Lecture 14

Optimal Distributed Control

2) Projected GD for Locally Optimal Distributed Controllers

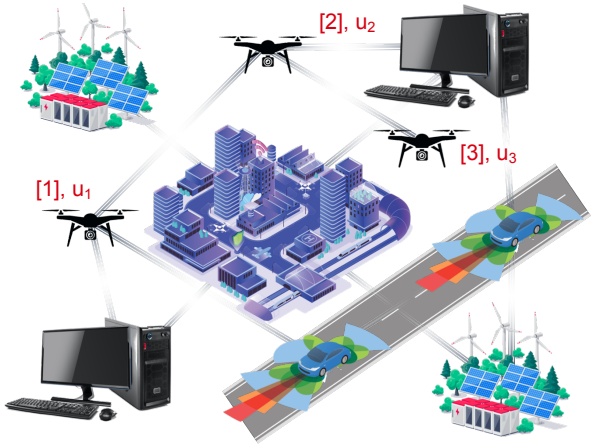
Giancarlo Ferrari Trecate[1]

[1]Dependable Control and Decision Group
École Polytechnique Fédérale de Lausanne (EPFL), Switzerland
giancarlo.ferraritrecate@epfl.ch

Slides: Luca Furieri

Recall the motivation

Optimal control at the large-scale



- Multiple, heterogeneous, dynamically coupled

$$x_{t+1}^{[1]} = \tau(x_t^{[2]})$$

- *Local* real-time measurements

Can we apply LQR/LQG? Not really...

- LQR controller :

$$u_t = \begin{bmatrix} u_t^{[1]} \\ u_t^{[2]} \\ u_t^{[3]} \end{bmatrix} = - \begin{bmatrix} k^{11} & k^{12} & k^{13} \\ k^{21} & k^{22} & k^{23} \\ k^{31} & k^{32} & k^{33} \end{bmatrix} \begin{bmatrix} x_t^{[1]} \\ x_t^{[2]} \\ x_t^{[3]} \end{bmatrix}$$

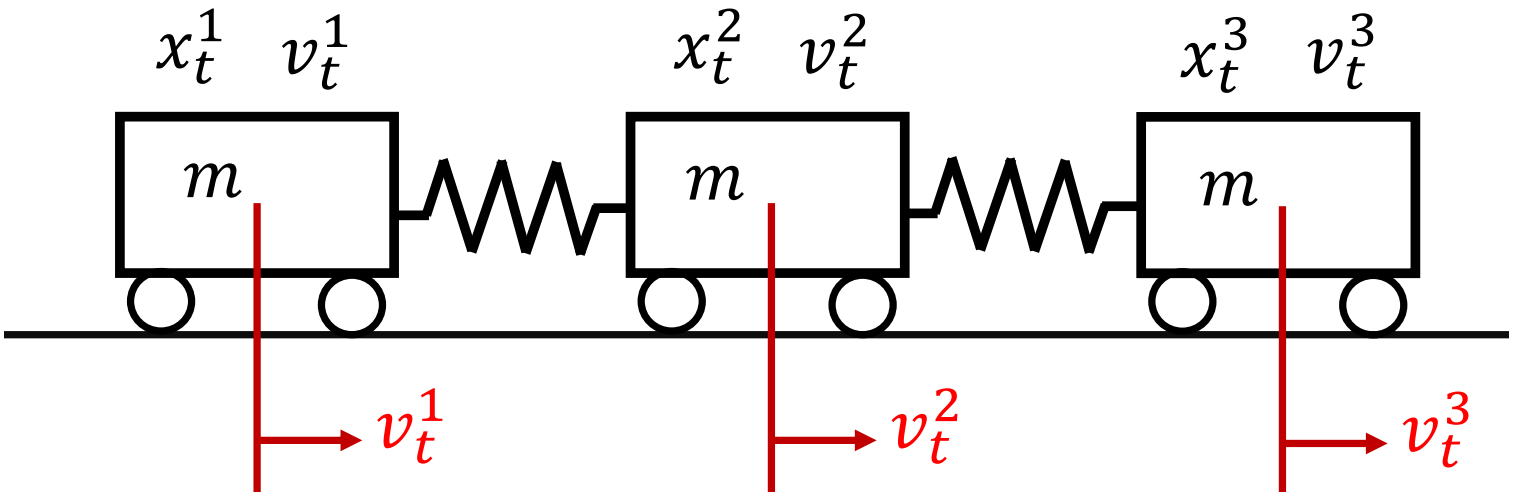
Not implementable! $u^{[3]}$ cannot measure $x^{[1]}$

$\Rightarrow u_t^{[3]} = k_{32} x_t^{[2]} + k_{33} x_t^{[3]}$

Plan of the lecture

1. **Projected GD for locally optimal distributed controllers**
   - Binary matrices, sparsity subspaces and Distributed S-LQR
   - Projected Gradient Descent (PGD) for DS-LQR
   - Convergence results

Motivating example



- Chain of mass-springs
  - (x_t^i, v_t^i) ="(pos, vel)" of mass i at time t
  - u_t^i = Force we apply at mass i at time t
- Overall, it is a 6-dimensional linear system

$$x_t = [x_t^1, v_t^1, x_t^2, v_t^2, x_t^3, v_t^3] \in \mathbb{R}^6$$

Motivating example

Goal

Find a controller K that minimizes $\mathbb{E}_{x_0}\left[\sum_{t=0}^{+\infty} x_t^\top x_t + 0.1 u_t^\top u_t\right]$. REQUIREMENT:

$$K = \begin{bmatrix} * & * & 0 & * & 0 & 0 \\ 0 & 0 & * & * & 0 & * \\ 0 & 0 & 0 & 0 & * & * \end{bmatrix}$$

Why is $K = \begin{bmatrix} * & * & 0 & * & 0 & 0 \\ 0 & 0 & * & * & 0 & * \\ 0 & 0 & 0 & 0 & * & * \end{bmatrix}$? $\quad x_t = \begin{bmatrix} x_t^1 & v_t^1 & x_t^2 & v_t^2 & x_t^3 & v_t^3 \end{bmatrix}$

Typical scenario: each mass/vehicle can only measure

1. its own position + speed;
2. the speed of the mass/vehicle in front of it.

That is:

- u_t^1 can only be a function of (x_t^1, v_t^1, v_t^2);
- u_t^2 can only be a function of (x_t^2, v_t^2, v_t^3);
- u_t^3 can only be a function of (x_t^3, v_t^3).

Since $u_t = -Kx_t$, we need that K is sparse as above.

Sparsity Subspaces

How do we describe sparsity requirements mathematically?

Sparsity

- Let $S \in \{0,1\}^{m \times n}$ be a BINARY MATRIX.
- We write $K \in \text{Sparse}(S)$ if and only if $K \in \mathbb{R}^{m \times n}$ has the same sparsity pattern encoded by S.

$$K \in \text{Sparse}(S) \iff S(i,j) = 0 \Rightarrow K(i,j) = 0$$

- $\text{Sparse}(S)$ is a SUBSPACE of $\mathbb{R}^{m \times n}$.

$$\forall K_1, K_2 \in \text{Sparse}(S), \alpha K_1 + \beta K_2 \in \text{Sparse}(S)$$

The distributed S-LQR problem

System

$$x_{t+1} = Ax_t + Bu_t$$
$$u_t = -Kx_t$$
$$x_0 \sim N(0, \Sigma_0)$$

Objective Function

$$J(K) = \mathbb{E}_{x_0}\left[\sum_{t=0}^{+\infty} x_t^\top Q x_t + u_t^\top R u_t\right]$$

where $Q \geq 0$, $R > 0$ and $\Sigma_0 = \mathbb{E}[x_0 x_0^\top] > 0$.

Information constraints

Binary matrix $S \in \{0,1\}^{m \times n}$

$S(i,j) = 0 \Rightarrow u_t(i)$ cannot measure $x_t(j)$ at any time.

DS-LQR problem

Find $\hat{K} = \arg\min_{K \in \mathbb{R}^{m \times n}} J(K)$ subject to $K \in \text{Sparse}(S)$. E.g.

$$S = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$ for mass-springs.

Challenges of DS-LQR

- We cannot use the S-LQR solution K^* for our mass-spring example

$$K^* = \begin{bmatrix} 1.9963 & 3.3700 & 0.4915 & 0.0482 & 0.0979 & 0.0253 \\ 0.4915 & 0.0482 & 1.6028 & 3.3471 & 0.4915 & 0.0482 \\ 0.0979 & 0.0253 & 0.4915 & 0.0482 & 1.9963 & 3.3700 \end{bmatrix}$$

- The optimal distributed controller may not even be a linear policy $u_t = -Kx_t$ with $K \in \text{Sparse}(S)$.
  $\Rightarrow$ For example, $u_t = -\tanh(Kx_t)$ may perform better (!!!)

Assumption: we look for the best **linear** sparse controller.

Naive approach

1. Compute the S-LQR controller

$$K^* = \begin{bmatrix} 1.9963 & 3.3700 & 0.4915 & 0.0482 & 0.0979 & 0.0253 \\ 0.4915 & 0.0482 & 1.6028 & 3.3471 & 0.4915 & 0.0482 \\ 0.0979 & 0.0253 & 0.4915 & 0.0482 & 1.9963 & 3.3700 \end{bmatrix}$$

2. Just set the undesired entries equal to 0

$$\hat{K} = \begin{bmatrix} 1.9963 & 3.3700 & \cancel{0.4915} & 0.0482 & \cancel{0.0979} & \cancel{0.0253} \\ \cancel{0.4915} & \cancel{0.0482} & 1.6028 & 3.3471 & \cancel{0.4915} & 0.0482 \\ \cancel{0.0979} & \cancel{0.0253} & \cancel{0.4915} & \cancel{0.0482} & 1.9963 & 3.3700 \end{bmatrix}$$

Intuition : K^* is the best controller... let us preserve the optimal values wherever we can.

The naive approach fails in general

- There are examples where,
  - for K^* full, $A - BK^*$ is Schur
  - for $\hat{K} \in \text{Sparse}(S)$ obtained by zeroing the elements in K^*, $A - B\hat{K}$ is not Schur
- Even if we don't ruin stability, performance may degrade a lot

Can we use Gradient Descent instead?

Assume $K_0 \in \text{Sparse}(S)$ is given. GD :

$$K_1 = \underbrace{K_0}_{\in \text{Sparse}(S)} - \eta \underbrace{\nabla J(K_0)}_{\notin \text{Sparse}(S)}$$

Why is $\nabla J(K_0)$ not in Sparse(S)? It depends on P_{K_0}, Σ_0^{CL} which are dense, in general.

Hence, $K_1 \notin \text{Sparse}(S)$.

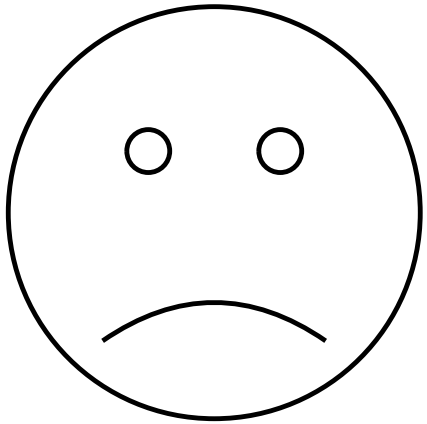
Can we use Gradient Descent instead?

Assume $K_0 \in \text{Sparse}(S)$ is given. GD :

$$K_1 = \underbrace{K_0}_{\in \text{Sparse}(S)} - \eta \underbrace{\nabla J(K_0)}_{\notin \text{Sparse}(S)}$$

Why is $\nabla J(K_0)$ not in Sparse(S)? It depends on P_{K_0}, Σ_0^{CL} which are dense, in general.

Hence, $K_1 \notin \text{Sparse}(S)$.



Your company ControlX® uses a very inefficient $K_0 \in \text{Sparse}(S)$... Even if you know $\nabla J(K_0)$, following the gradient to improve the cost ruins the sparsity.

Plan of the lecture

1. **Projected GD for locally optimal distributed controllers**
   - Binary matrices, sparsity subspaces and Distributed S-LQR
   - Projected Gradient Descent (PGD) for DS-LQR
   - Convergence results

Projected Gradient Descent

Projected Gradient Descent

Given a continuously differentiable function $f : \mathbb{R}^n \to \mathbb{R}$, an affine subspace $\mathcal{X} \subseteq \mathbb{R}^n$, and a sufficiently small step-size $\eta > 0$, if we assume that $x_0 \in \mathcal{X}$, then, the PGD algorithm

$$x_{j+1} = x_j + \eta \Pi_x(- \nabla f(x_j))$$

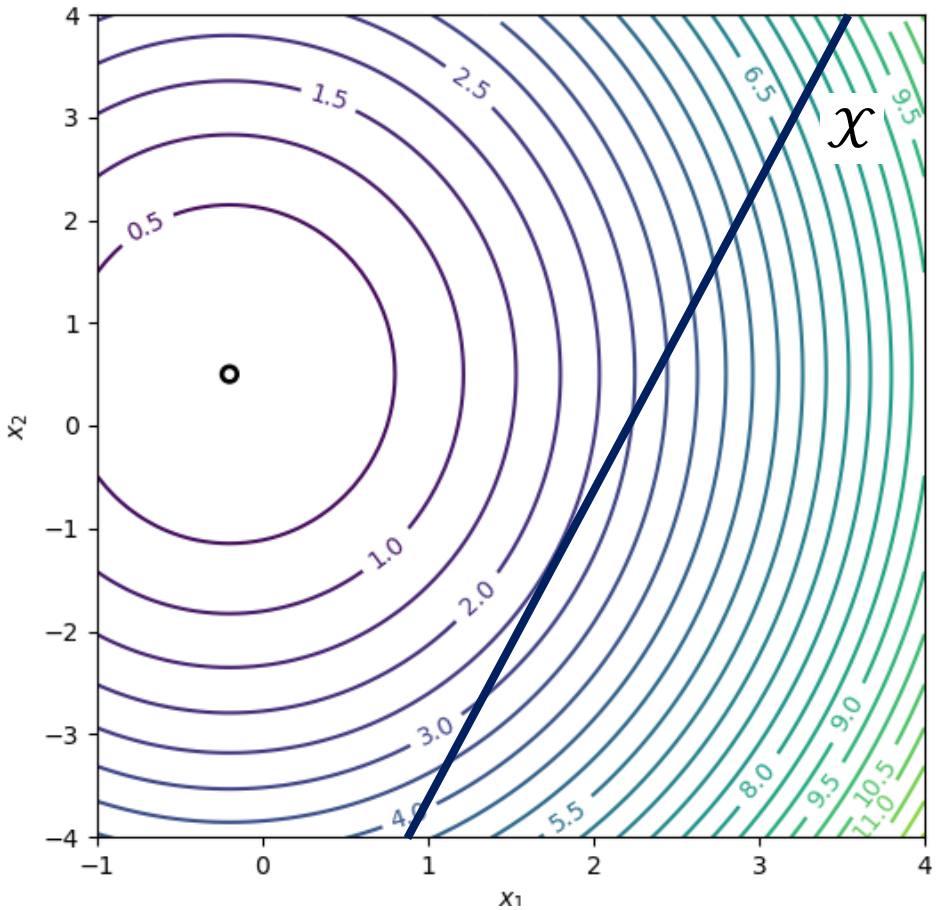
where $\Pi_x(\cdot)$ denotes the projection onto $\mathcal{X}$, converges to a point $\bar{x} \in \mathcal{X}$ such that $\Pi_x(\nabla f(\bar{x})) = 0$, or equivalently

$$\lim_{j \to \infty} \underbrace{\nabla f(x_j)^\top v}_{\text{scalar product equal to } 0 \equiv \nabla f(x_\infty) \perp v,\ \forall v \in \mathcal{X}} = 0, \quad \forall v \in \mathcal{X}$$

Example: Projected GD

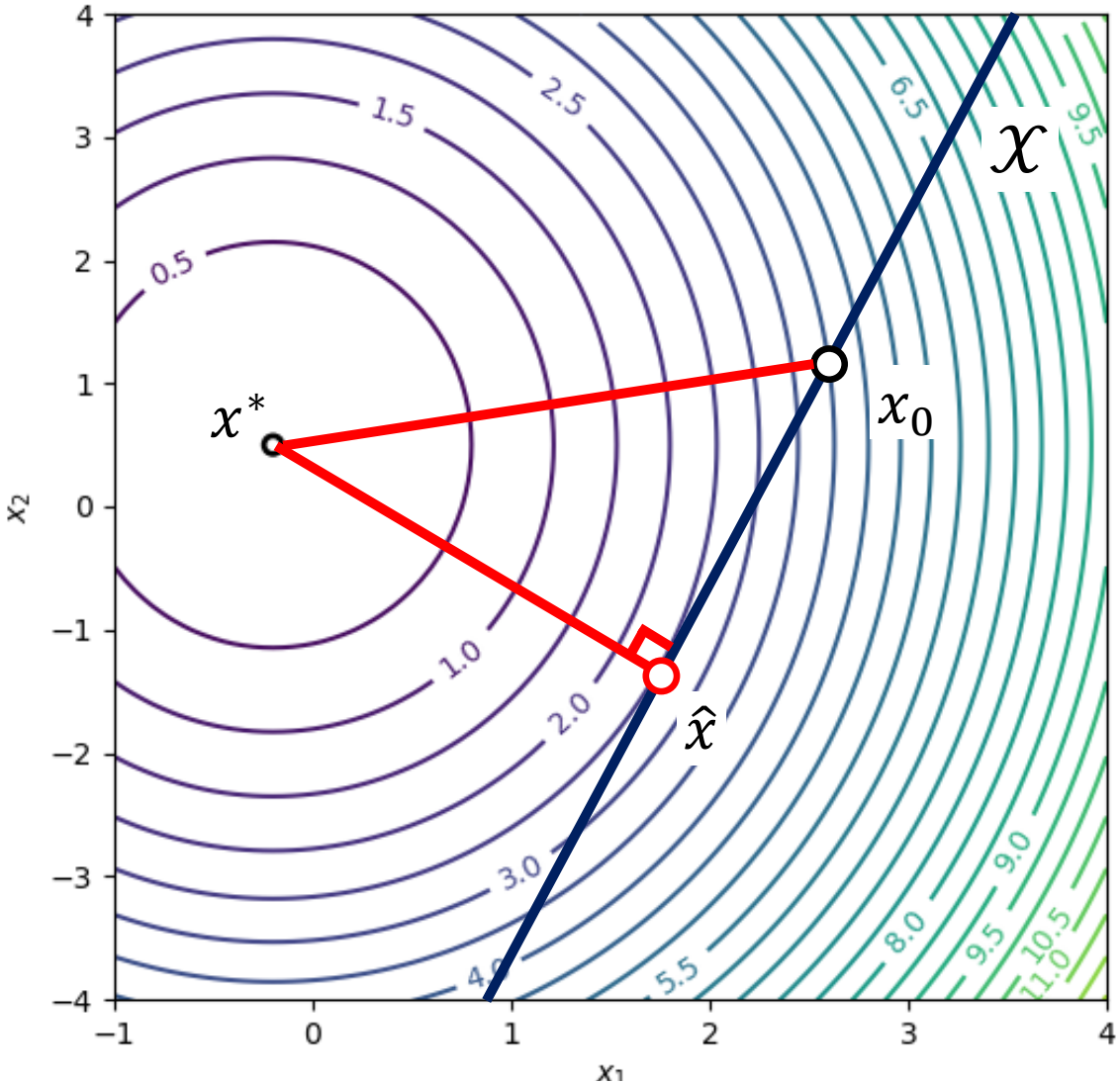
Goal

Find $\bar{x} \in \mathcal{X}$ that minimizes $f(\cdot)$ using PGD.



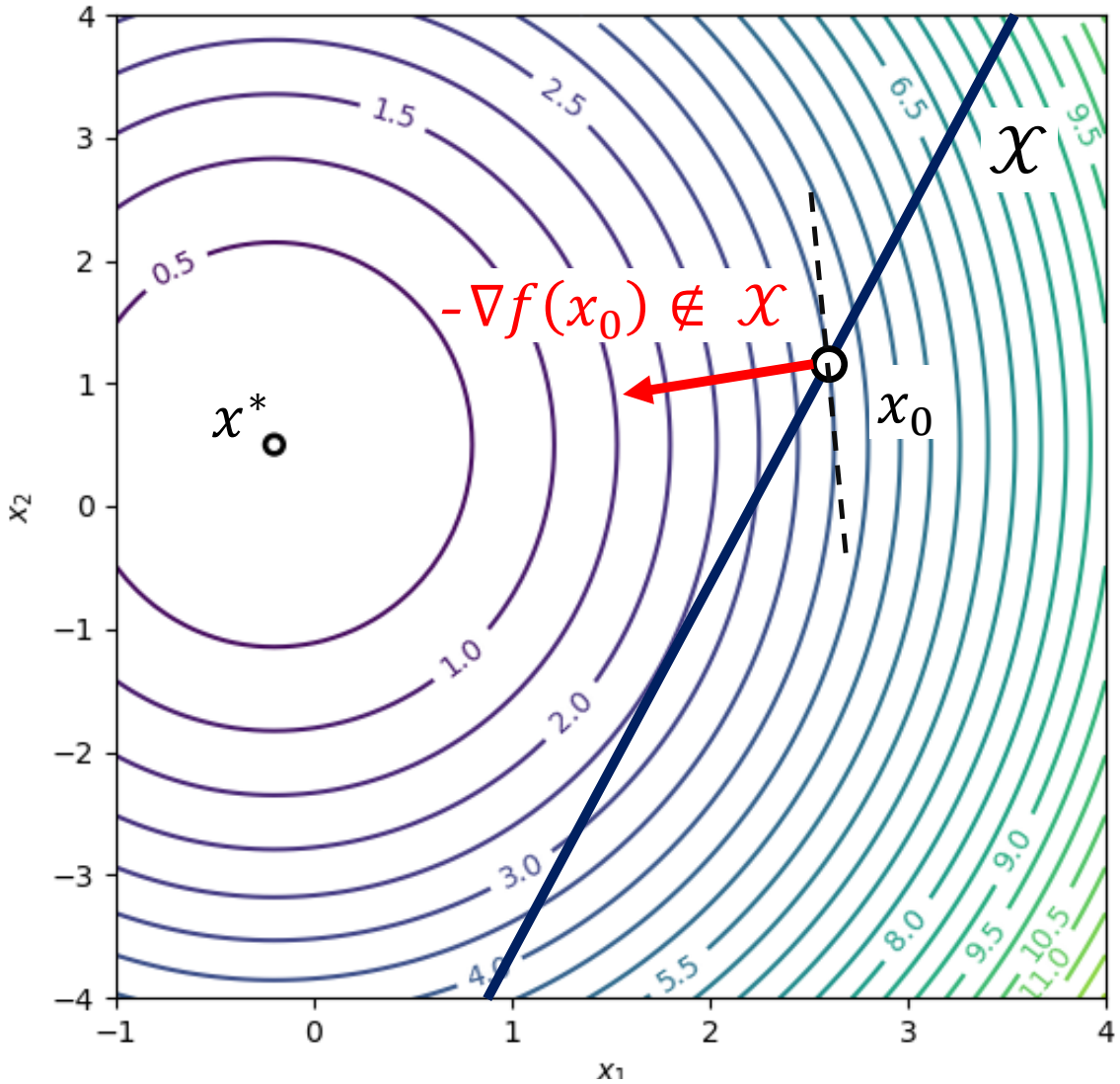
- $f(x_1, x_2) = (x_1 + 0.2)^2 + (x_2 - 0.5)^2$
- $\mathcal{X}$ is an (affine) subspace

Example: Projected GD

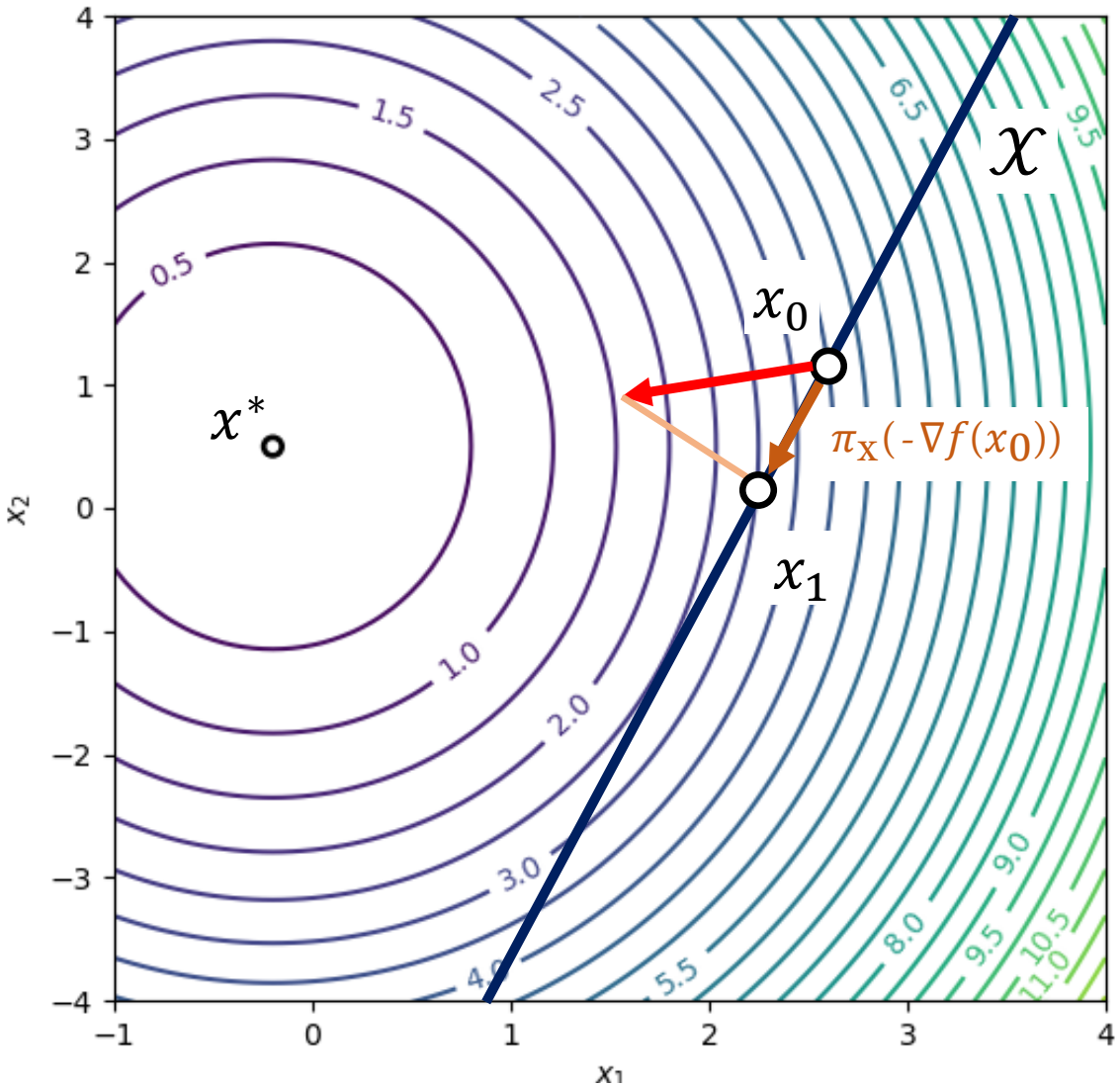


- x^* global optimum
- $\bar{x}$ optimum subject to $\bar{x} \in \mathcal{X}$

Example: Projected GD



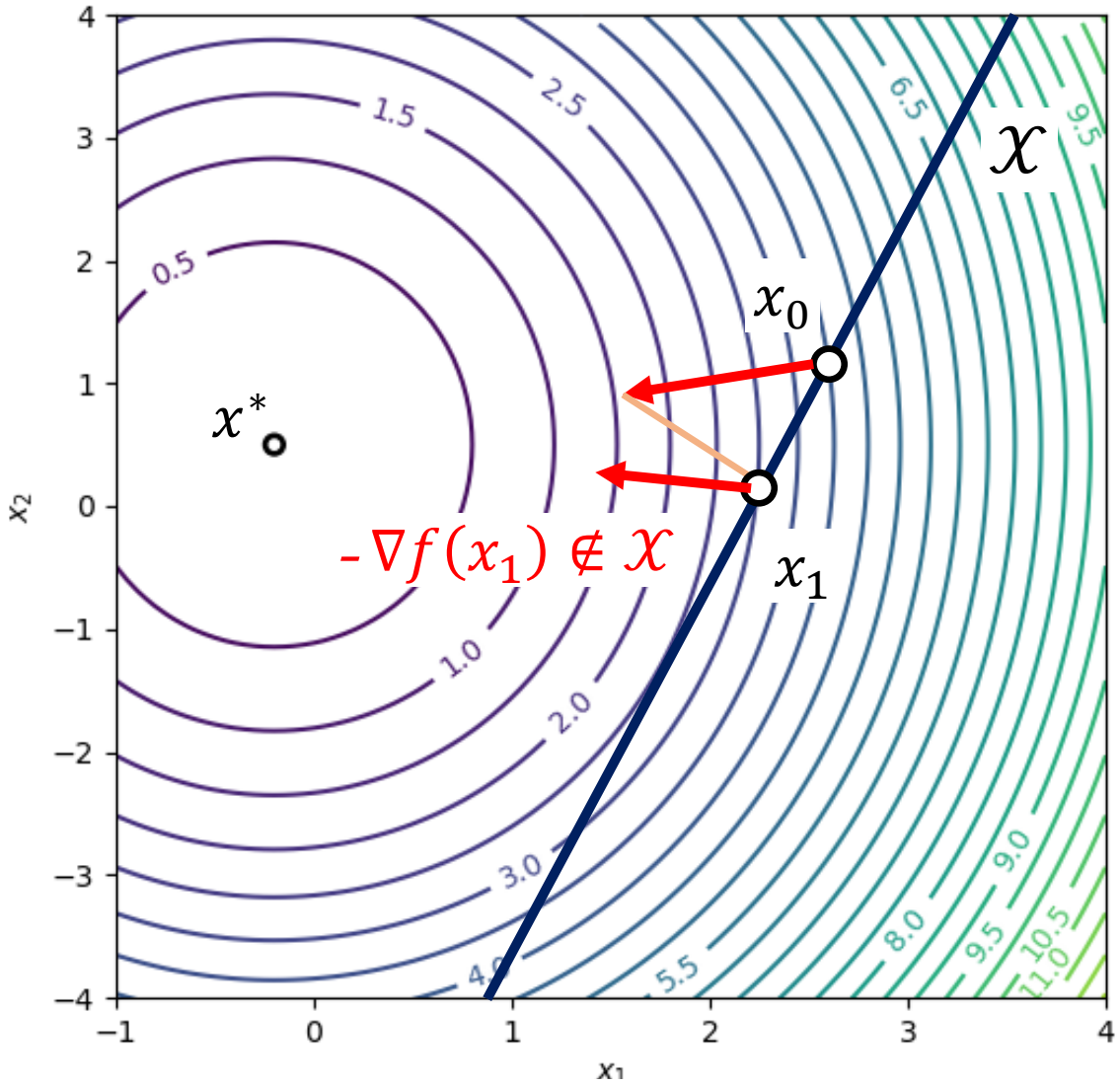
Example: Projected GD



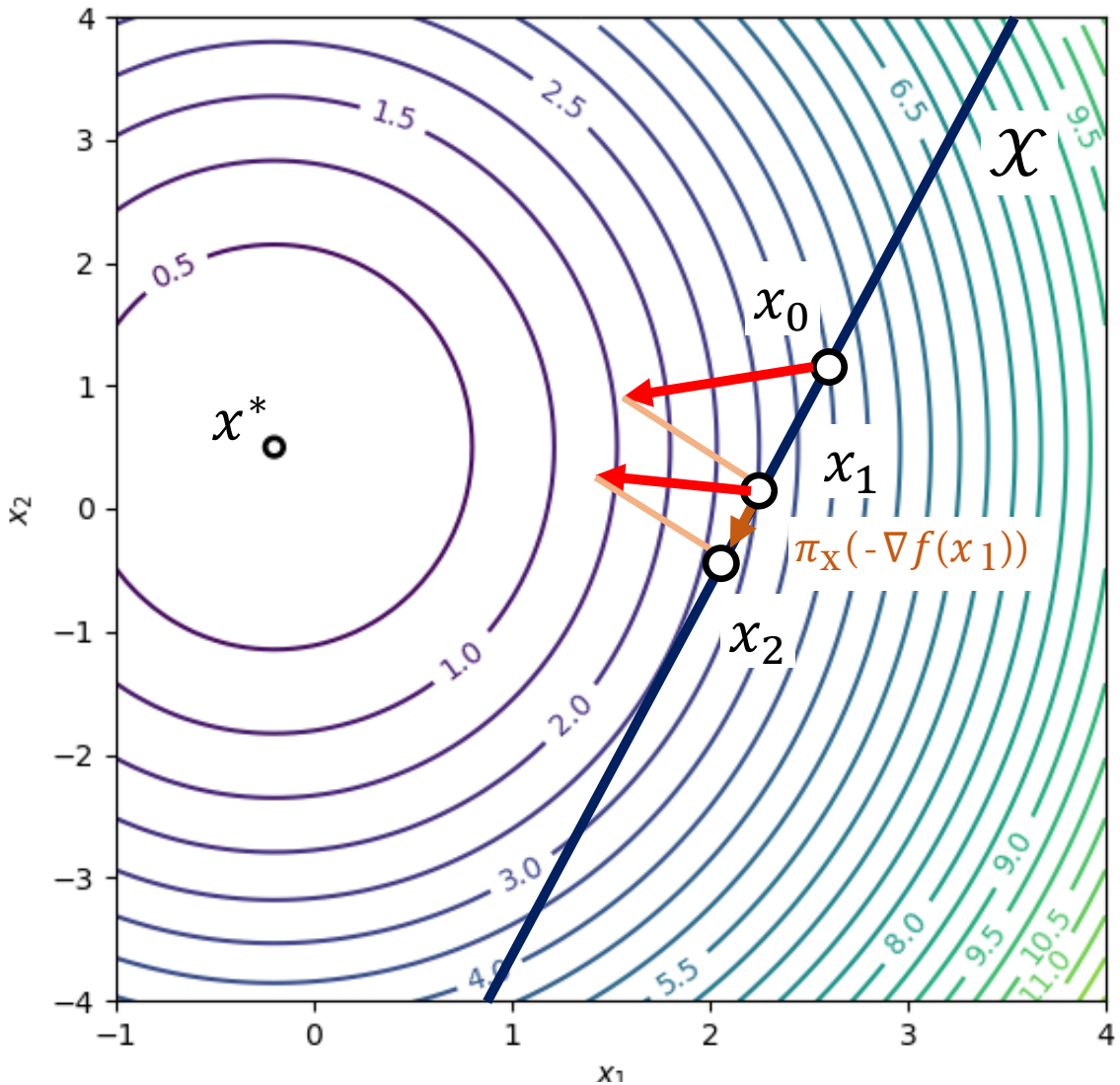
- $\Pi_x(-\nabla f(x_0))$ is the orthogonal projection of $-\nabla f(x_0)$ onto $\mathcal{X}$.
- Now, $x_1 = x_0 + \eta \Pi_x(-\nabla f(x_0)) \in \mathcal{X}$ again !

Let's go on...

Example: Projected GD

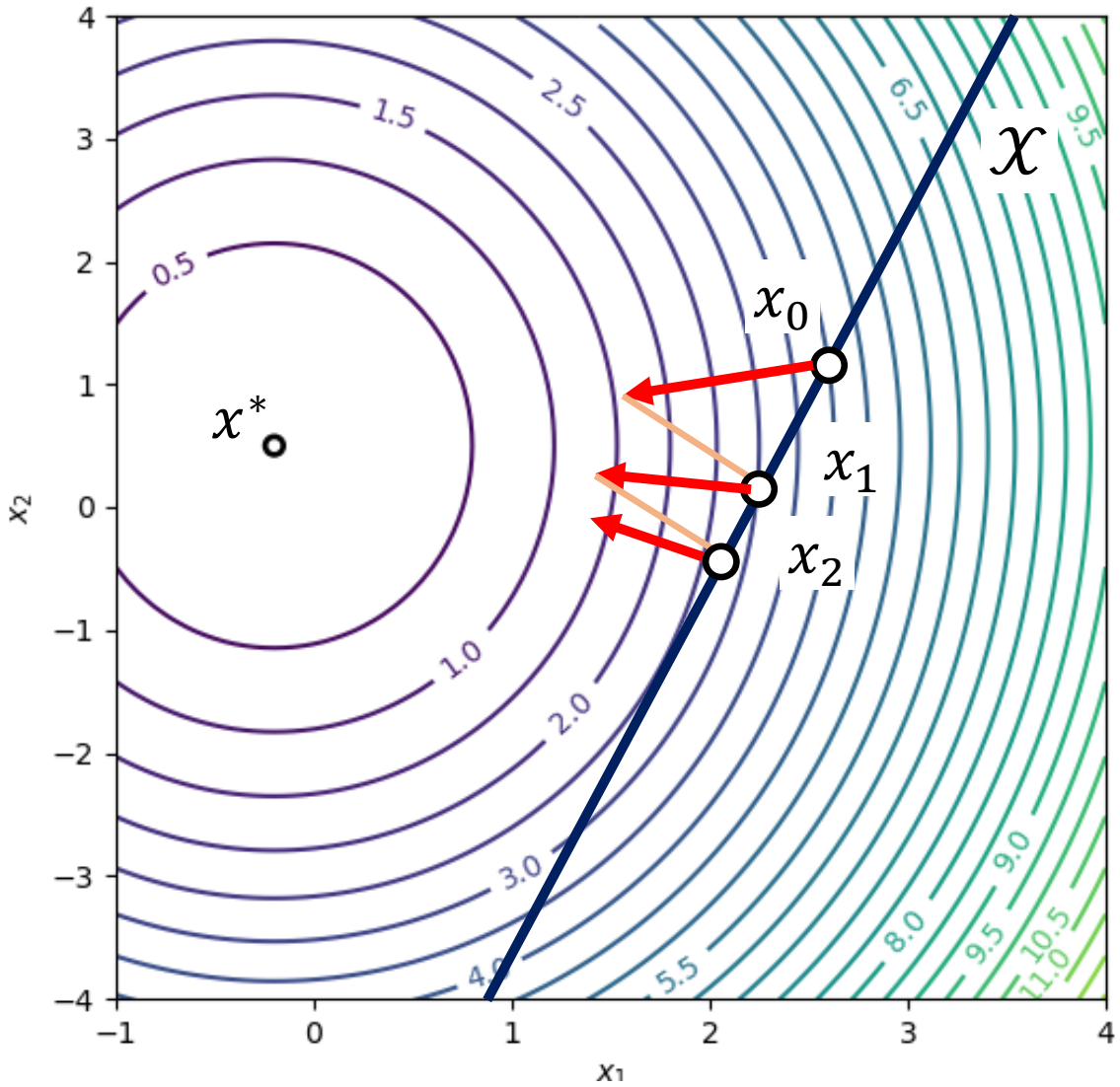


Example: Proiected GD

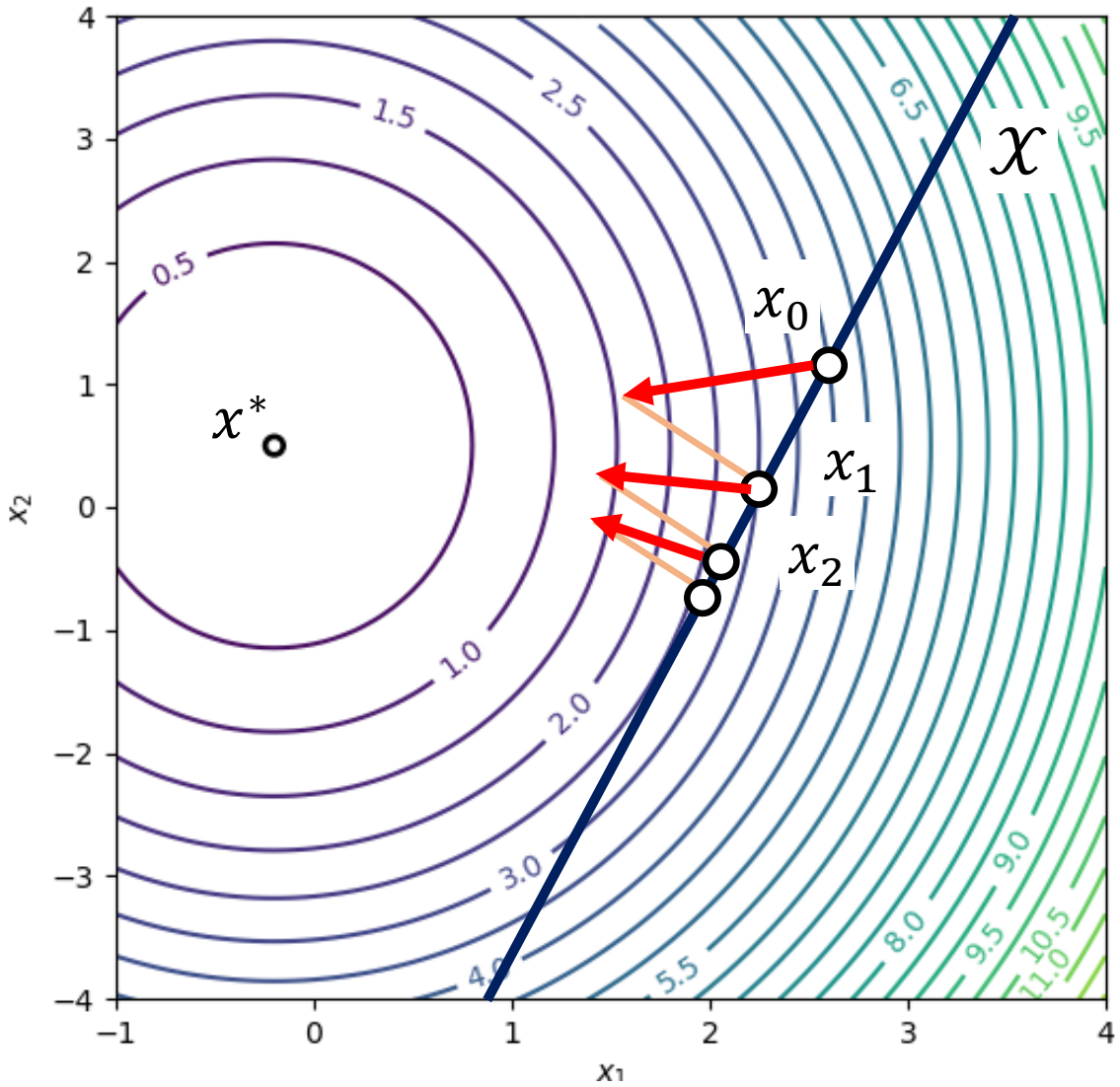


$$x_2 = x_1 + \eta \Pi_{\mathcal{X}}\left(-\nabla f(x_0)\right) \in \mathcal{X}! \ \ldots$$

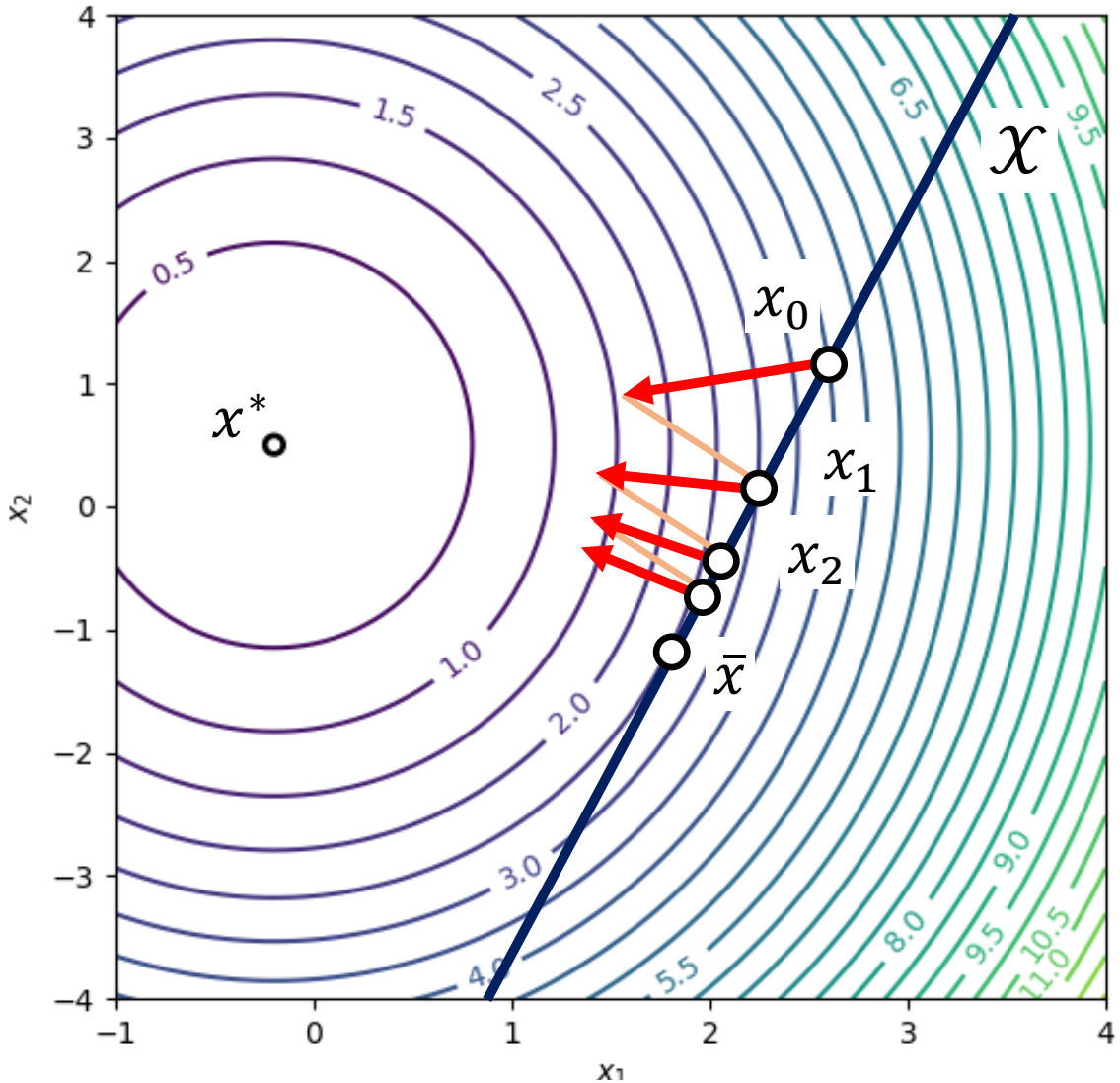
Example: Projected GD



Example: Projected GD

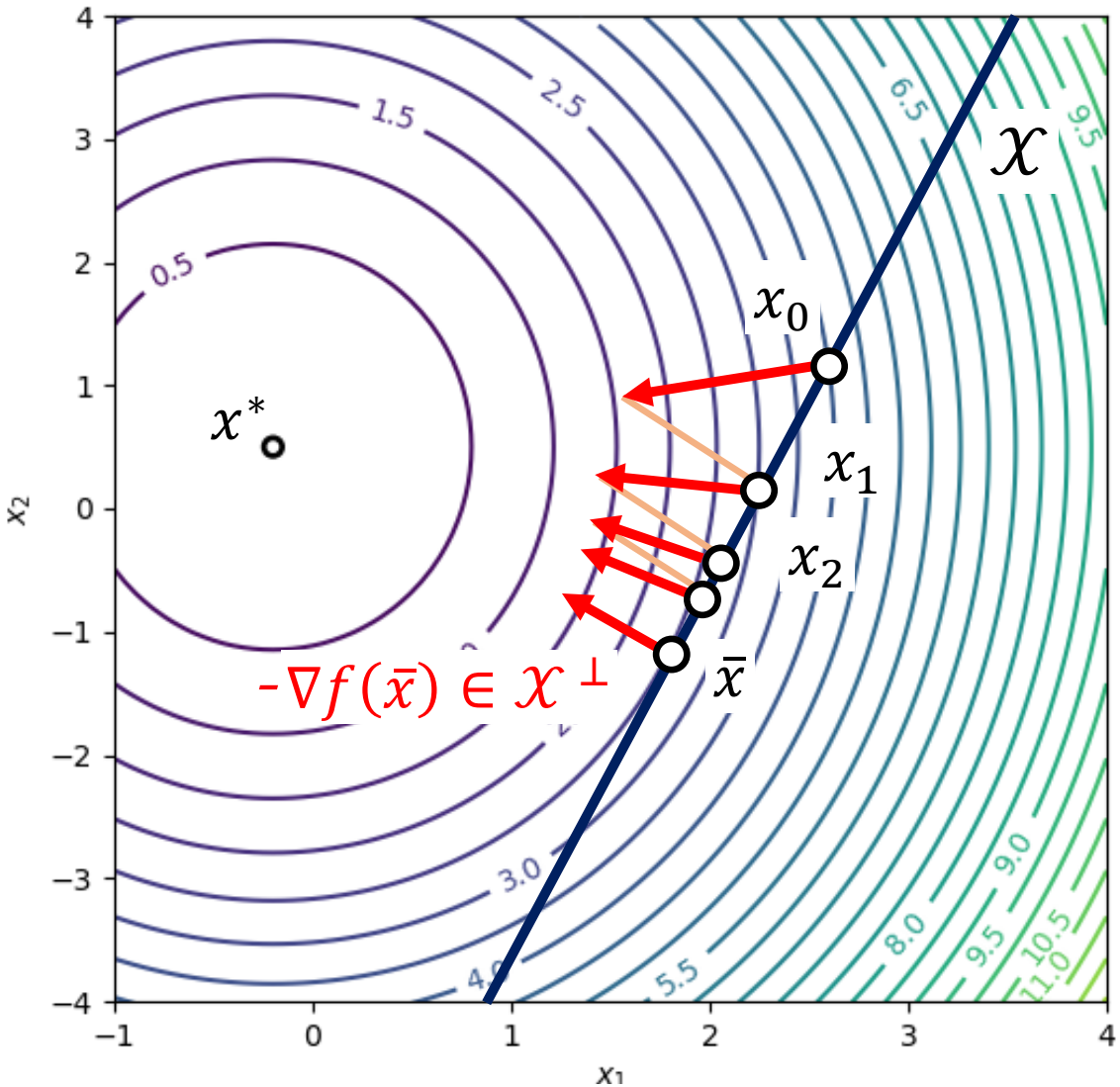


Example: Proiected GD



Smaller and smaller improvements. . .

Example: Projected GD



- After many steps . . . We are stuck! Why?
- $\nabla f(\bar{x}) \in \mathcal{X}^\perp$ i.e. $\forall v \in \mathcal{X},\ \nabla f(\bar{x})^\top v = 0$

PGD has converged to a point where $\nabla f(\bar{x}) \neq 0$, but $\Pi_x\left(\nabla f(\bar{x})\right) = 0$

Projected Gradient Descent

Remark

In this case, PGD converges to

$$\bar{x} = \arg\min_{x} f(x) \quad \text{subject to } x \in \mathcal{X}$$

That is, the GLOBAL OPTIMUM of the constrained problem. This is possible because $f(x_1, x_2) = (x_1 + 0.2)^2 + (x_2 - 0.5)^2$ is strongly convex.
$\Rightarrow$ in general, $\Pi_x(\nabla f(\bar{x})) = 0 \not\Rightarrow$ GLOBAL OPTIMUM of the constrained problem.
(Spoiler) $\Rightarrow$ unfortunately, DS-LQR can have many local minima.

PGD for DS-LQR

- We know how to compute $\nabla J(K)$ for any $K \in \text{Sparse}(S)$.
  - However, $\nabla J(K) \notin \text{Sparse}(S)$

PGD

Same as GD, but

$$K_{j+1} = \underbrace{K_j}_{\in \text{Sparse}(S)} - \eta \underbrace{\Pi_{\text{Sparse}(S)}(\nabla J(K_j))}_{\in \text{Sparse}(S)}$$

$\Rightarrow K_{j+1} \in \text{Sparse}(S)$!

By induction, $K_j \in \text{Sparse}(S)$ for all $j = 0, \ldots, \infty$.

PGD for DS-LQR

Questions

Q1) How to compute $\Pi_{\text{Sparse}(S)}(\nabla J(K_j))$?

Q2a) Does PGD converge?

Q2b) If yes, does it converge to a global optimum of the DS-LQR problem?

PGD for DQ-LQR

Q1) Projection onto Sparse(S)

By definition

$$\Pi_x(v) = \arg\min_y \|y - v\|_2 \quad \text{subject to } y \in \mathcal{X}$$

"closest point to v living in $\mathcal{X}$".

PGD for DQ-LQR

Q1) Projection onto Sparse(S)

Computing the projection is very easy for sparsity subspaces

Example

$$S = \begin{bmatrix} 1 \\ 0 \end{bmatrix},$$

$$\Pi_{\text{Sparse}(S)}\left(\begin{bmatrix} a \\ b \end{bmatrix}\right) = \underset{y_1, y_2 \text{ subject to } y_2 = 0}{\arg\min} \left\| \begin{bmatrix} a \\ b \end{bmatrix} - \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \right\|_2$$

$$= \arg\min \left\| \begin{bmatrix} a - y_1 \\ b \end{bmatrix} \right\|_2$$

The solution is $y_1 = a$, $y_2 = 0$!

$\Rightarrow$ Simply set "undesired entries" to 0.

$$S = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}, K = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \Rightarrow \Pi_{\text{Sparse}(S)}(K) = \begin{bmatrix} 1 & 0 \\ 3 & 4 \end{bmatrix}.$$

PGD for DQ-LQR

Q1) Projection onto Sparse(S)

We have that

Projection onto Sparse(S)

$$\Pi_{\text{Sparse}(S)}\left(\nabla J(K)\right) = \nabla J(K) \odot S$$

where $\odot$ denotes the element-wise product (Hadamard product)

Example

$$S = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}, \quad \nabla J(K) = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

$$\Pi_{\text{Sparse}(S)}(\nabla J(K)) = \nabla J(K) \odot S = \begin{bmatrix} a & 0 \\ c & d \end{bmatrix}.$$

$\Rightarrow$ " .*" in MatLab

PGD for DQ-LQR

Q1) Projection onto Sparse(S)

Algorithm

INPUT : $K_0 \in \text{Sparse}(S)$, $(A - BK_0)$ is stable, (A, B, Q, R, Σ_0), $\eta > 0$, tolerance $\epsilon > 0$

$K = K_0$

while $(\|\nabla J(K) \odot S\| > \epsilon)$:

  Compute $\nabla J(K) = 2F_K \Sigma_K^{CL}$

  $K \leftarrow K - \eta \nabla J(K) \odot S$

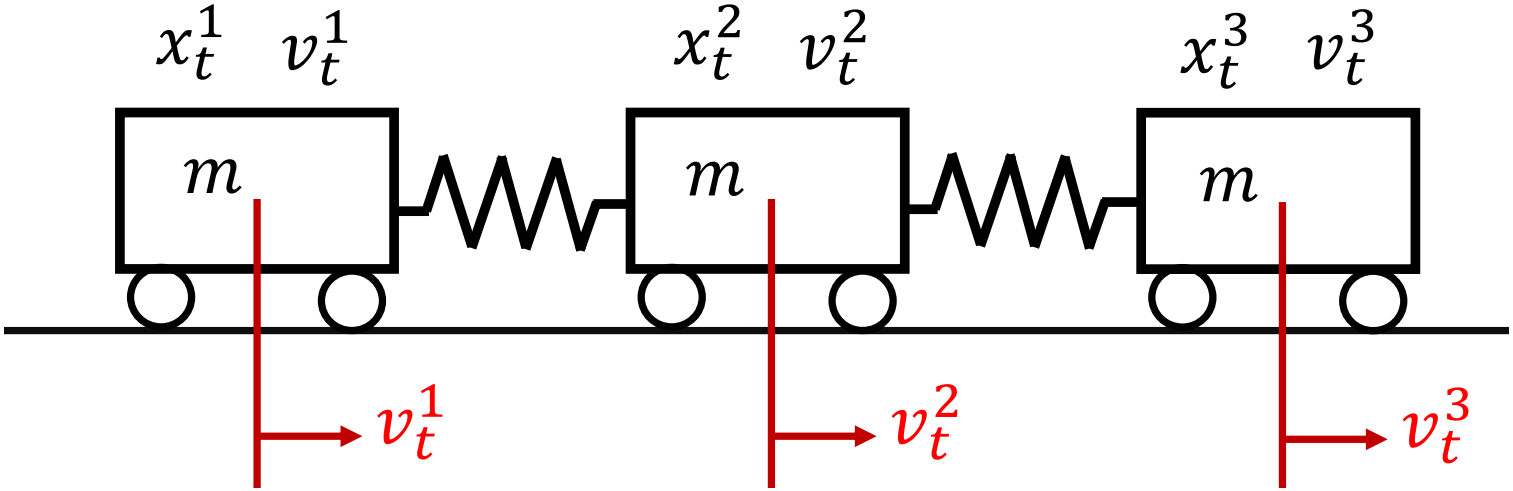
end

Remark

Finding a stabilizing $K_0 \in \text{Sparse}(S)$ may be difficult in general

$\Rightarrow$ cannot use pole-placement, which does not preserve sparsity.

Example (continued)



We have

$$S = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

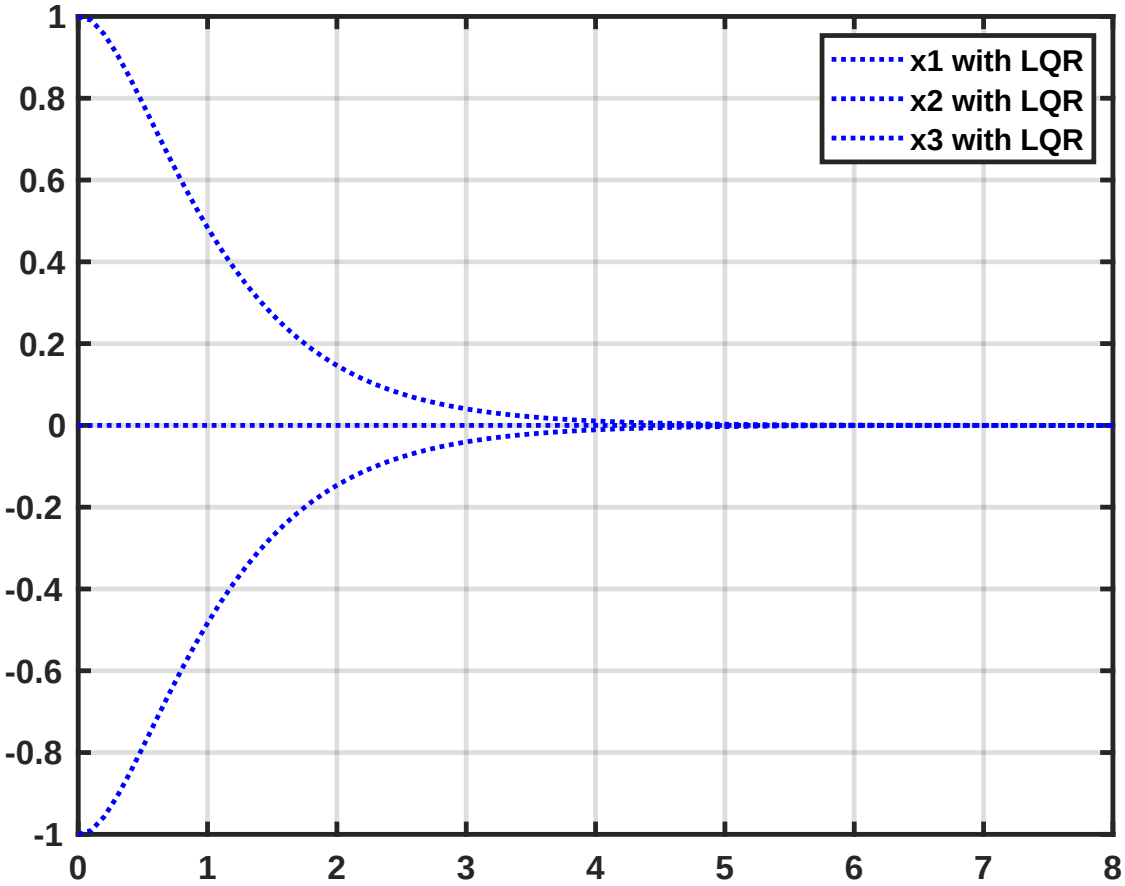
1. We compute the S-LQR controller

$$K^* = \begin{bmatrix} 1.9963 & 3.3700 & 0.4915 & 0.0482 & 0.0979 & 0.0253 \\ 0.4915 & 0.0482 & 1.6028 & 3.3471 & 0.4915 & 0.0482 \\ 0.0979 & 0.0253 & 0.4915 & 0.0482 & 1.9963 & 3.3700 \end{bmatrix}$$

Example (continued)

$$x_0 = \begin{bmatrix} -1 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Centralized S-LQR



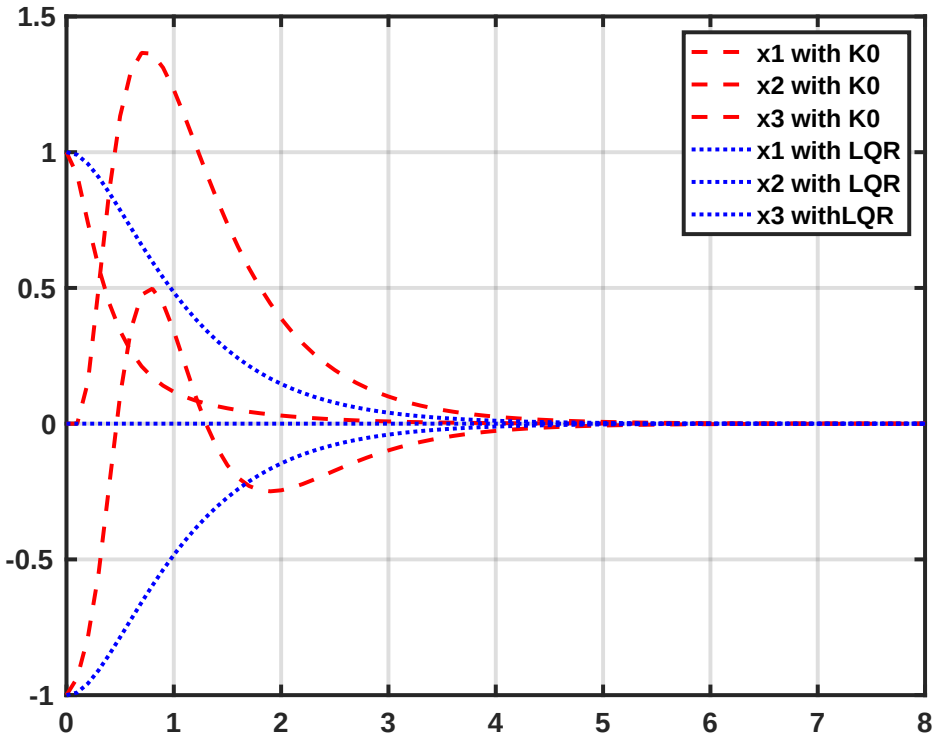
Ideal case!

- Cannot do better than this;
- Cannot be implemented...

Example (continued)

$$K_0 = \begin{bmatrix} 12.7604 & 6.7527 & 0 & -4.6106 & 0 & 0 \\ 0 & 0 & 7.1836 & 7.2582 & 0 & 17.2495 \\ 0 & 0 & 0 & 0 & 16.8597 & 8.8817 \end{bmatrix} \in \text{Sparse}(S)$$

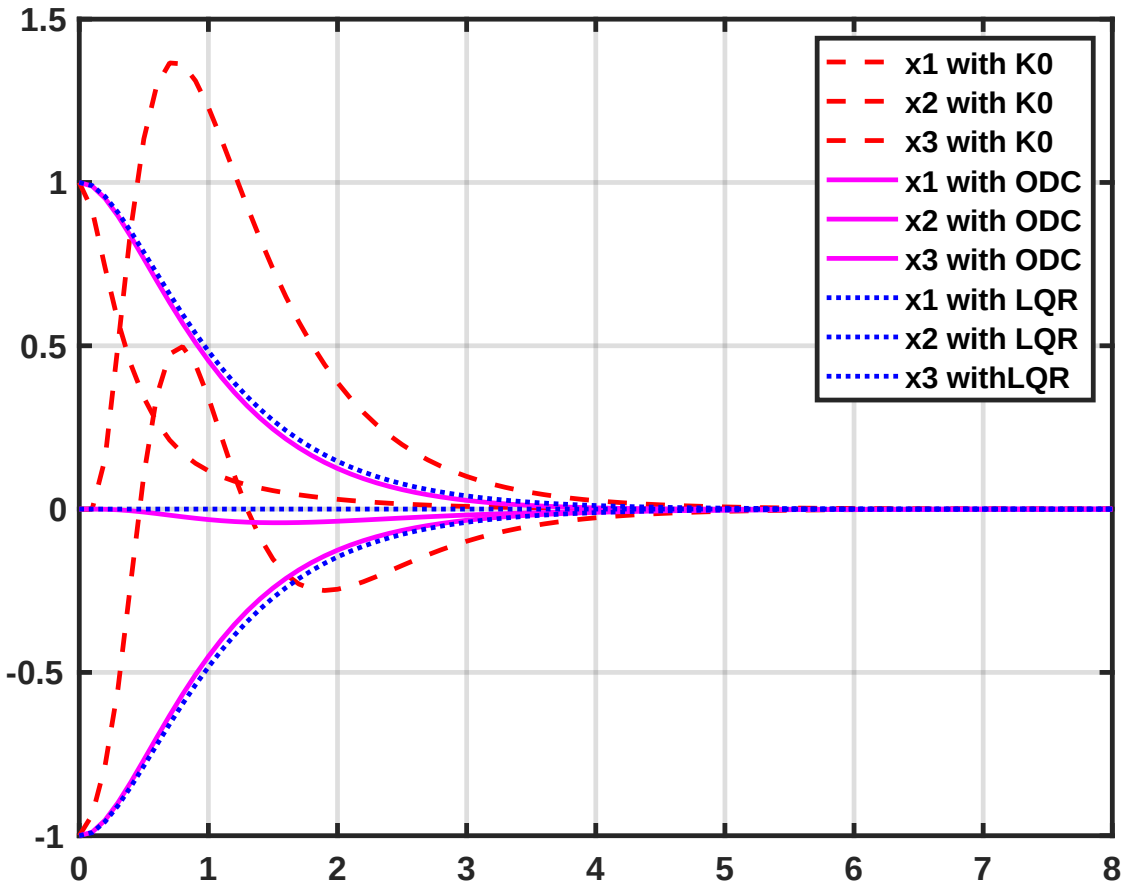
Stabilizing, but bad performance.



Example (continued)

We run PGD until convergence. We end up with

$$\hat{K} = \begin{bmatrix} 2.1555 & 3.3969 & 0 & -0.2928 & 0 & 0 \\ 0 & 0 & 1.9857 & 3.5402 & 0 & -0.3200 \\ 0 & 0 & 0 & 0 & 2.2346 & 3.5776 \end{bmatrix} \in \text{Sparse}(S)$$



- ODC : Optimal Distributed Control;
- Almost optimal performance;
- Can be implemented through available sensors.

Plan of the lecture

1. **Projected GD for locally optimal distributed controllers**
   - Binary matrices, sparsity subspaces and Distributed S-LQR
   - Projected Gradient Descent (PGD) for DS-LQR
   - Convergence results

Q2a) Does PGD converge?

- We know that, for small-enough $\eta > 0$

$$J\left(K - \eta \nabla J(K)\right) \leq J(K)$$

  $\Rightarrow \nabla J(K)$ is a progress direction.
- Is $\nabla J(K) \odot S$ a progress direction?

Progress directions

Let $f : \mathbb{R}^n \to \mathbb{R}$. The vector "$\mathbf{d}$" is a progress direction if

$$f(\mathbf{x} - \eta \mathbf{d}) \leq f(\mathbf{x})$$

Q2a) Does PGD converge?

Taylor Approximation

$$f(\mathbf{x} - \eta \mathbf{d}) \approx f(\mathbf{x}) - \eta \nabla f(\mathbf{x})^\top \mathbf{d}$$

$\Rightarrow$ for small-enough $\eta > 0$, "$\mathbf{d}$" is a progress direction iff

$$f(\mathbf{x}) - \eta \nabla f(\mathbf{x})^\top \mathbf{d} \leq f(\mathbf{x}) \quad \Rightarrow \quad \eta \nabla f(\mathbf{x})^\top \mathbf{d} \geq 0$$

- Equivalent for matrices: $D \in \mathbb{R}^{m \times n}$ is a progess direction iff

$$\text{Trace}\left(\nabla^\top J(K) D\right) \geq 0$$

- Clearly, $D = \nabla J(K)$ is a progress direction because

$$\text{Trace}\left(\nabla^\top J(K) \nabla J(K)\right) = \sum_{i=1}^{m} \sum_{j=1}^{n} \nabla^2 J(K)_{ij} \geq 0$$

Q2a) Does PGD converge?

Lemma

$\nabla J(K) \odot S$ is a progress direction for any $S \in \{0,1\}^{m \times n}$.

Proof: $\text{Trace}(\nabla J(K)^\top \nabla J(K) \odot S) = \sum_{(i,j)|S(i,j)=1} \nabla^2 J(K)_{ij} \geq 0$

Theorem

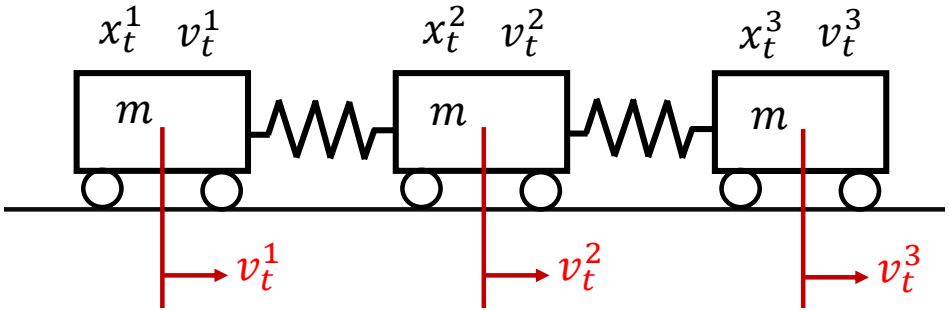
Let $\eta > 0$ be small enough for the standard GD algorithm to converge (see previous lecture). Then, PGD improves the cost at every iteration:

$$J(K_{j+1}) \leq J(K_j)$$

and it converges to a point $\hat{K} = \lim_{j \to \infty} K_j \in \text{Sparse}(S)$ such that

$$\nabla J(\hat{K}) \odot S = 0.$$

Example (continued 2...)



1. K^* (S-LQR, centralized)

$$K^* = \begin{bmatrix} 1.9963 & 3.3700 & 0.4915 & 0.0482 & 0.0979 & 0.0253 \\ 0.4915 & 0.0482 & 1.6028 & 3.3471 & 0.4915 & 0.0482 \\ 0.0979 & 0.0253 & 0.4915 & 0.0482 & 1.9963 & 3.3700 \end{bmatrix}, \quad J(K^*) = 60.3285$$

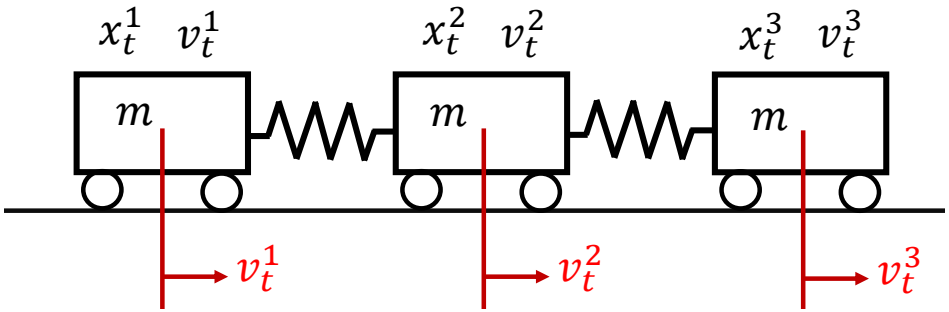
2. $K_0 \in \text{Sparse}(S)$ (stabilizing, sparse, suboptimal)

$$K_0 = \begin{bmatrix} 12.7604 & 6.7527 & 0 & -4.6106 & 0 & 0 \\ 0 & 0 & 7.1836 & 7.2582 & 0 & 17.2495 \\ 0 & 0 & 0 & 0 & 16.8597 & 8.8817 \end{bmatrix}, \quad J(K_0) = 500$$

3. $\hat{K} \in \text{Sparse}(S)$ (DS-LQR, sparse)

$$\hat{K} = \begin{bmatrix} 2.1555 & 3.3969 & 0 & -0.2928 & 0 & 0 \\ 0 & 0 & 1.9857 & 3.5402 & 0 & -0.3200 \\ 0 & 0 & 0 & 0 & 2.2346 & 3.5776 \end{bmatrix}, \quad J(\hat{K}) = 61.0764$$

Example (continued 2...)



1. $\nabla J(K^*) = \begin{bmatrix} -0.00 & -0.00 & 0.00 & -0.00 & 0.00 & -0.00 \\ 0.00 & -0.00 & 0.00 & -0.00 & 0.00 & -0.00 \\ -0.00 & 0.00 & 0.00 & -0.00 & -0.00 & 0.00 \end{bmatrix}$ Globally optimal

2. $\nabla J(K_0) = \begin{bmatrix} 4.80 & -20.24 & 1.40 & -46.41 & -1.24 & 8.86 \\ 1.77 & -10.50 & 3.17 & -43.19 & -9.34 & 41.50 \\ -2.34 & 7.02 & -6.06 & 35.82 & 34.91 & -37.86 \end{bmatrix}$ Suboptimal

3. $\nabla J(\hat{K}) = \begin{bmatrix} 0.00 & 0.00 & -0.54 & 0.00 & -0.39 & 0.20 \\ -0.96 & 0.58 & 0.00 & 0.00 & -0.55 & 0.00 \\ -0.46 & 0.17 & -0.90 & 0.58 & 0.00 & 0.00 \end{bmatrix}$ $\nabla J(K) \odot S = 0$

$S = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}$

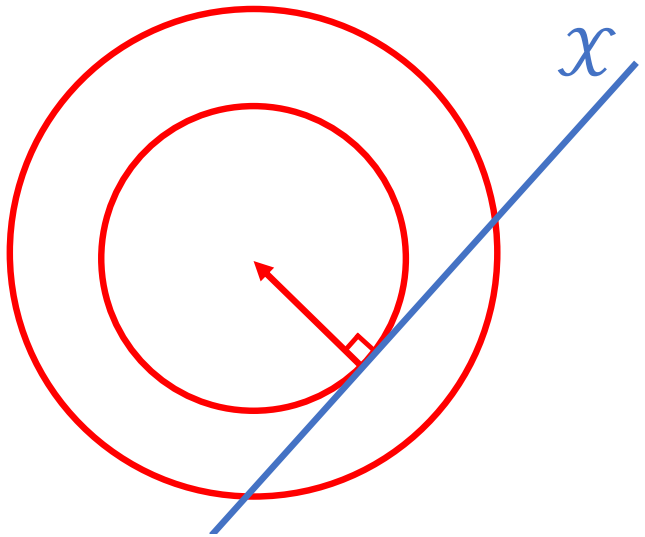
Example (continued 2...)

$$\nabla J(\hat{K}) = \begin{bmatrix} 0.00 & 0.00 & -0.54 & 0.00 & -0.39 & 0.20 \\ -0.96 & 0.58 & 0.00 & 0.00 & -0.55 & 0.00 \\ -0.46 & 0.17 & -0.90 & 0.58 & 0.00 & 0.00 \end{bmatrix}$$

$\nabla J(K) \odot S = 0$

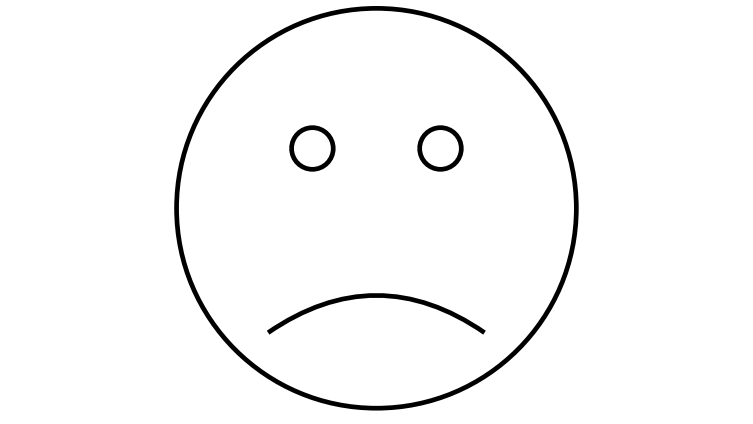
- The gradient has converged to Sparse($\bar{S}$), where $\bar{S}(i,j) = 1 \iff S(i,j) = 0$ ("opposite sparsity")

- MEANING: "We cannot further improve the cost by applying small-enough changes to the free entities of $K \in \text{Sparse}(S)$."



Q2b) Convergence to optimum

Theorem



Proof: By counter-example, there can be multiple matrices $\hat{K}_1, \ldots, \hat{K}_M$ such that $J(\hat{K}_1) \neq J(\hat{K}_2) \neq \ldots \neq J(\hat{K}_M)$ but

$$\nabla J(\hat{K}_1) \odot S = \nabla J(\hat{K}_2) \odot S = \cdots = \nabla J(\hat{K}_M) \odot S = 0.$$

Exercise 4 of today's session $\Rightarrow$ counter example!

Summary

- Binary matrices and sparsity constraints

$$K \in \text{Sparse}(S)$$

  can be used to model distributed control scenarios.
- Given initial stabilizing $K_0 \in \text{Sparse}(S)$, can use PGD

$$K_{j+1} = K_j - \eta \nabla J(K) \odot S$$

  to converge to locally optimal $\hat{K} \in \text{Sparse}(S)$!
- Centralized case: GD converges to S-LQR solution
- Distributed case: PGD might not converge to DS-LQR solution $\rightarrow$ only locally optimal