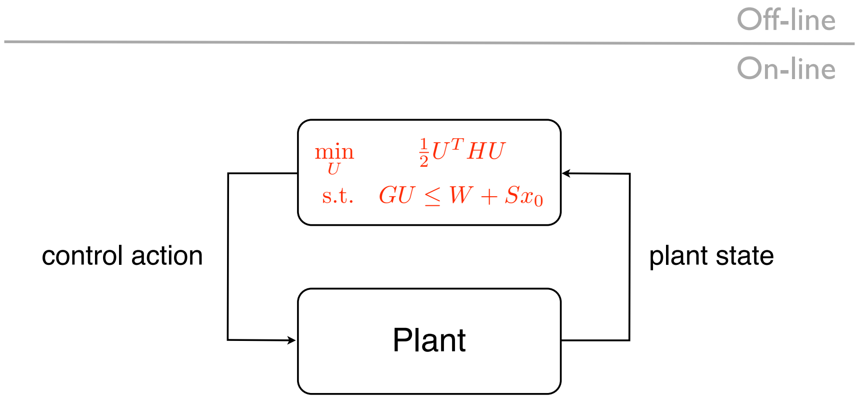
Low-Complexity Polynomial Approximation of Explicit MPC via Linear Programming

M. Kvasnica[1], J. Löfberg[2], M. Herceg[1], Ľ. Čirka[1], M. Fikar[1]

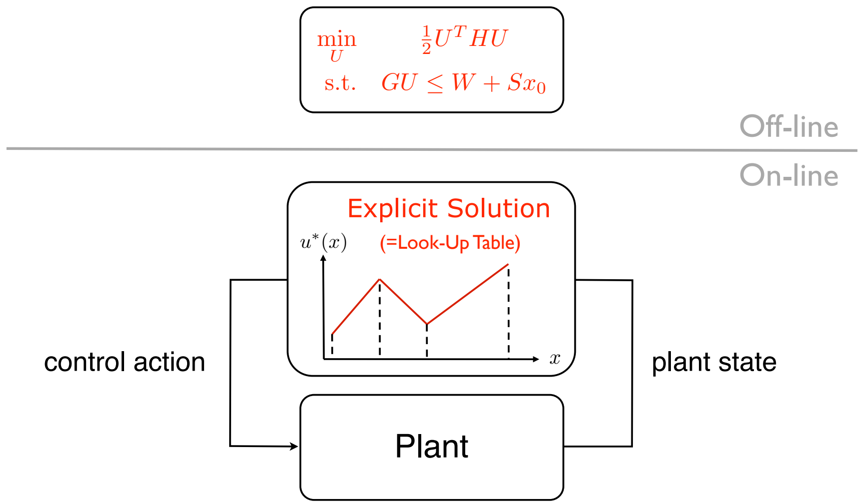
[1]Slovak University of Technology in Bratislava
[2]Linköping University

How to implement MPC using limited resources?
(100 bytes, 100 FLOPS)

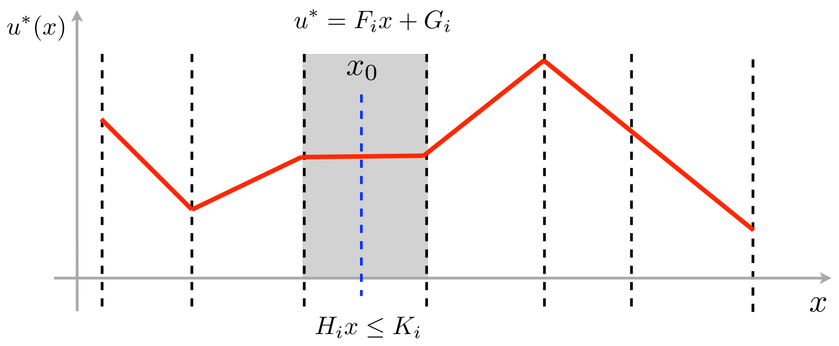
Explicit Model Predictive Control



Explicit Model Predictive Control



Explicit Model Predictive Control

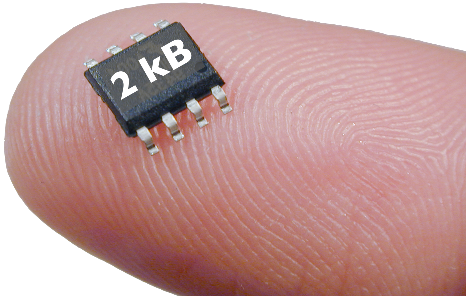


- Trading implementation speed for memory
- Memory storage is proportional to the number of regions

Main Issue: Memory Consumption

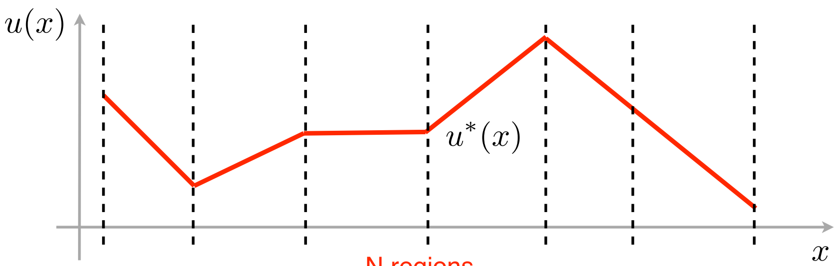
100 bytes x 100 regions = 10 kB of RAM

Need to reduce the number of regions as much as possible.
Ideally, remove all of them.



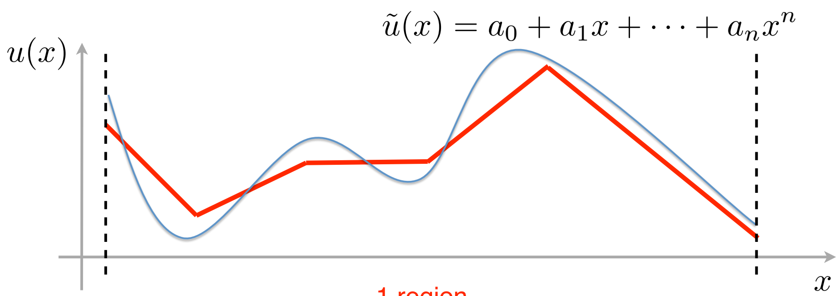
The Idea

- Find an approximate feedback which
  - is defined over a single region (hence saves memory)
  - guarantees closed-loop stability & constraint satisfaction
  - trades off performance for implementation cost



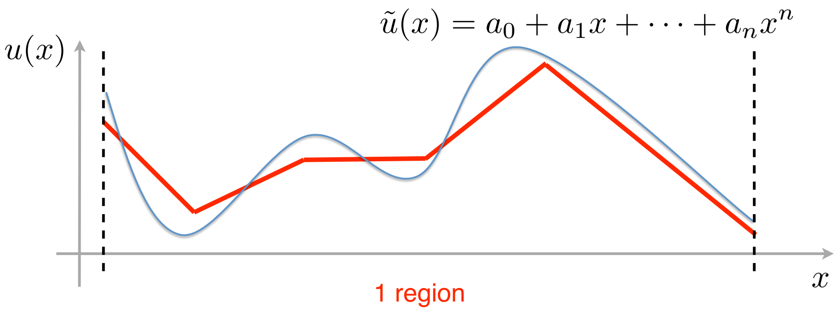
The Idea

- Find an approximate feedback which
  - is defined over a single region (hence saves memory)
  - guarantees closed-loop stability & constraint satisfaction
  - trades off performance for implementation cost
- Polynomial is an ideal candidate (low storage, fast evaluation)



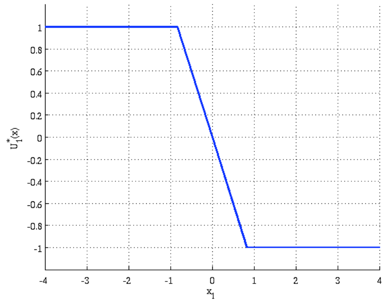
The Idea

- Find an approximate feedback which
  - is defined over a single region (hence saves memory)
  - **guarantees closed-loop stability & constraint satisfaction**
  - trades off performance for implementation cost
- Polynomial is an ideal candidate (low storage, fast evaluation)



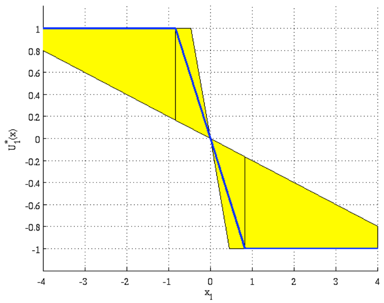
The Idea Continued...

- Given is:
  - LTI or PWA discrete-time system
  - explicit MPC feedback which guarantees closed-loop stability
  - PWA Lyapunov function
- Is it the only feedback which gives stability?



The Idea Continued...

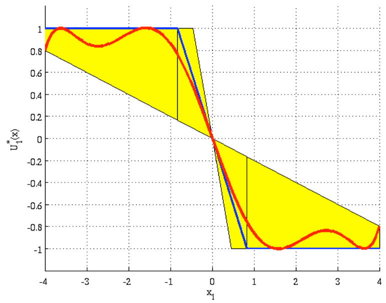
- Given is:
  - LTI or PWA discrete-time system
  - explicit MPC feedback which guarantees closed-loop stability
  - PWA Lyapunov function
- Is it the only feedback which gives stability?
- Theorem:
  - a set of stabilizing feedbacks exists
  - it is represented by polytopes
  - it can be computed



Christophersen; Springer 2007

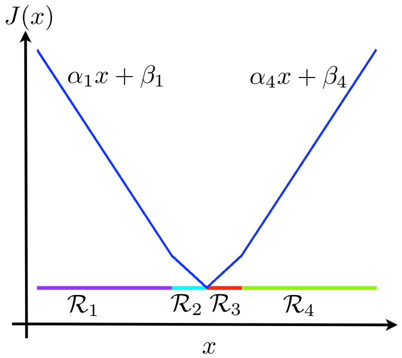
The Idea Continued...

- Given is:
  - LTI or PWA discrete-time system
  - explicit MPC feedback which guarantees closed-loop stability
  - PWA Lyapunov function
- Is it the only feedback which gives stability?
- Theorem:
  - a set of stabilizing feedbacks exists
  - it is represented by polytopes
  - it can be computed
- Corollary:
  - if the polynomial resides in the set, stability is guaranteed



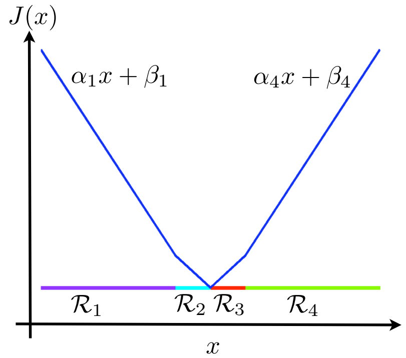
Set of Stabilizing Controllers

- Given:
  - linear discrete-time system $x^+ = Ax + Bu$
  - PWA Lyapunov function $J(x) = \alpha_i x + \beta_i$ if $x \in \mathcal{R}_i$



Set of Stabilizing Controllers

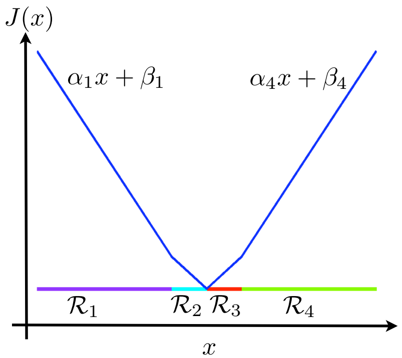
- Given:
  - linear discrete-time system $x^+ = Ax + Bu$
  - PWA Lyapunov function $J(x) = \alpha_i x + \beta_i$ if $x \in \mathcal{R}_i$
- Sketch:
  - any control action which guarantees decrease of the Lyapunov function is stabilizing



Set of Stabilizing Controllers

- Given:
  - linear discrete-time system $x^+ = Ax + Bu$
  - PWA Lyapunov function $J(x) = \alpha_i x + \beta_i$ if $x \in \mathcal{R}_i$
- Sketch:
  - any control action which guarantees decrease of the Lyapunov function is stabilizing
  - formally:

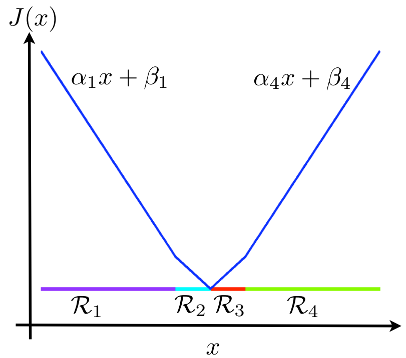
$$\boxed{\begin{aligned} & u_{\min} \le u \le u_{\max} \\ & x \in \mathcal{R}_i \\ & x^+ \in \mathcal{R}_j \\ & J(x^+) - J(x) \le -\gamma \end{aligned}}$$



Set of Stabilizing Controllers

- Given:
  - linear discrete-time system $x^+ = Ax + Bu$
  - PWA Lyapunov function $J(x) = \alpha_i x + \beta_i$ if $x \in \mathcal{R}_i$
- Sketch:
  - any control action which guarantees decrease of the Lyapunov function is stabilizing
  - formally:

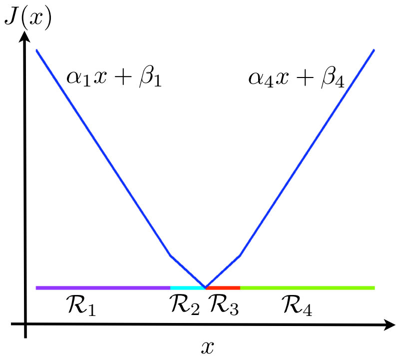
$$\boxed{\begin{aligned} & u_{\min} \le u \le u_{\max} \\ & H_i x \le K_i \\ & x^+ \in \mathcal{R}_j \\ & J(x^+) - J(x) \le -\gamma \end{aligned}}$$



Set of Stabilizing Controllers

- Given:
  - linear discrete-time system $x^+ = Ax + Bu$
  - PWA Lyapunov function $J(x) = \alpha_i x + \beta_i$ if $x \in \mathcal{R}_i$
- Sketch:
  - any control action which guarantees decrease of the Lyapunov function is stabilizing
  - formally:

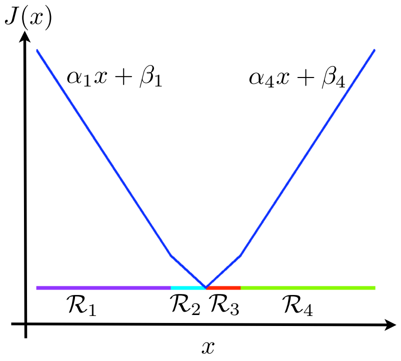
$$\boxed{\begin{aligned} & u_{\min} \le u \le u_{\max} \\ & H_i x \le K_i \\ & H_j(Ax + Bu) \le K_j \\ & J(x^+) - J(x) \le -\gamma \end{aligned}}$$



Set of Stabilizing Controllers

- Given:
  - linear discrete-time system $x^+ = Ax + Bu$
  - PWA Lyapunov function $J(x) = \alpha_i x + \beta_i$ if $x \in \mathcal{R}_i$
- Sketch:
  - any control action which guarantees decrease of the Lyapunov function is stabilizing
  - formally:

$$\boxed{\begin{aligned} & u_{\min} \le u \le u_{\max} \\ & H_i x \le K_i \\ & H_j(Ax + Bu) \le K_j \\ & [\alpha_j(Ax + Bu) + \beta_j] - [\alpha_i x + \beta_i] \le -\gamma \end{aligned}}$$

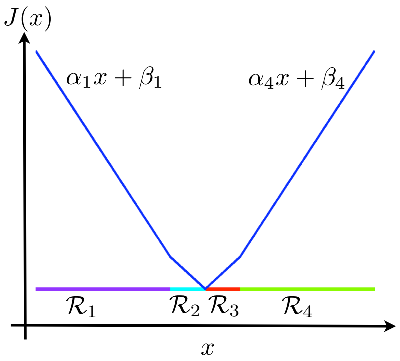


Set of Stabilizing Controllers

- Given:
  - linear discrete-time system $x^+ = Ax + Bu$
  - PWA Lyapunov function $J(x) = \alpha_i x + \beta_i$ if $x \in \mathcal{R}_i$
- Sketch:
  - any control action which guarantees decrease of the Lyapunov function is stabilizing
  - formally:

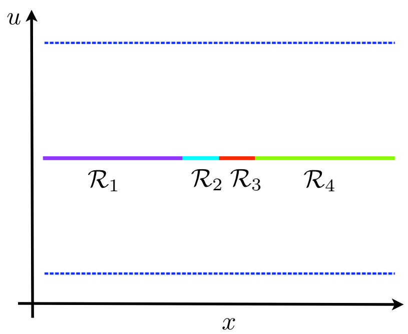
$$\boxed{\begin{aligned} & u_{\min} \le u \le u_{\max} \\ & H_i x \le K_i \\ & H_j(Ax + Bu) \le K_j \\ & [\alpha_j(Ax + Bu) + \beta_j] - [\alpha_i x + \beta_i] \le -\gamma \end{aligned}}$$

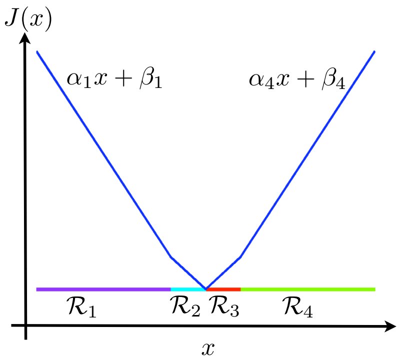
  - for each *i*, *j* pair the constraints are linear, thus they form a polytope in the state-input space



Set of Stabilizing Controllers

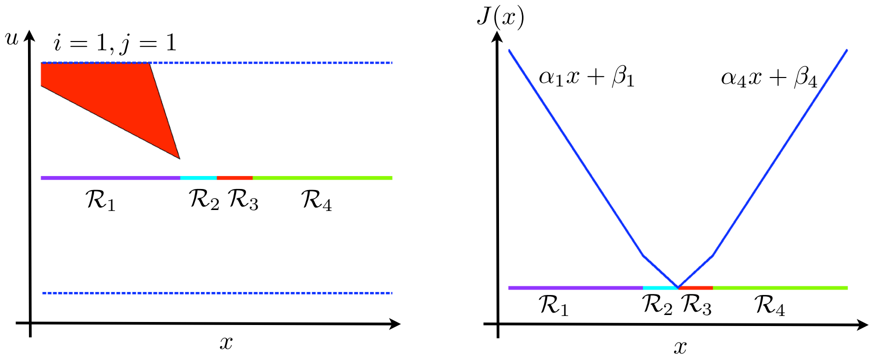
- Given:
  - linear discrete-time system $x^+ = Ax + Bu$
  - PWA Lyapunov function $J(x) = \alpha_i x + \beta_i$ if $x \in \mathcal{R}_i$
- Sketch:
  - any control action which guarantees decrease of the Lyapunov function is stabilizing





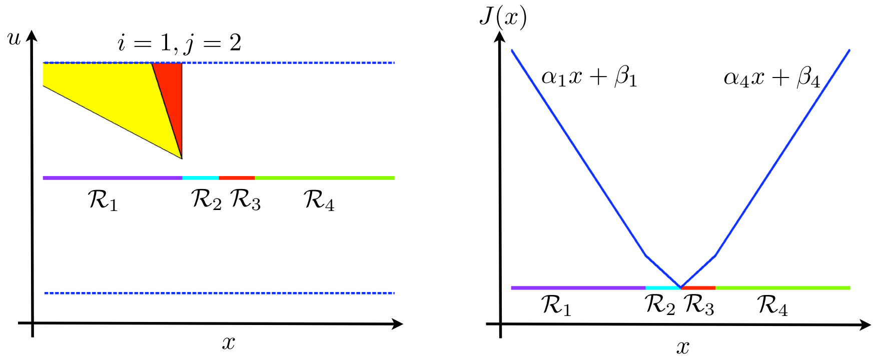
Set of Stabilizing Controllers

- Given:
  - linear discrete-time system $x^+ = Ax + Bu$
  - PWA Lyapunov function $J(x) = \alpha_i x + \beta_i$ if $x \in \mathcal{R}_i$
- Sketch:
  - any control action which guarantees decrease of the Lyapunov function is stabilizing



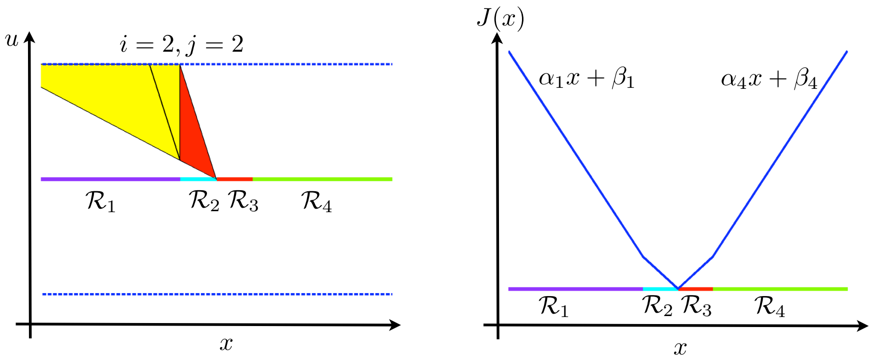
Set of Stabilizing Controllers

- Given:
  - linear discrete-time system $x^+ = Ax + Bu$
  - PWA Lyapunov function $J(x) = \alpha_i x + \beta_i$ if $x \in \mathcal{R}_i$
- Sketch:
  - any control action which guarantees decrease of the Lyapunov function is stabilizing



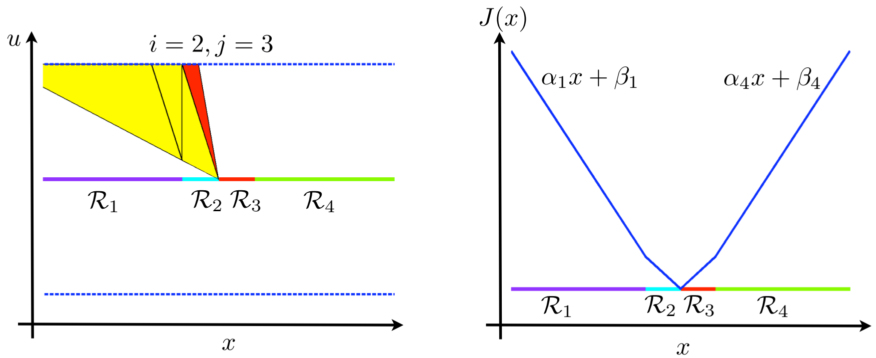
Set of Stabilizing Controllers

- Given:
  - linear discrete-time system $x^+ = Ax + Bu$
  - PWA Lyapunov function $J(x) = \alpha_i x + \beta_i$ if $x \in \mathcal{R}_i$
- Sketch:
  - any control action which guarantees decrease of the Lyapunov function is stabilizing



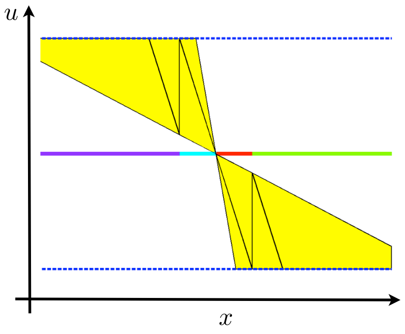
Set of Stabilizing Controllers

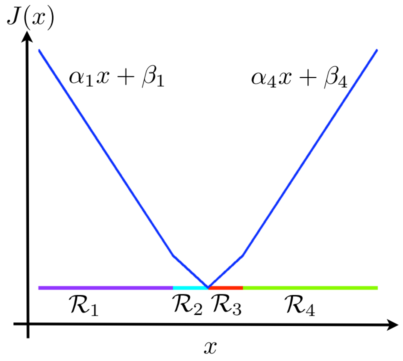
- Given:
  - linear discrete-time system $x^+ = Ax + Bu$
  - PWA Lyapunov function $J(x) = \alpha_i x + \beta_i$ if $x \in \mathcal{R}_i$
- Sketch:
  - any control action which guarantees decrease of the Lyapunov function is stabilizing



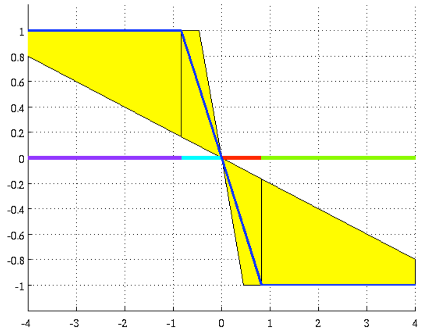
Set of Stabilizing Controllers

- Given:
  - linear discrete-time system $x^+ = Ax + Bu$
  - PWA Lyapunov function $J(x) = \alpha_i x + \beta_i \text{ if } x \in \mathcal{R}_i$
- Sketch:
  - any control action which guarantees decrease of the Lyapunov function is stabilizing



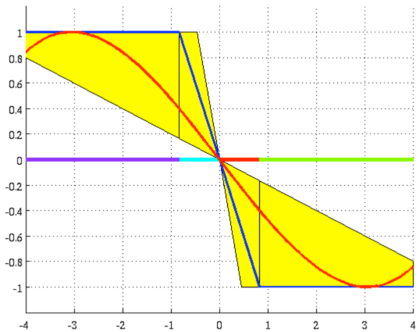


Set of Stabilizing Controllers



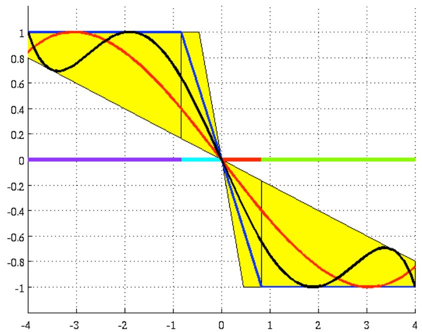
- This is not the set of **<u>all</u>** stabilizing controllers!
- Merely it is a set of inputs which render a given PWA Lyapunov function a Control Lyapunov function

Finding the Polynomial



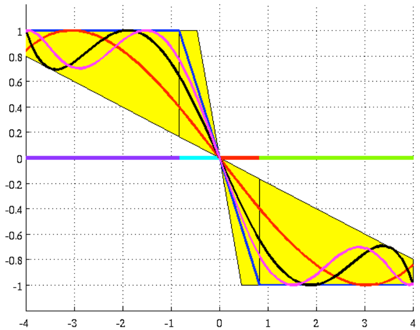
- Objectives:
  - the polynomial must never leave the set
  - it should be close to the optimal feedback
- Tuning parameter: degree of the polynomial

Finding the Polynomial



- Objectives:
  - the polynomial must never leave the set
  - it should be close to the optimal feedback
- Tuning parameter: degree of the polynomial

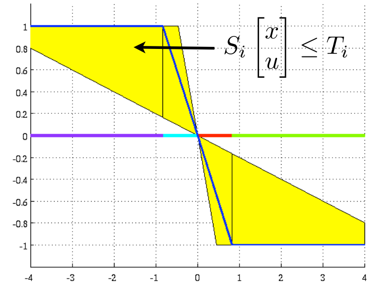
Finding the Polynomial



- Objectives:
  - the polynomial must never leave the set
  - it should be close to the optimal feedback
- Tuning parameter: degree of the polynomial

Finding the Polynomial

- Fix the degree of $\tilde{u}(x) = a_0 + a_1 x + \cdots + a_n x^n$



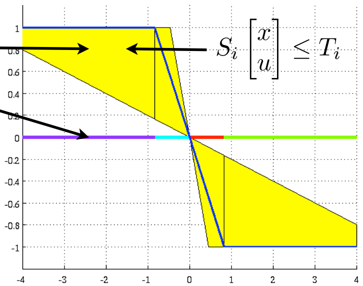
Finding the Polynomial

- Fix the degree of $\tilde{u}(x) = a_0 + a_1 x + \cdots + a_n x^n$
- Optimize for performance:

$$\min_{\mathbf{a}=[a_0,\ldots,a_n]} \quad \|u^*(x) - \tilde{u}(x)\|_1$$

$$\text{s.t} \quad T_i - S_i \begin{bmatrix} x \\ \tilde{u}(x) \end{bmatrix} \geq 0$$

$$\forall x \in \{x \mid K_i - H_i x \geq 0\}$$



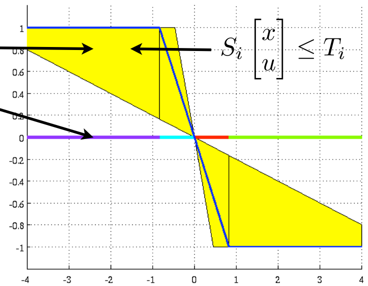
Finding the Polynomial

- Fix the degree of $\tilde{u}(x) = a_0 + a_1 x + \cdots + a_n x^n$
- Optimize for performance:

$$\min_{\mathbf{a}=[a_0,\ldots,a_n]} \quad \|u^*(x) - \tilde{u}(x)\|_1$$

$$\text{s.t} \quad \boxed{T_i - S_i \begin{bmatrix} x \\ \tilde{u}(x) \end{bmatrix} \geq 0}$$

$$\forall x \in \{x \mid K_i - H_i x \geq 0\}$$



- I.e. find coefficients for which the polynomials are non-negative over corresponding regions

Polya's Theorem

- Non-negativity of polynomial $p(\mathbf{a}, x)$ over a polytope is related to non-negativity of coefficients of the extended polynomial over vertices

$$p(\mathbf{a}, x) \cdot \left(\sum x_i \right)^M$$

- Notice that the coefficients enter linearly:

$$\begin{aligned} \min_{\mathbf{a}=[a_0,\ldots,a_n]} \quad & \|u^*(x) - \tilde{u}(x)\|_1 \\ \text{s.t} \quad & T_i - S_i \begin{bmatrix} x \\ \tilde{u}(x) \end{bmatrix} \geq 0, \quad i = 1, \ldots, N \\ & \forall x \in \{x \mid K_i - H_i x \geq 0\}, \quad i = 1, \ldots, N \\ & \tilde{u}(x) = a_0 + a_1 x + \cdots + a_n x^n \end{aligned}$$

- Therefore they can be found by a single linear program!

Numerical Examples

	# of regions	Explicit MPC	Polynomial	Performance drop
2 states	146	13 000 B	24 B (degree 3)	24%
	170	16 000 B	40 B (degree 5)	18%
3 states	66	11 000 B	60 B (degree 5)	31%
	122	19 000 B	60 B (degree 5)	5%

Conclusions

PROs:

- the polynomial is easily found using linear programming
- extremely low memory footprint (< 100 bytes)
- guarantees stability, feasibility, and bounded performance drop
- works for linear and PWA systems

CONs:

- suboptimality
- Polya's theorem is just a sufficient condition
- expensive symbolic computation of the extended Polya's polynomial