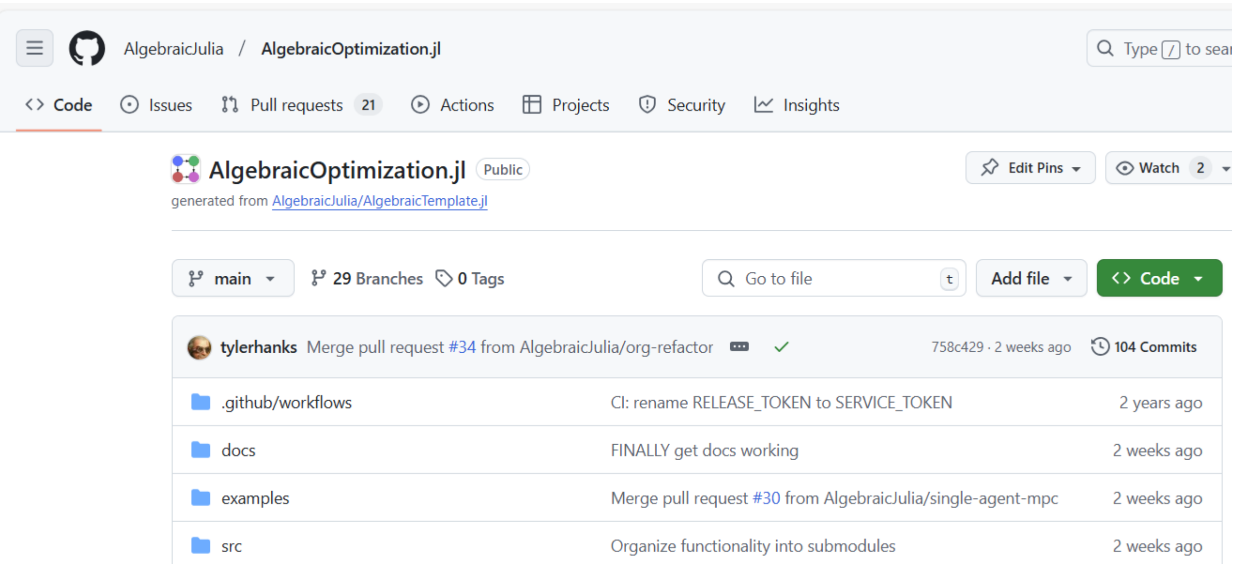
- Connected graph G = (V, E)
- Vertex and edge stalks
- Restriction maps
- Sheaf Laplacian

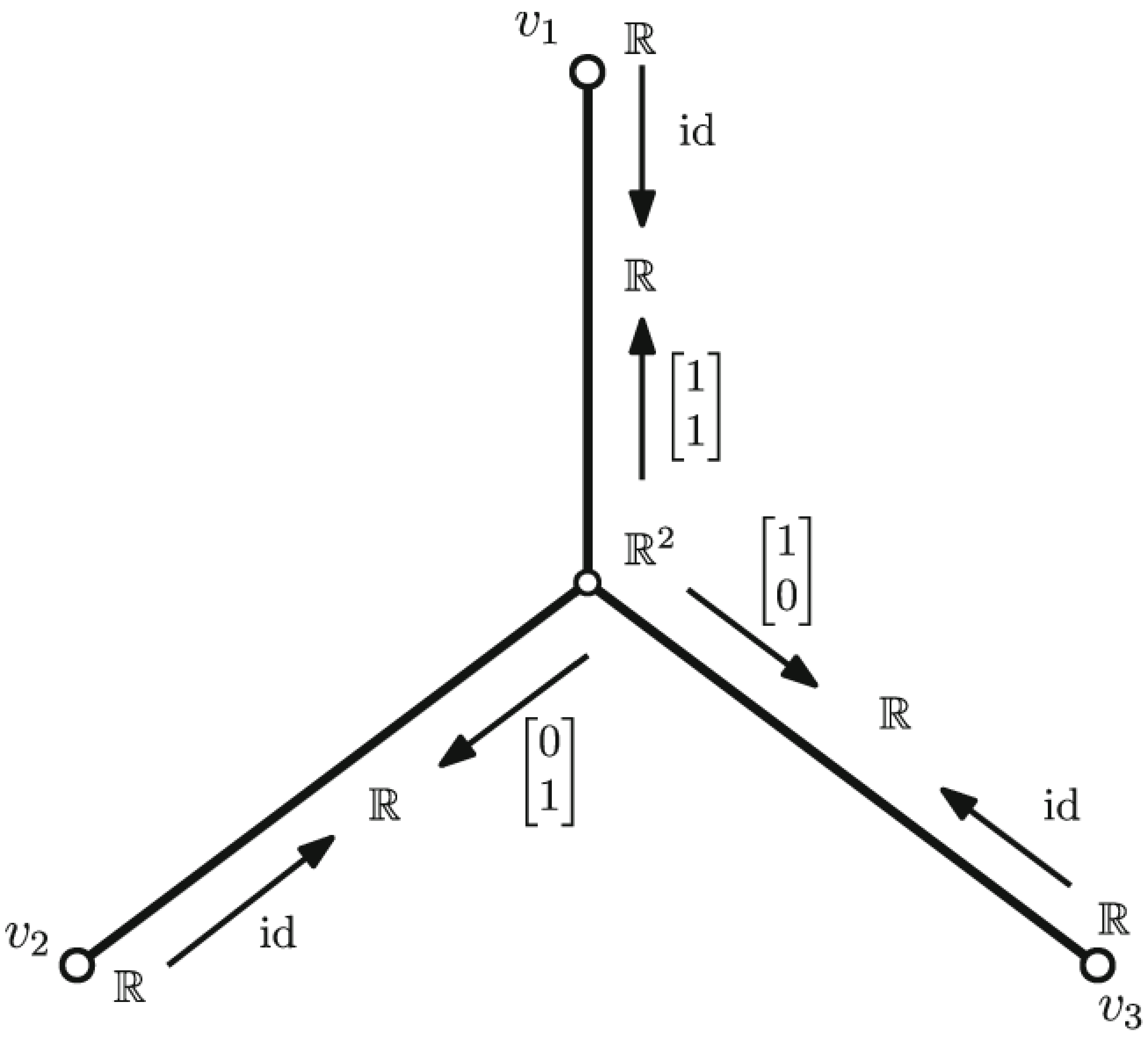


Cellular Sheaf Optimization

Minimize $\sum f(x)$

Subject to $Lx = 0$





A Sheaf Theoretic Framework for Multi-Agent Model Predictive Control

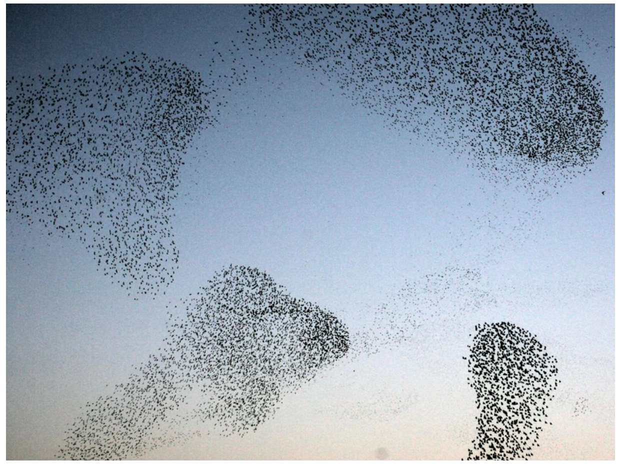
- We can do multi-agent MPC using cellular sheaves

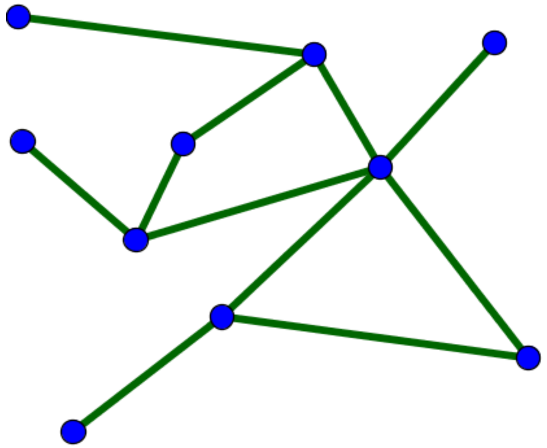
A. Flocking Problem

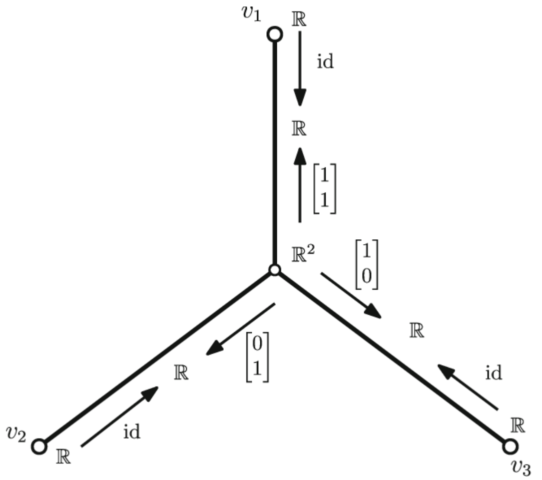
Develop a decentralized consensus protocol $u_i(k) = f_i(p_i(k), p_j(k), q_i(k), q_j(k)), j \in \mathcal{N}_i(k), i \in \mathcal{V}$, satisfying (2) and

$$\begin{aligned} &\lim_{k\to\infty} \|q_i(k) - q_j(k)\| = 0, \\ &\lim_{k\to\infty} \|p_i(k) - p_j(k)\| = 0, \quad \forall i, j \in \mathcal{V}. \end{aligned} \tag{4}$$

Note that "flocking" refers to a group of agents moving or migrating together with the same velocity.



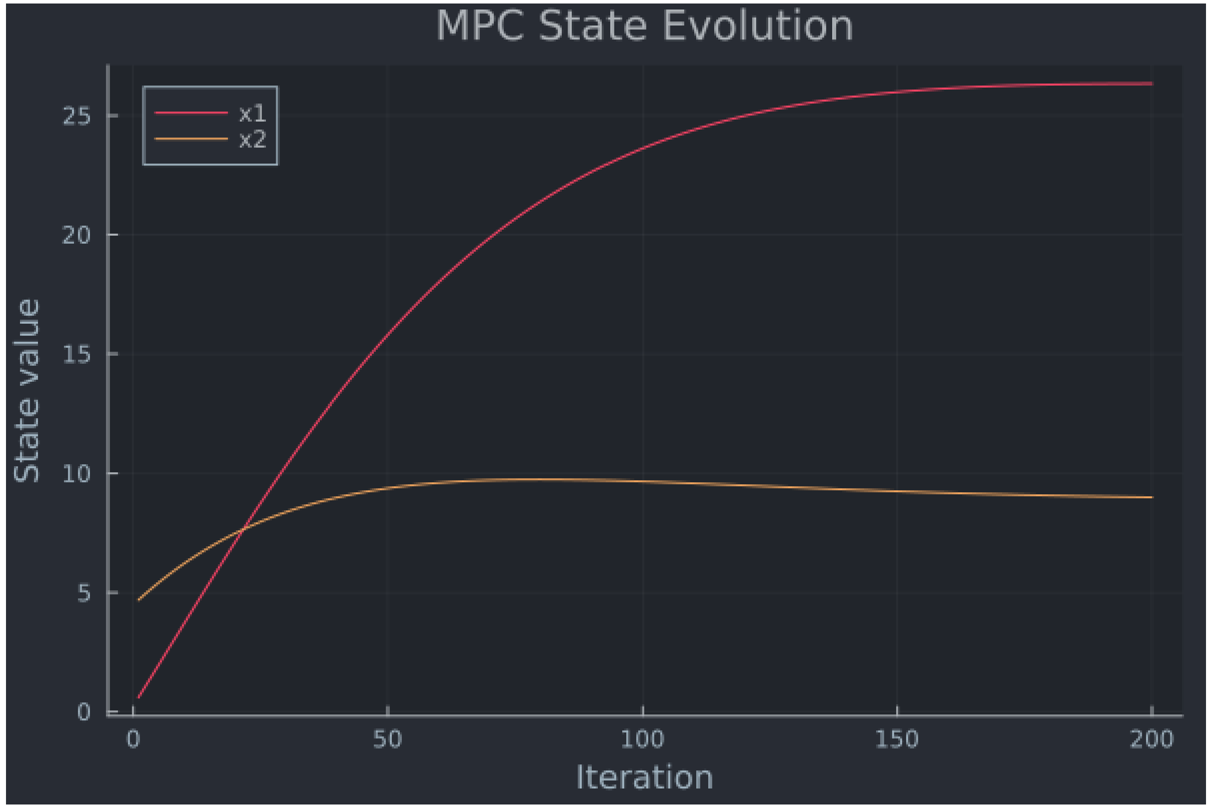




Implementation

Optimization

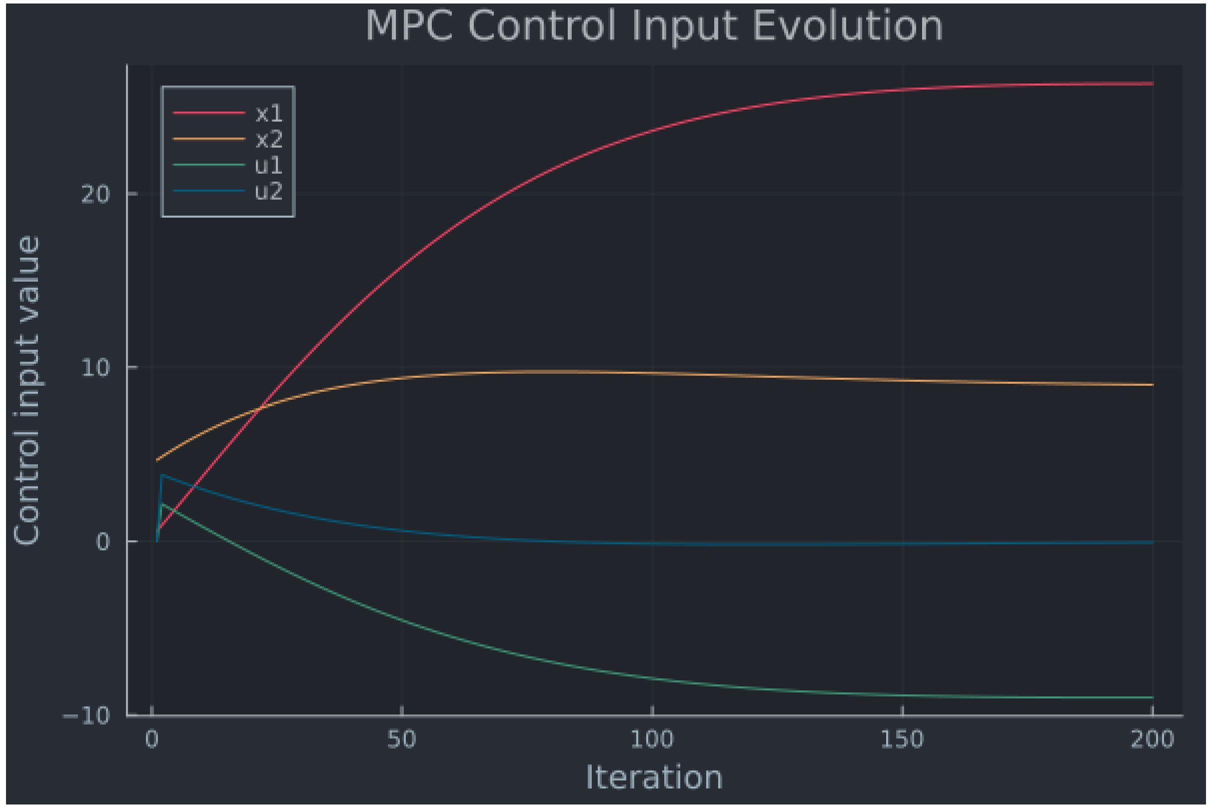
- Parameters:
  - Current state matrix, x_k, at time step k
  - Objective state weight matrix, Q
  - Objective control weight matrix, R
  - LinearSystem, s
    - Constraint state weight matrix, A
    - Constraint state weight matrix, B
  - Desired final position, x_{target}
- Constraints:
  - The change at each time step, $u_k \in [-20, 20]$
  - Maintain $x_{k+1} = s.Ax_k + s.Bu_k$
- Objective: **minimize distance** x_k and x_{target} across time steps $[k, k + horizon]$



The graph shows the changes in the x positions independently of each other throughout 200 iterations. Initial x1 = 0.590, initial x2 = 4.684

Iteration

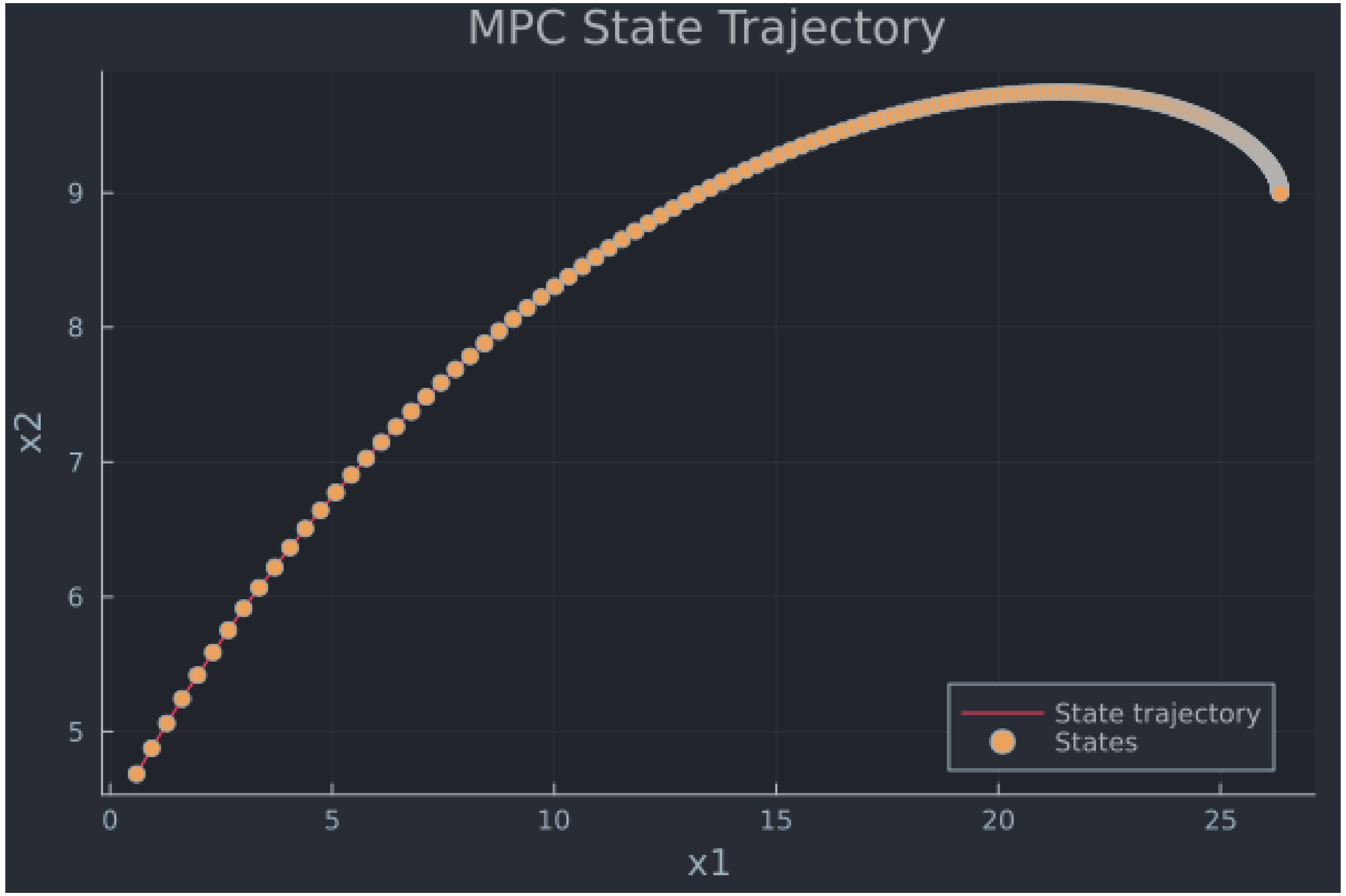
- Parameters:
  - Initial state matrix, x_0
  - Initial control matrix, u_0
  - LinearSystem, s
- Objective:
  - Generate weights and x_{target}
  - Run N optimization steps
  - Store each x_k and u_k in list
  - Plot results of state vs iteration



The graph shows the same information as the previous graph, with the addition of the controls independently of each other throughout the same 200 iterations. Initial x1 = 0.590, initial x2 = 4.684

Trajectory

- Smooth graph shows minimal (to no) oscillations
- Satisfies all constraints
- $x1$ and $x2$ converge to x_{target}
- u_k decrease each iteration



Up Next

Paper due March 31

Up Next

1. Finish programming basic cellular sheaf MPC example
2. Program cellular sheaf MPC for the flocking problem (or another example)
3. Write up results
4. Increase efficiency and compare to existing benchmarks (if time)
5. Submit to CDC!

Thanks for listening!