Performance Bounds of Model Predictive Control for Unconstrained and Constrained Linear Quadratic Problems and Beyond

Yuchao Li[†], *Aren Karapetyan*[‡], *John Lygeros*[‡], *Karl H. Johansson*[†], *Jonas Mårtensson*[†]

[†]Division of Decision and Control Systems, KTH Royal Institute of Technology, Stockholm, Sweden

[‡]Automatic Control Laboratory, Swiss Federal Institute of Technology in Zürich, Switzerland

Motivation

- Model predictive control (MPC) is a well-established scheme dealing effectively constraints on state and control.

Motivation

- Model predictive control (MPC) is a well-established scheme dealing effectively constraints on state and control.
- The function and constraint designed at the end of predicting horizon, referred to as **terminal ingredients**, play an important role in its performance and analysis.

Motivation

- Model predictive control (MPC) is a well-established scheme dealing effectively constraints on state and control.
- The function and constraint designed at the end of predicting horizon, referred to as **terminal ingredients**, play an important role in its performance and analysis.
- The framework developed in [Ber22] couches on the dynamic programming (DP) theory, and regards MPC as an approximate scheme for solving a functional equation.

Motivation

- Model predictive control (MPC) is a well-established scheme dealing effectively constraints on state and control.
- The function and constraint designed at the end of predicting horizon, referred to as **terminal ingredients**, play an important role in its performance and analysis.
- The framework developed in [Ber22] couches on the dynamic programming (DP) theory, and regards MPC as an approximate scheme for solving a functional equation.
- We leverage on the new framework, and investigate how the terminal ingredients affect the performance of MPC.

Difference in Perspective

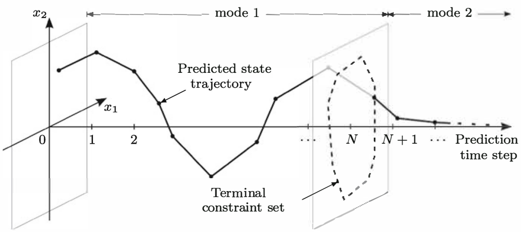
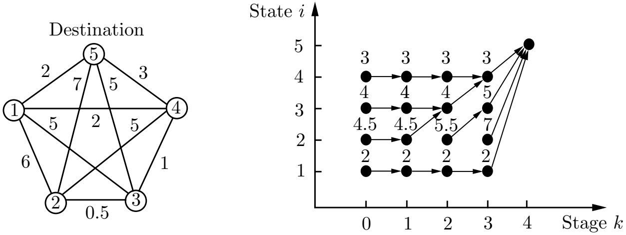


Figure source: [KC16, Fig. 2.1]

- MPC analyzes the predicted trajectory into the future: Forward in time, focusing on trajectories of states.

Difference in Perspective



(a) (b)

Figure source: [Ber17, Fig. 2.1.2]

- MPC analyzes the predicted trajectory into the future: Forward in time, focusing on trajectories of states.
- DP analysis proceeds as the algorithm progress: Forward in algorithmic iteration, focusing on functions of states.

Dynamic Programming Model

The scalar linear quadratic regulation (LQR) problem:

$$x_{k+1} = ax_k + bu_k, \quad \min_{u_k, k=0,1,\dots} \lim_{N\to\infty} \sum_{k=0}^{N-1} (qx_k^2 + ru_k^2).$$

Dynamic Programming Model

The scalar linear quadratic regulation (LQR) problem:

$$x_{k+1} = ax_k + bu_k, \quad \min_{u_k, k=0,1,\ldots} \lim_{N \to \infty} \sum_{k=0}^{N-1} (qx_k^2 + ru_k^2).$$

- The **state space** X is $\Re$, the **control constraint set** $U(x) \subset U$ is $\Re$.

Dynamic Programming Model

The scalar linear quadratic regulation (LQR) problem:

$$x_{k+1} = ax_k + bu_k, \quad \min_{u_k, k=0,1,\ldots} \lim_{N\to\infty} \sum_{k=0}^{N-1} (qx_k^2 + ru_k^2).$$

- The **state space** X is $\Re$, the **control constraint set** $U(x) \subset U$ is $\Re$.
- The **system dynamics** $f : X \times U \to X$ and the **stage cost** $g : X \times U \to \Re$

$$f(x,u) = ax + bu, \quad g(x,u) = qx^2 + ru^2.$$

Dynamic Programming Model

The scalar linear quadratic regulation (LQR) problem:

$$x_{k+1} = ax_k + bu_k, \qquad \min_{u_k, k=0,1,\dots} \lim_{N\to\infty} \sum_{k=0}^{N-1} (qx_k^2 + ru_k^2).$$

- The **state space** X is $\Re$, the **control constraint set** $U(x) \subset U$ is $\Re$.
- The **system dynamics** $f : X \times U \to X$ and the **stage cost** $g : X \times U \to \Re$

$$f(x,u) = ax + bu, \quad g(x,u) = qx^2 + ru^2.$$

- A **policy** $\mu : X \to U$ with $\mu(x) \in U(x)$ for all x and its **cost function**

$$\mu(x) = Lx, \quad J_\mu(x_0) = \lim_{N\to\infty} \sum_{k=0}^{N-1} \left(qx_k^2 + r(Lx_k)^2\right).$$

Dynamic Programming Model

The scalar linear quadratic regulation (LQR) problem:

$$x_{k+1} = ax_k + bu_k, \qquad \min_{u_k, k=0,1,\ldots} \lim_{N\to\infty} \sum_{k=0}^{N-1} (qx_k^2 + ru_k^2).$$

- The **state space** X is $\Re$, the **control constraint set** $U(x) \subset U$ is $\Re$.
- The **system dynamics** $f : X \times U \to X$ and the **stage cost** $g : X \times U \to \Re$

$$f(x, u) = ax + bu, \quad g(x, u) = qx^2 + ru^2.$$

- A **policy** $\mu : X \to U$ with $\mu(x) \in U(x)$ for all x and its **cost function**

$$\mu(x) = Lx, \quad J_\mu(x_0) = \lim_{N\to\infty} \sum_{k=0}^{N-1} \left(qx_k^2 + r(Lx_k)^2\right).$$

- The **optimal cost function** $J^* : X \to \Re$ and **optimal policy** $\mu^* : X \to U$:

$$J^*(x_0) = \min_{u_k, k=0,1,\ldots} \lim_{N\to\infty} \sum_{k=0}^{N-1} (qx_k^2 + ru_k^2), \quad \mu^*(x) = L^* x.$$

Bellman's Equation

- The optimal cost function J^* fulfills **Bellman's equation**:

$$J^*(x) = \min_{u \in U(x)} \Big[g(x, u) + J^*\big(f(x, u)\big) \Big], \quad \text{for all } x.$$

Bellman's Equation

- The optimal cost function J^* fulfills **Bellman's equation**:

$$J^*(x) = \min_{u \in U(x)} \Big[g(x,u) + J^*\big(f(x,u)\big) \Big], \quad \text{for all } x.$$

- For the LQR problem, we know one solution $J^*(x) = K^* x^2$, so that

$$K^* x^2 = \min_{u \in \Re} \left[q x^2 + r u^2 + K^* (ax + bu)^2 \right], \quad \text{for all } x.$$

Bellman's Equation

- The optimal cost function J^* fulfills **Bellman's equation**:

$$J^*(x) = \min_{u \in U(x)} \left[g(x,u) + J^*\left(f(x,u)\right) \right], \quad \text{for all } x.$$

- For the LQR problem, we know one solution $J^*(x) = K^* x^2$, so that

$$K^* x^2 = \min_{u \in \Re} \left[q x^2 + r u^2 + K^* (ax + bu)^2 \right], \quad \text{for all } x.$$

- The optimization problem is transformed as solving **fixed point equations**:

$$\min_{u_k, k=0,1,\dots} \lim_{N \to \infty} \sum_{k=0}^{N-1} (q x_k^2 + r u_k^2) \implies J^*(x) = \min_{u \in \Re} \left[q x^2 + r u^2 + J^*(ax + bu) \right].$$

Bellman's Equation

- The optimal cost function J^* fulfills **Bellman's equation**:

$$J^*(x) = \min_{u \in U(x)} \Big[g(x,u) + J^*\big(f(x,u)\big) \Big], \quad \text{for all } x.$$

- For the LQR problem, we know one solution $J^*(x) = K^* x^2$, so that

$$K^* x^2 = \min_{u \in \Re} \left[qx^2 + ru^2 + K^*(ax+bu)^2 \right], \quad \text{for all } x.$$

- The optimization problem is transformed as solving **fixed point equations**:

$$\min_{u_k, k=0,1,\ldots} \lim_{N \to \infty} \sum_{k=0}^{N-1} (qx_k^2 + ru_k^2) \implies J^*(x) = \min_{u \in \Re} \left[qx^2 + ru^2 + J^*(ax+bu) \right].$$

- Bellman's equation holds for a given policy μ:

$$J_\mu(x) = g\big(x, \mu(x)\big) + J_\mu\Big(f\big(x, \mu(x)\big)\Big), \quad \text{for all } x.$$

Value Iteration

- The VI algorithm generates a sequence of functions $\{J_k\}$ by

$$J_{k+1}(x) = \min_{u \in U(x)} \Big[g(x,u) + J_k\big(f(x,u)\big) \Big], \quad k = 0, 1, \ldots .$$

Value Iteration

- The VI algorithm generates a sequence of functions $\{J_k\}$ by

$$J_{k+1}(x) = \min_{u \in U(x)} \Big[g(x,u) + J_k\big(f(x,u)\big) \Big], \quad k = 0, 1, \ldots.$$

- For the LQR problem with $J_0(x) = K_0 x^2$, this is simplified as

$$K_{k+1} = F(K_k) \text{ where } F(K) = \frac{a^2 r K}{r + b^2 K} + q.$$

Value Iteration

- The VI algorithm generates a sequence of functions $\{J_k\}$ by

$$J_{k+1}(x) = \min_{u \in U(x)} \Big[g(x,u) + J_k\big(f(x,u)\big)\Big], \quad k = 0, 1, \ldots.$$

- For the LQR problem with $J_0(x) = K_0 x^2$, this is simplified as

$$K_{k+1} = F(K_k) \text{ where } F(K) = \frac{a^2 rK}{r + b^2 K} + q.$$

- **Principle of optimality** yields

$$\begin{aligned}\min_{u_0, u_1 \in \Re} \left[qx_0^2 + ru_0^2 + qx_1^2 + ru_1^2 + Kx_2^2\right] &= \min_{u_0 \in \Re} \left[qx_0^2 + ru_0^2 + \min_{u_1 \in \Re} \left[qx_1^2 + ru_1^2 + Kx_2^2\right]\right] \\ &= \min_{u_0 \in \Re} \left[qx_0^2 + ru_0^2 + F(K)x_1^2\right]\end{aligned}$$

Policy Iteration

- The PI algorithm generates a sequence of functions $\{J_{\mu^k}\}$ by **policy evaluation**

$$J_{\mu^k}(x) = g\big(x, \mu^k(x)\big) + J_{\mu^k}\Big(f\big(x, \mu^k(x)\big)\Big).$$

Policy Iteration

- The PI algorithm generates a sequence of functions $\{J_{\mu^k}\}$ by **policy evaluation**

$$J_{\mu^k}(x) = g\left(x, \mu^k(x)\right) + J_{\mu^k}\Big(f\left(x, \mu^k(x)\right)\Big).$$

- This is followed by **policy improvement**

$$\mu^{k+1}(x) \in \arg \min_{u \in U(x)} \Big[g\left(x, \mu^k(x)\right) + J_{\mu^k}\Big(f\left(x, \mu^k(x)\right)\Big)\Big].$$

Policy Iteration

- The PI algorithm generates a sequence of functions $\{J_{\mu^k}\}$ by **policy evaluation**

$$J_{\mu^k}(x) = g\left(x, \mu^k(x)\right) + J_{\mu^k}\Big(f\left(x, \mu^k(x)\right)\Big).$$

- This is followed by **policy improvement**

$$\mu^{k+1}(x) \in \arg\min_{u \in U(x)} \left[g\left(x, \mu^k(x)\right) + J_{\mu^k}\Big(f\left(x, \mu^k(x)\right)\Big)\right].$$

- For the LQR problem with $\mu^0 = L_0 x$, policy evaluation is solving the Lyapunov equation

$$K_k x^2 = q x^2 + r(L_k x)^2 + K_k(ax + bL_k x)^2 \iff K_k = q + rL_k^2 + K_k(a + bL_k)^2$$

Policy Iteration

- The PI algorithm generates a sequence of functions $\{J_{\mu^k}\}$ by **policy evaluation**

$$J_{\mu^k}(x) = g\left(x, \mu^k(x)\right) + J_{\mu^k}\Big(f\left(x, \mu^k(x)\right)\Big).$$

- This is followed by **policy improvement**

$$\mu^{k+1}(x) \in \arg\min_{u \in U(x)} \Big[g\left(x, \mu^k(x)\right) + J_{\mu^k}\Big(f\left(x, \mu^k(x)\right)\Big)\Big].$$

- For the LQR problem with $\mu^0 = L_0 x$, policy evaluation is solving the Lyapunov equation

$$K_k x^2 = qx^2 + r(L_k x)^2 + K_k(ax + bL_k x)^2 \iff K_k = q + rL_k^2 + K_k(a + bL_k)^2$$

- The policy improvement fulfills

$$(a + bL_{k+1})^2 = \frac{\partial F}{\partial K}(K_k)$$

A Dynamic Programming View: Unconstrained Problem

- Consider MPC scheme for solving LQR:

$$\begin{aligned} \min_{\{u_k\}_{k=0}^{\ell-1}} \quad & K x_\ell^2 + \sum_{k=0}^{\ell-1} q x_k^2 + r u_k^2 \\ \mathrm{s.t.} \quad & x_0 = x,\ x_{k+1} = a x_k + b u_k,\ k = 0, \ldots, \ell - 1. \end{aligned}$$

A Dynamic Programming View: Unconstrained Problem

- Consider MPC scheme for solving LQR:

$$\min_{\{u_k\}_{k=0}^{\ell-1}} \quad Kx_\ell^2 + \sum_{k=0}^{\ell-1} qx_k^2 + ru_k^2$$
$$\mathrm{s.t.} \quad x_0 = x,\ x_{k+1} = ax_k + bu_k,\ k = 0, \dots, \ell - 1.$$

- The policy $\tilde{\mu}$ is obtained via $\ell - 2$ steps of VIs followed by one step of PI.

A Dynamic Programming View: Unconstrained Problem

- Consider MPC scheme for solving LQR:

$$\min_{\{u_k\}_{k=0}^{\ell-1}} \quad Kx_\ell^2 + \sum_{k=0}^{\ell-1} qx_k^2 + ru_k^2$$
$$\mathrm{s.t.} \quad x_0 = x,\ x_{k+1} = ax_k + bu_k,\ k = 0, ..., \ell - 1.$$

- The policy $\tilde{\mu}$ is obtained via $\ell - 2$ steps of VIs followed by one step of PI.
- In particular, minimizing over $\{u_k\}_{k=1}^{\ell-1}$ amounts to computing

$$K_{i+1} = F(K_i), \quad i = 0, \ldots, \ell - 2,$$

with $K_0 = K$ as the initial guess of the solution to Bellman's equation.

A Dynamic Programming View: Unconstrained Problem

- Consider MPC scheme for solving LQR:

$$\min_{\{u_k\}_{k=0}^{\ell-1}} \quad Kx_\ell^2 + \sum_{k=0}^{\ell-1} qx_k^2 + ru_k^2$$
$$\text{s.t.} \quad x_0 = x,\ x_{k+1} = ax_k + bu_k,\ k = 0, ..., \ell - 1.$$

- The policy $\tilde{\mu}$ is obtained via $\ell - 2$ steps of VIs followed by one step of PI.
- In particular, minimizing over $\{u_k\}_{k=1}^{\ell-1}$ amounts to computing

$$K_{i+1} = F(K_i), \quad i = 0, \dots, \ell - 2,$$

with $K_0 = K$ as the initial guess of the solution to Bellman's equation.

- The policy $\tilde{\mu}$ is set as $\tilde{\mu}(x) = \tilde{L}x$ where

$$(a + b\tilde{L})^2 = \frac{\partial F}{\partial K}(K_{\ell-1}).$$

A Dynamic Programming View: Unconstrained Problem

- Consider MPC scheme for solving LQR:

$$\min_{\{u_k\}_{k=0}^{\ell-1}} \quad Kx_\ell^2 + \sum_{k=0}^{\ell-1} qx_k^2 + ru_k^2$$
$$\text{s.t.} \quad x_0 = x,\ x_{k+1} = ax_k + bu_k,\ k = 0, ..., \ell - 1.$$

- The policy $\tilde{\mu}$ is obtained via $\ell - 2$ steps of VIs followed by one step of PI.
- In particular, minimizing over $\{u_k\}_{k=1}^{\ell-1}$ amounts to computing

$$K_{i+1} = F(K_i), \quad i = 0, \dots, \ell - 2,$$

with $K_0 = K$ as the initial guess of the solution to Bellman's equation.

- The policy $\tilde{\mu}$ is set as $\tilde{\mu}(x) = \tilde{L}x$ where

$$(a + b\tilde{L})^2 = \frac{\partial F}{\partial K}(K_{\ell-1}).$$

- The performance of $\tilde{\mu}$ is judged by $K_{\tilde{L}}$, which is obtained by policy evaluation [Ber22]

$$K_{\tilde{L}} = q + r\tilde{L}^2 + K_{\tilde{L}}(a + b\tilde{L})^2.$$

A Dynamic Programming View: Unconstrained Problem

- Consider MPC scheme for solving LQR:

$$\min_{\{u_k\}_{k=0}^{\ell-1}} \quad Kx_\ell^2 + \sum_{k=0}^{\ell-1} qx_k^2 + ru_k^2$$
$$\text{s.t.} \quad x_0 = x,\ x_{k+1} = ax_k + bu_k,\ k = 0, ..., \ell - 1.$$

- The policy $\tilde{\mu}$ is obtained via $\ell - 2$ steps of VIs followed by one step of PI.
- In particular, minimizing over $\{u_k\}_{k=1}^{\ell-1}$ amounts to computing

$$K_{i+1} = F(K_i), \quad i = 0, \dots, \ell - 2,$$

with $K_0 = K$ as the initial guess of the solution to Bellman's equation.
- The policy $\tilde{\mu}$ is set as $\tilde{\mu}(x) = \tilde{L}x$ where

$$(a + b\tilde{L})^2 = \frac{\partial F}{\partial K}(K_{\ell-1}).$$

- The performance of $\tilde{\mu}$ is judged by $K_{\tilde{L}}$, which is obtained by policy evaluation [Ber22]

$$K_{\tilde{L}} = q + r\tilde{L}^2 + K_{\tilde{L}}(a + b\tilde{L})^2.$$

- The cost $K_{\tilde{L}}$ is the approximate solution to Bellman's equation computed via MPC.

A Dynamic Programming View: Constrained Problem

- Consider a constrained problem involving state constraint $\hat{X}$ and control constraint U.

A Dynamic Programming View: Constrained Problem

- Consider a constrained problem involving state constraint $\hat{X}$ and control constraint U.
- When applying MPC for this problem, we define a policy $\tilde{\mu}$ pointwise by solving the following optimization:

$$
\begin{aligned}
\min_{\{u_k\}_{k=0}^{\ell-1}} \quad & K x_\ell^2 + \sum_{k=0}^{\ell-1} q x_k^2 + r u_k^2 \\
\text{s. t.} \quad & x_{k+1} = a x_k + b u_k,\ k = 0, \ldots, \ell - 1, \\
& x_k \in \hat{X},\ k = 0, \ldots, \ell - 1, \\
& u_k \in U,\ k = 0, \ldots, \ell - 1, \\
& x_\ell \in S \subset \hat{X}, \quad x_0 = x,
\end{aligned}
$$

with suitably designed K and S.

A Dynamic Programming View: Constrained Problem

- Consider a constrained problem involving state constraint $\hat{X}$ and control constraint U.
- When applying MPC for this problem, we define a policy $\tilde{\mu}$ pointwise by solving the following optimization:

$$\begin{aligned} \min_{\{u_k\}_{k=0}^{\ell-1}} \quad & Kx_\ell^2 + \sum_{k=0}^{\ell-1} qx_k^2 + ru_k^2 \\ \text{s.t.} \quad & x_{k+1} = ax_k + bu_k,\ k = 0, \dots, \ell - 1, \\ & x_k \in \hat{X},\ k = 0, \dots, \ell - 1, \\ & u_k \in U,\ k = 0, \dots, \ell - 1, \\ & x_\ell \in S \subset \hat{X}, \quad x_0 = x, \end{aligned}$$

  with suitably designed K and S.
- Identical interpretation: $\ell - 1$ steps VIs followed by one step policy improvement, with K and S collectively as initial guess.

Main Result

- Proposition 13 (informal): Let $\{x_i^*\}_{i=0}^{\ell}$ be the sequence of states under μ^* starting from x. Then we have

$$J_{\tilde{\mu}}(x) - J^*(x) \leq K(x_\ell^*)^2 - J^*(x_\ell^*) \leq (K - K^*)(x_\ell^*)^2.$$

Main Result

- Proposition 13 (informal): Let $\{x_i^*\}_{i=0}^{\ell}$ be the sequence of states under μ^* starting from x. Then we have

$$J_{\tilde{\mu}}(x) - J^*(x) \leq K(x_\ell^*)^2 - J^*(x_\ell^*) \leq (K - K^*)(x_\ell^*)^2.$$

- The last inequality follows from the fact that $J^*(x)$ is the optimal cost for the constrained problem, while K^*x^2 is the optimal cost for the unconstrained problem. As a result, we have $J^*(x) \geq K^*x^2$ for all x.

Main Result

- Proposition 13 (informal): Let $\{x_i^*\}_{i=0}^{\ell}$ be the sequence of states under μ^* starting from x. Then we have

$$J_{\tilde{\mu}}(x) - J^*(x) \leq K(x_\ell^*)^2 - J^*(x_\ell^*) \leq (K - K^*)(x_\ell^*)^2.$$

- The last inequality follows from the fact that $J^*(x)$ is the optimal cost for the constrained problem, while K^*x^2 is the optimal cost for the unconstrained problem. As a result, we have $J^*(x) \geq K^*x^2$ for all x.
- Note that for this problem, μ^* is *not* linear, and J^* is *not* quadratic.

Main Result

- Proposition 13 (informal): Let $\{x_i^*\}_{i=0}^{\ell}$ be the sequence of states under μ^* starting from x. Then we have

$$J_{\tilde{\mu}}(x) - J^*(x) \leq K(x_\ell^*)^2 - J^*(x_\ell^*) \leq (K - K^*)(x_\ell^*)^2.$$

- The last inequality follows from the fact that $J^*(x)$ is the optimal cost for the constrained problem, while K^*x^2 is the optimal cost for the unconstrained problem. As a result, we have $J^*(x) \geq K^*x^2$ for all x.
- Note that for this problem, μ^* is *not* linear, and J^* is *not* quadratic.
- Given that μ^* drives the system to zero exponentially, even large $K - K^*$ would results in a small bound value, as x_ℓ^* is likely to be small.

Conventional Wisdom

- Focusing here on the MPC with both terminal constraints and terminal costs.

Conventional Wisdom

- Focusing here on the MPC with both terminal constraints and terminal costs.
- The conventional form of MPC is

$$\begin{aligned} \min_{\{u_k\}_{k=0}^{\ell-1}} \quad & K^* x_\ell^2 + \sum_{k=0}^{\ell-1} q x_k^2 + r u_k^2 \\ \text{s.t.} \quad & x_{k+1} = a x_k + b u_k,\ k = 0, \dots, \ell - 1, \\ & x_k \in \hat{X},\ u_k \in U,\ \ k = 0, \dots, \ell - 1, \\ & x_\ell \in S^*,\ x_0 = x, \end{aligned}$$

where K^* solves the Riccati equation, and $S^* \subset \hat{X}$ has the property that

$$x \in S^* \implies L^* x \in U \text{ and } (a + bL^*) x \in S^*,$$

where L^* is optimal LQR control.

Conventional Wisdom

- Focusing here on the MPC with both terminal constraints and terminal costs.
- The conventional form of MPC is

$$\begin{aligned} \min_{\{u_k\}_{k=0}^{\ell-1}} \quad & K^* x_\ell^2 + \sum_{k=0}^{\ell-1} q x_k^2 + r u_k^2 \\ \text{s.t.} \quad & x_{k+1} = a x_k + b u_k,\ k = 0, \ldots, \ell-1, \\ & x_k \in \hat{X},\ u_k \in U,\ \ k = 0, \ldots, \ell-1, \\ & x_\ell \in S^*,\ x_0 = x, \end{aligned}$$

  where K^* solves the Riccati equation, and $S^* \subset \hat{X}$ has the property that

$$x \in S^* \implies L^* x \in U \text{ and } (a + bL^*)x \in S^*,$$

  where L^* is optimal LQR control.
- The reason for choosing $K^* x^2$ as terminal cost is to have good performance $J_{\tilde{\mu}}$, **which seems not necessary, according to our bound!**

New Design Based on Our Insight

- To get a good closed loop performance judged by the cost $J_{\tilde{\mu}}$, it is not crucial for the terminal cost to be K^*, which may produce a small S^*, if L^* is large.

New Design Based on Our Insight

- To get a good closed loop performance judged by the cost $J_{\tilde{\mu}}$, it is not crucial for the terminal cost to be K^*, which may produce a small S^*, if L^* is large.
- Instead, we may try to use some K that is related to 'small' control Lx. As a result, the corresponding terminal set S can be larger than S^*.

New Design Based on Our Insight

- To get a good closed loop performance judged by the cost $J_{\tilde{\mu}}$, it is not crucial for the terminal cost to be K^*, which may produce a small S^*, if L^* is large.
- Instead, we may try to use some K that is related to 'small' control Lx. As a result, the corresponding terminal set S can be larger than S^*.
- Then new design is

$$\begin{aligned}
\min_{\{u_k\}_{k=0}^{\ell-1}} \quad & Kx_\ell^2 + \sum_{k=0}^{\ell-1} qx_k^2 + ru_k^2 \\
\text{s. t.} \quad & x_{k+1} = ax_k + bu_k,\ k = 0, \dots, \ell - 1, \\
& x_k \in \hat{X},\ u_k \in U,\ \ k = 0, \dots, \ell - 1, \\
& x_\ell \in S,\ x_0 = x,
\end{aligned}$$

where a choice for K could be the solution K_L that solves $K_L = q + rL^2 + K_L(a + bL)^2$.

New Design Based on Our Insight

- To get a good closed loop performance judged by the cost $J_{\tilde{\mu}}$, it is not crucial for the terminal cost to be K^*, which may produce a small S^*, if L^* is large.
- Instead, we may try to use some K that is related to 'small' control Lx. As a result, the corresponding terminal set S can be larger than S^*.
- Then new design is

$$\begin{aligned} \min_{\{u_k\}_{k=0}^{\ell-1}} \quad & Kx_\ell^2 + \sum_{k=0}^{\ell-1} qx_k^2 + ru_k^2 \\ \text{s.t.} \quad & x_{k+1} = ax_k + bu_k,\ k = 0, ..., \ell - 1, \\ & x_k \in \hat{X},\ u_k \in U,\ k = 0, ..., \ell - 1, \\ & x_\ell \in S,\ x_0 = x, \end{aligned}$$

  where a choice for K could be the solution K_L that solves $K_L = q + rL^2 + K_L(a + bL)^2$.
- Since S is larger, the feasible set of initial states x for the modified MPC is also larger, while the performance should be rather comparable.

Two Dimensional Example

- Consider a double integrator with both state and control constraints.

Two Dimensional Example

- Consider a double integrator with both state and control constraints.
- The matrix K is designed as the solution of Riccati equation

$$K = A'\big(K - KB(B'KB + \hat{R})^{-1}B'K\big)A + Q,$$

where $\hat{R} = 50R$.

Two Dimensional Example

- Consider a double integrator with both state and control constraints.
- The matrix K is designed as the solution of Riccati equation

$$K = A'\big(K - KB(B'KB + \hat{R})^{-1}B'K\big)A + Q,$$

where $\hat{R} = 50R$.

- As a result, $\|K\| > 6\|K^*\|$.

Two Dimensional Example

- Consider a double integrator with both state and control constraints.
- The matrix K is designed as the solution of Riccati equation

$$K = A'\big(K - KB(B'KB + \hat{R})^{-1}B'K\big)A + Q,$$

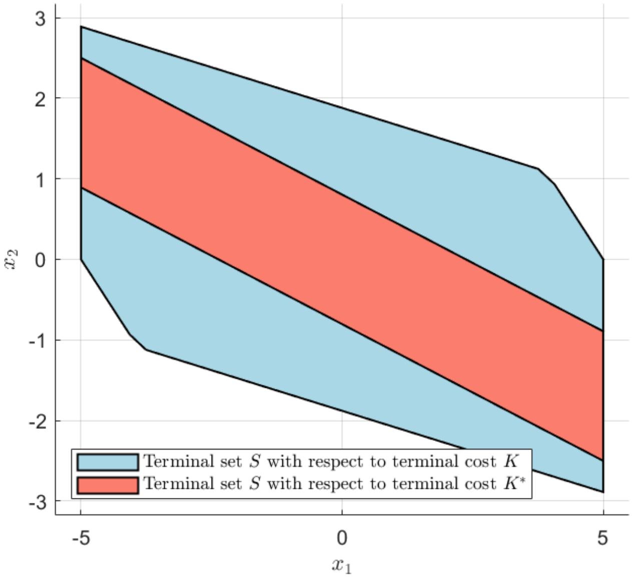
  where $\hat{R} = 50R$.
- As a result, $\|K\| > 6\|K^*\|$.
- The set $S \subset \hat{X}$ has the property that

$$x \in S \implies Lx \in U \text{ and } (a + bL)x \in S,$$

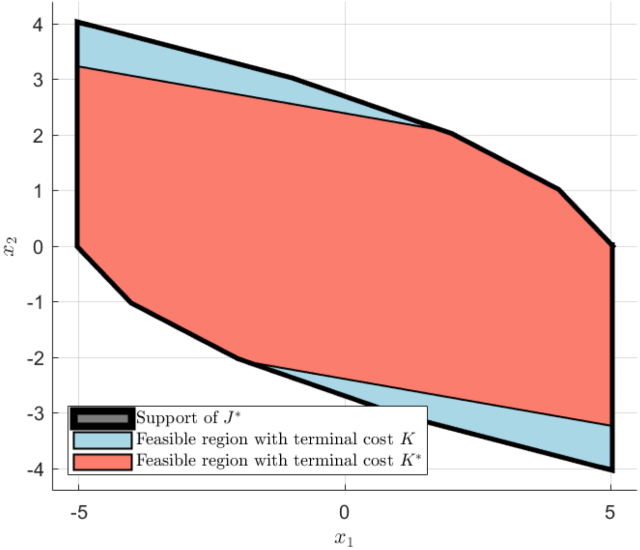
  where L is computed as

$$L = -(B'KB + \hat{R})^{-1}B'KA.$$

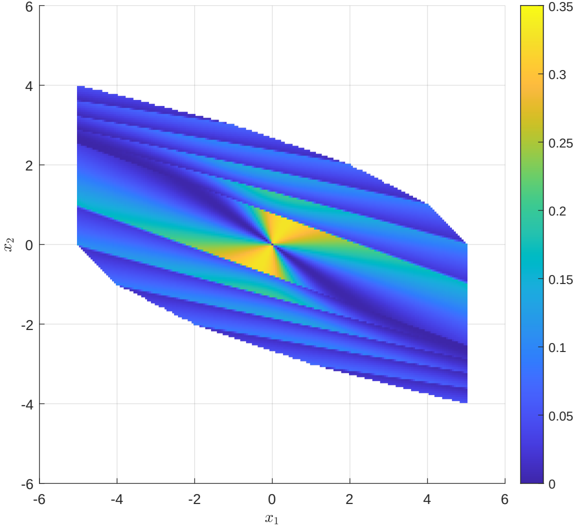
Two Dimensional Example



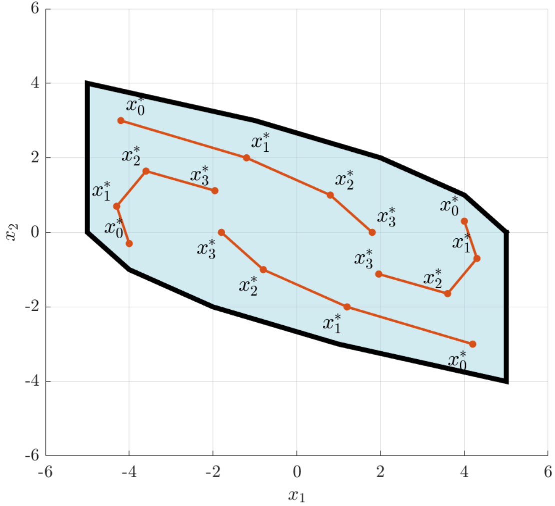
Two Dimensional Example



Two Dimensional Example



Two Dimensional Example



Conclusions

- We analyzed the performance of MPC via the perspective of DP for unconstrained and constrained LQR problems;

Conclusions

- We analyzed the performance of MPC via the perspective of DP for unconstrained and constrained LQR problems;
- The insights obtained led to new designs of terminal ingredients with larger feasible regions while costing little in performance.

References I

[Ber17] Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control*, volume 1. Athena Scientific, 4 edition, 2017.

[Ber22] Dimitri P. Bertsekas. *Lessons from AlphaZero for Optimal, Model Predictive, and Adaptive Control*. Athena Scientific, 2022.

[KC16] Basil Kouvaritakis and Mark Cannon. *Model Predictive Control*. Springer, 2016.