Examples of Relaxed Dynamic Programming

Bo Lincoln, Anders Rantzer
Dept. of Automatic Control
Lund Institute of Technology, Sweden
lincoln @ control.lth.se

Relaxed Dynamic Programming

Dynamic Programming (DP) is a beautiful idea – but seldomly used.

Most important part of DP: The value function $V(x)$:

- Gives the cost to start from state x.
- Implicitly gives a control law .
- Optimal $V(x)$ may be very hard to represent (except in special cases such as shortest-path-problems or LQG).

Relaxed Dynamic Programming

Dynamics of a system:

$$x(n+1) = f\left(x(n), u(n)\right)$$

Cost function:

$$J = \sum_{n=0}^{\infty} l(x(n), u(n))$$

Bellman's equation:

$$V^*(x) = \min_u \Big\{ V^*(f(x,u) + l(x,u) \Big\}$$

$V^*(x)$ is the optimal value function.

Relaxed Dynamic Programming

Value iteration:

$$V_{k+1}(x) = \min_u \left\{ V_k(f(x,u) + l(x,u) \right\}$$

Main idea – Relaxed value iteration :

$$\underline{V}_{k+1}(x) = \min_u \left\{ V_k(f(x,u)) + \underline{l}(x,u) \right\}$$

$$\leq V_{k+1}(x) \leq$$

$$\min_u \left\{ V_k(f(x,u)) + \overline{l}(x,u) \right\} = \overline{V}_{k+1}(x)$$

where

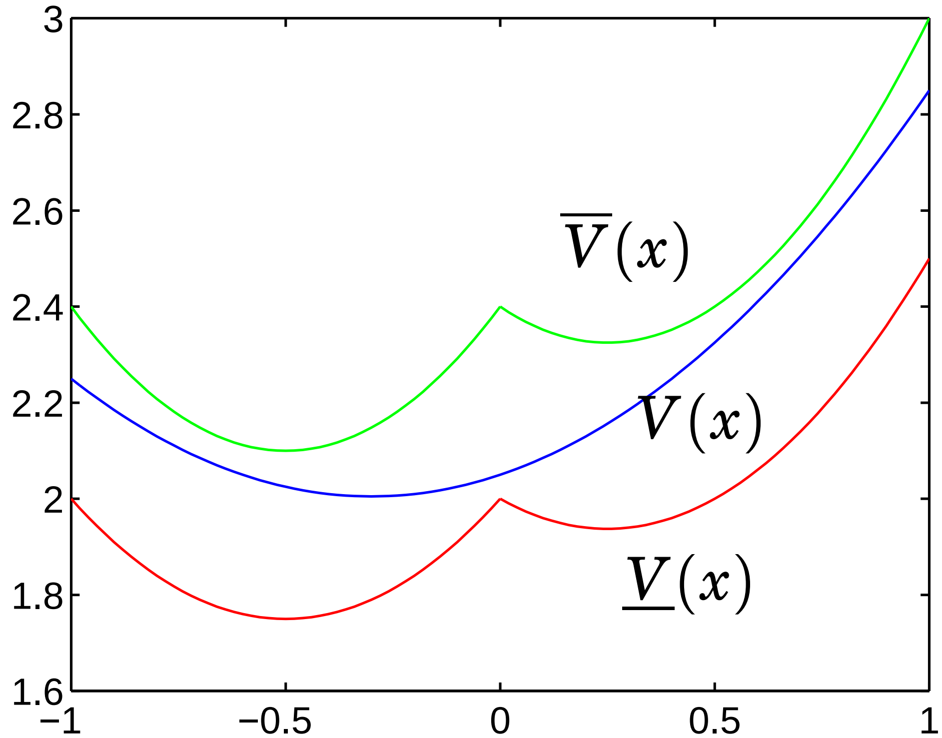
$$\overline{l}(x,u) = \overline{\alpha} l(x,u), \quad \overline{\alpha} \geq 1$$
$$\underline{l}(x,u) = \underline{\alpha} l(x,u), \quad \underline{\alpha} \leq 1.$$

Relaxed Dynamic Programming

$$\underline{V}_{k+1}(x) = \min_u \Big\{ V_k\big(f(x,u)\big) + \underline{l}(x,u) \Big\}$$

$$\leq V_{k+1}(x) \leq$$

$$\min_u \Big\{ V_k\big(f(x,u)\big) + \overline{l}(x,u) \Big\} = \overline{V}_{k+1}(x)$$



Relaxed Dynamic Programming

Iterating the inequality

$$\underline{V}_{k+1}(x) = \min_u \Big\{ V_k\big(f(x,u)\big) + \underline{l}(x,u) \Big\}$$

$$\leq V_{k+1}(x) \leq$$

$$\min_u \Big\{ V_k\big(f(x,u)\big) + \overline{l}(x,u) \Big\} = \overline{V}_{k+1}(x)$$

leads to

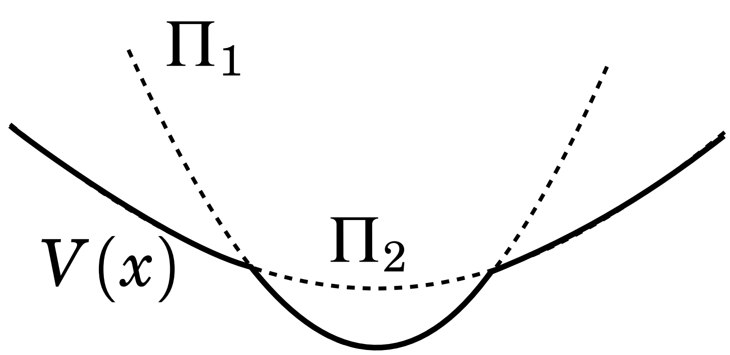
$$\underline{\alpha} \min_u \sum_{n=0}^{k} l(x(n), u(n)) \leq V_k(x) \leq \overline{\alpha} \min_u \sum_{n=0}^{k} l(x(n), u(n))$$

Applications

Switched linear systems:

- Finite set of system matrices (linear system), quadratic cost.
- Controller chooses both continuous control signal u and system dynamics. No autonomous switching.
- Value function on form

$$V(x) = \min_i x^T \Pi_i x$$

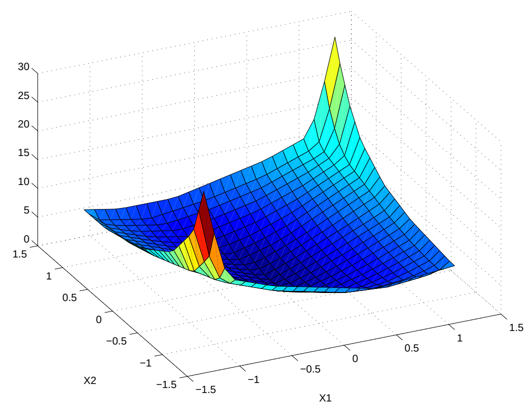


Applications

Piecewise linear cost:

- LTI system with constraints.
- Piecewise linear cost.
- Value function on form

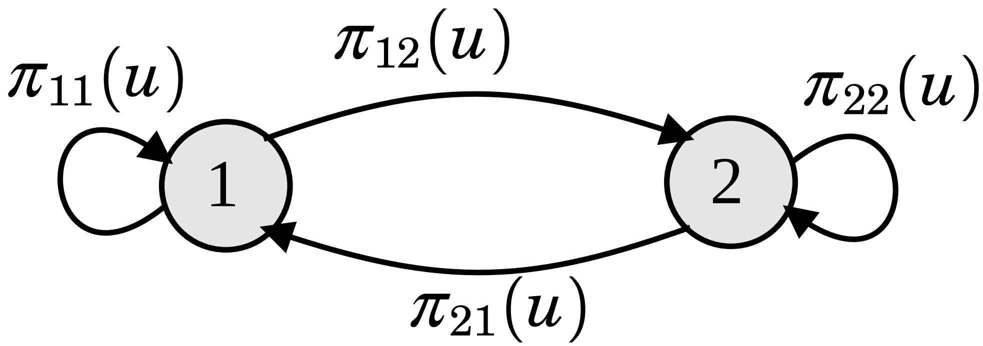
$$V(x) = \max_i \Pi_i^T \begin{bmatrix} x \\ u \\ 1 \end{bmatrix}$$



Applications

Partially Observable Markov Decision Networks (POMDPs):

- Markov control problem with unknown state.
- Boils down to the “Piecewise linear cost problem”.



Which problems can use Relaxed DP?

The main issue in each iteration is finding $V_{k+1}(x)$:

$$\underline{V}(x) = \min_u \Big\{ V_k\big(f(x,u)\big) + \underline{l}(x,u) \Big\}$$

$$\leq V_{k+1}(x) \leq$$

$$\min_u \Big\{ V_k\big(f(x,u)\big) + \overline{l}(x,u) \Big\} = \overline{V}(x)$$

In the previous applications:

- Add elements from $\underline{V}(x)$ or $\overline{V}(x)$ to $V_{k+1}(x)$ until both inequalities hold.
- Typical inequality test: LP or LMI.

New Applications

Internet routing tables:

Relaxed DP can be used to decrease size of routing tables.
Idea: Hosts with similar addresses are often on approximately the same distance from some source node.

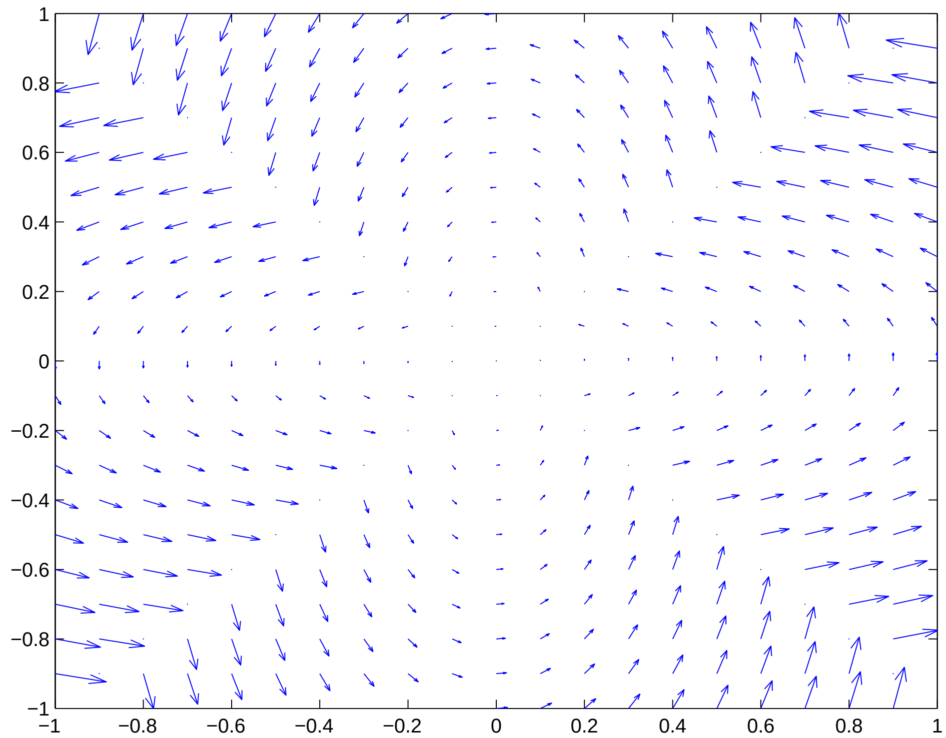
Piecewise Linear Quadratic Control:

Systems with *state-dependent* switching are harder than controlled switchings.

Piecewise Linear Quadratic Control

$$x(n+1) = \Phi_i x(n) + \Gamma_i u(n), \quad i = \begin{cases} 1 & x^T G x \geq 0 \Leftrightarrow x \in \Omega_1 \\ 2 & x^T G x < 0 \Leftrightarrow x \in \Omega_2 \end{cases}$$

$$J = \sum \begin{bmatrix} x \\ u \end{bmatrix}^T Q_i \begin{bmatrix} x \\ u \end{bmatrix}$$



Value function

Let, for example, $V_0 = x^T \Pi x$.

Then Bellman's equation gives

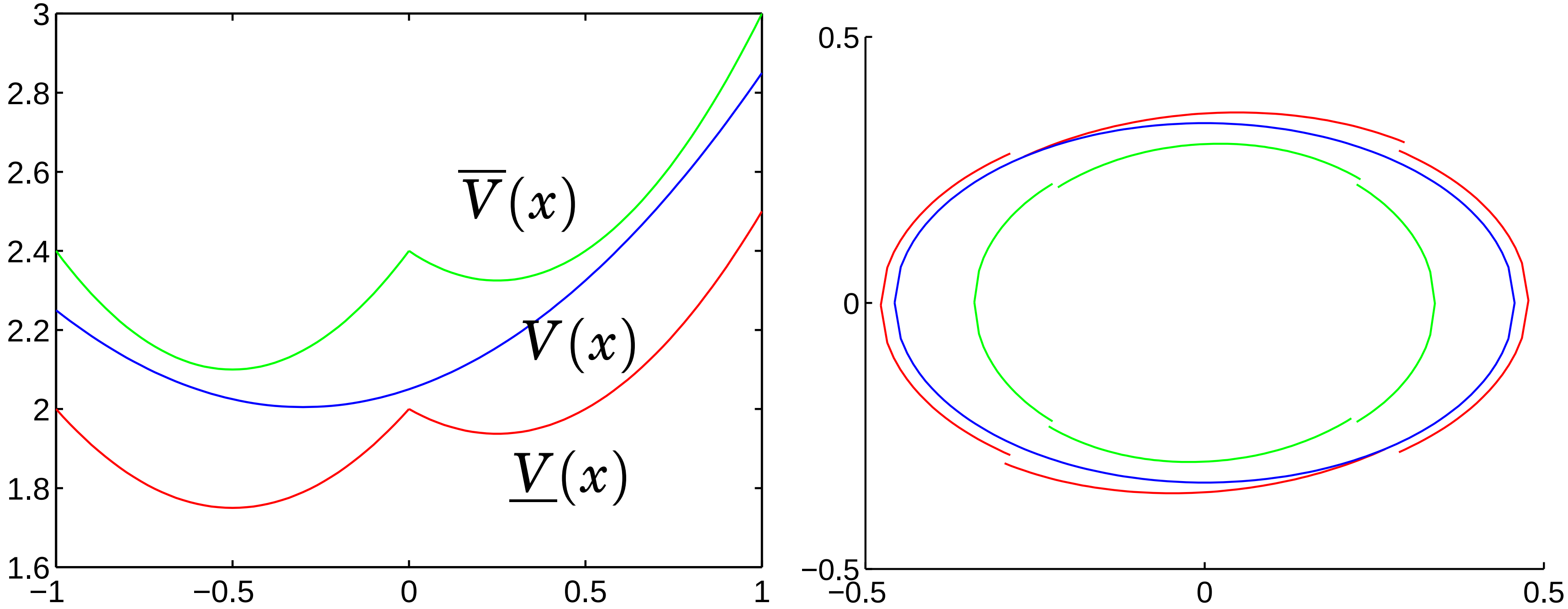
$$V_1(x) = \begin{cases} \min\limits_u \left(V(\Phi_1 x + \Gamma_1 u) + \begin{bmatrix} x \\ u \end{bmatrix}^T Q_1 \begin{bmatrix} x \\ u \end{bmatrix} \right) = x^T \Pi_1 x & x \in \Omega_1 \\ \min\limits_u \left(V(\Phi_2 x + \Gamma_2 u) + \begin{bmatrix} x \\ u \end{bmatrix}^T Q_2 \begin{bmatrix} x \\ u \end{bmatrix} \right) = x^T \Pi_2 x & x \in \Omega_2 \end{cases}$$

This is not on the same form as $V_0(x)$, as it contains the condition $x \in \Omega_i$. Using exact value iteration, the number of regions grows exponentially.

Try relaxed DP.

Relaxed value function

Idea: Use a value function that cannot describe $V^*(x)$ exactly.



Use for example

$$V_k(x) = \min_{\Pi \in P_k} x^T \Pi x$$

Relaxed value iteration

Assume $V_k(x) = \min_{\Pi \in P_k} x^T \Pi x$.

Using Bellman's equation we calculate

$$\overline{V}_{k+1}(x) = \begin{cases} \min_{\Pi \in \overline{P}_{k+1}^1} x^T \Pi x & x \in \Omega_1 \\ \min_{\Pi \in \overline{P}_{k+1}^2} x^T \Pi x & x \in \Omega_2 \end{cases}$$

and

$$\underline{V}_{k+1}(x) = \begin{cases} \min_{\Pi \in \underline{P}_{k+1}^1} x^T \Pi x & x \in \Omega_1 \\ \min_{\Pi \in \underline{P}_{k+1}^2} x^T \Pi x & x \in \Omega_2 \end{cases}$$

Algorithm to find $V_{k+1}(x)$

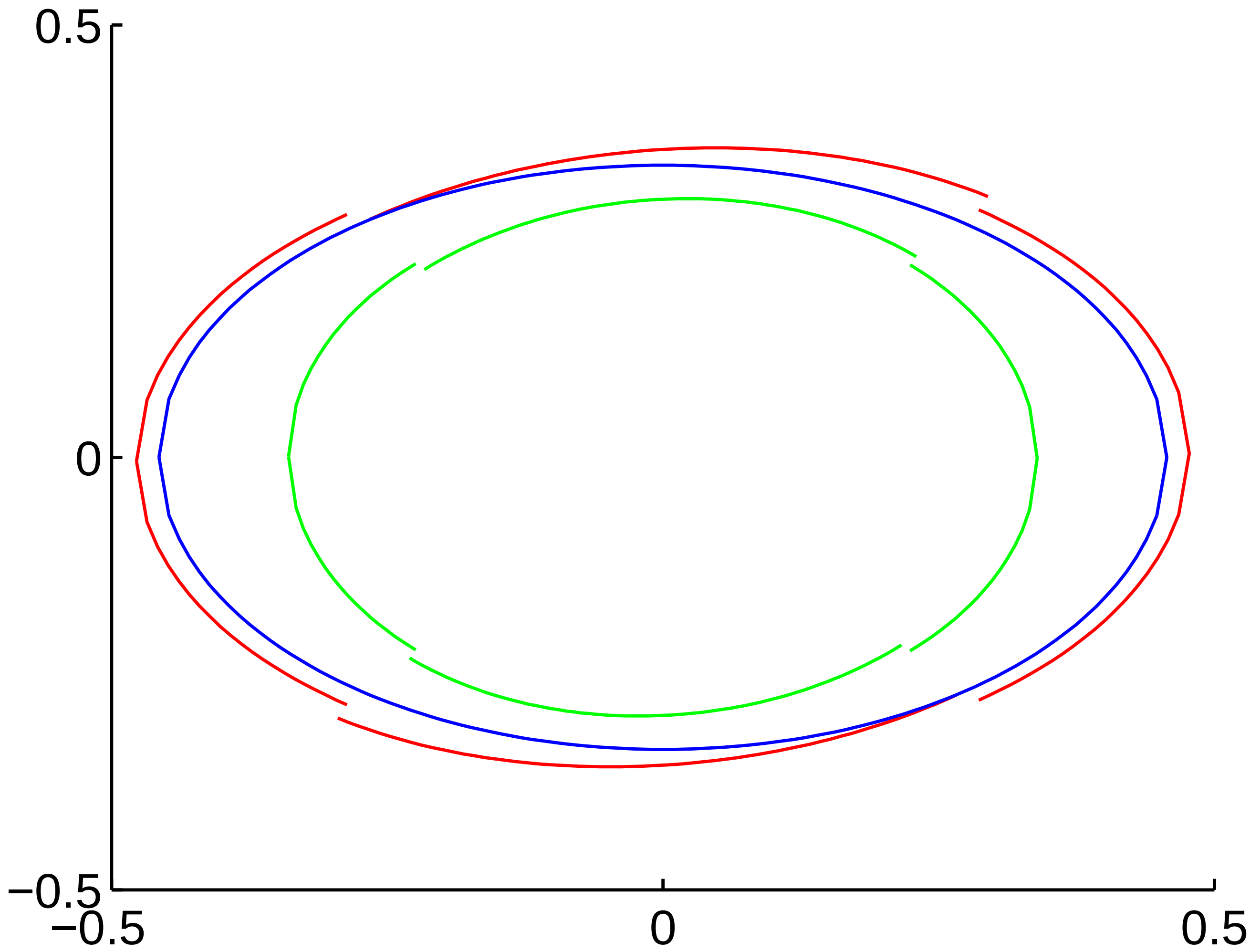
1. Define $V_{k+1}(x) = \min_{\Pi \in P_{k+1}} x^T \Pi x$ and let $P_{k+1} = \emptyset$.
2. If there exists $\overline{\Pi} \in \left\{ \overline{P}^1_{k+1} \cup \overline{P}^2_{k+1} \right\}$ and x such that $x^T \overline{\Pi} x < V_{k+1}(x)$ (i.e. the upper bound does not hold) then find a Π such that

$$
\begin{aligned}
x^T \Pi x &\leq x^T \overline{\Pi} x \quad \forall x \in \Omega_i \\
x^T \Pi x &\geq x^T \underline{\Pi} x \quad \forall x \in \Omega_1, \ \underline{\Pi} \in \underline{P}^1_{k+1} \\
x^T \Pi x &\geq x^T \underline{\Pi} x \quad \forall x \in \Omega_2, \ \underline{\Pi} \in \underline{P}^2_{k+1}
\end{aligned}
$$

   Add this Π to P_{k+1}.
3. Repeat from 2.

Step 2 can be relaxed to a Linear Matrix Inequality, LMI .

Algorithm to find $V_{k+1}(x)$



Note: The LMI may fail for several reasons $\Rightarrow$ Increase slack.

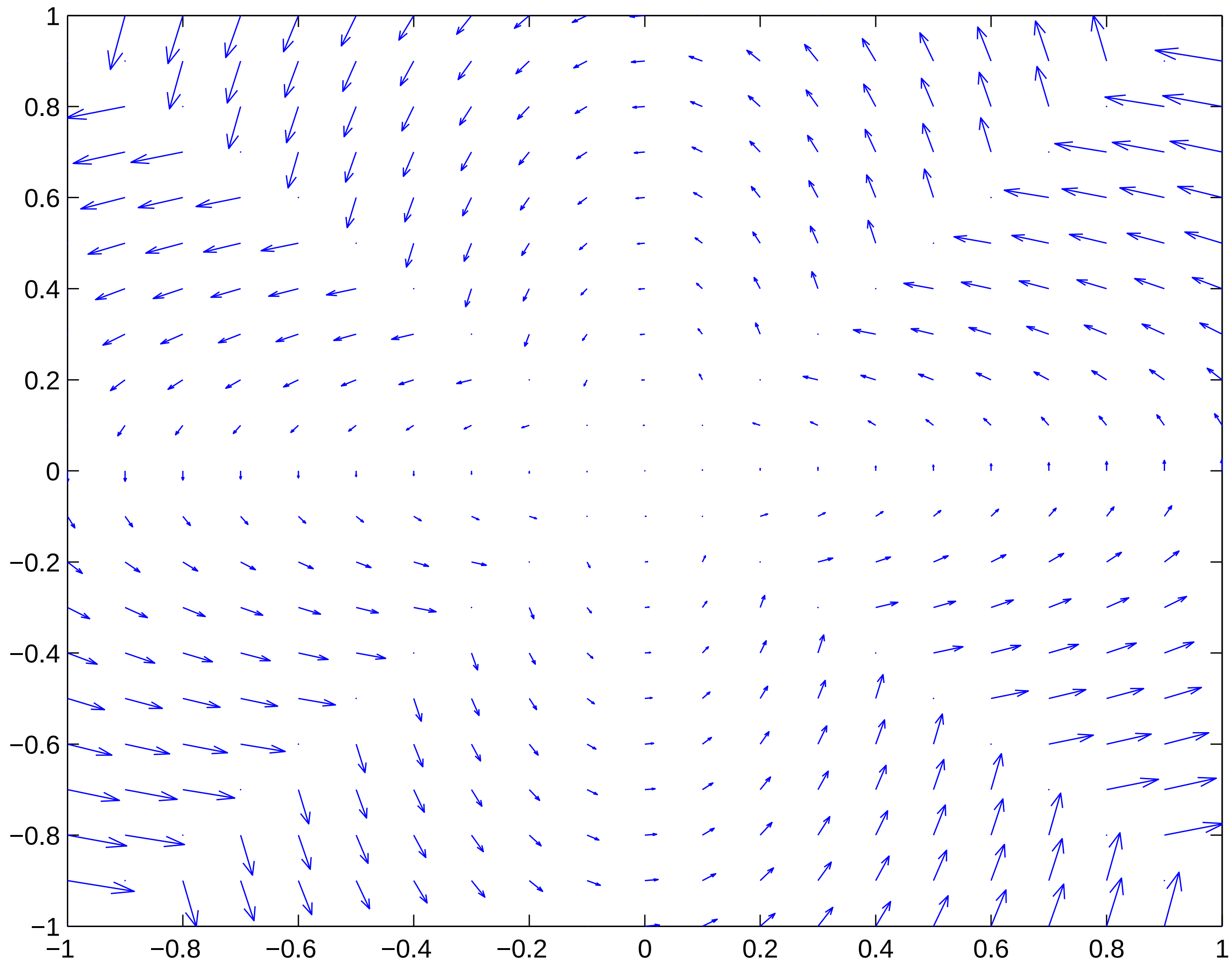
Example 1: 2D Flower

$$x(n+1) = \Phi_i x(n) + \Gamma_i u(n), \quad i = \begin{cases} 1 & x^T G x \geq 0 \Leftrightarrow x \in \Omega_1 \\ 2 & x^T G x < 0 \Leftrightarrow x \in \Omega_2 \end{cases}$$

$$\Phi_1 = \begin{bmatrix} 1.0000 & -0.0010 \\ 0.0050 & 0.9999 \end{bmatrix} \quad \Gamma_1 = \begin{bmatrix} 0.0100 \\ 0.0000 \end{bmatrix} \quad G = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$\Phi_2 = \begin{bmatrix} 1.0000 & -0.0050 \\ 0.0010 & 0.9999 \end{bmatrix} \quad \Gamma_2 = \begin{bmatrix} 0.0100 \\ 0.0000 \end{bmatrix} \quad Q_1 = Q_2 = I$$

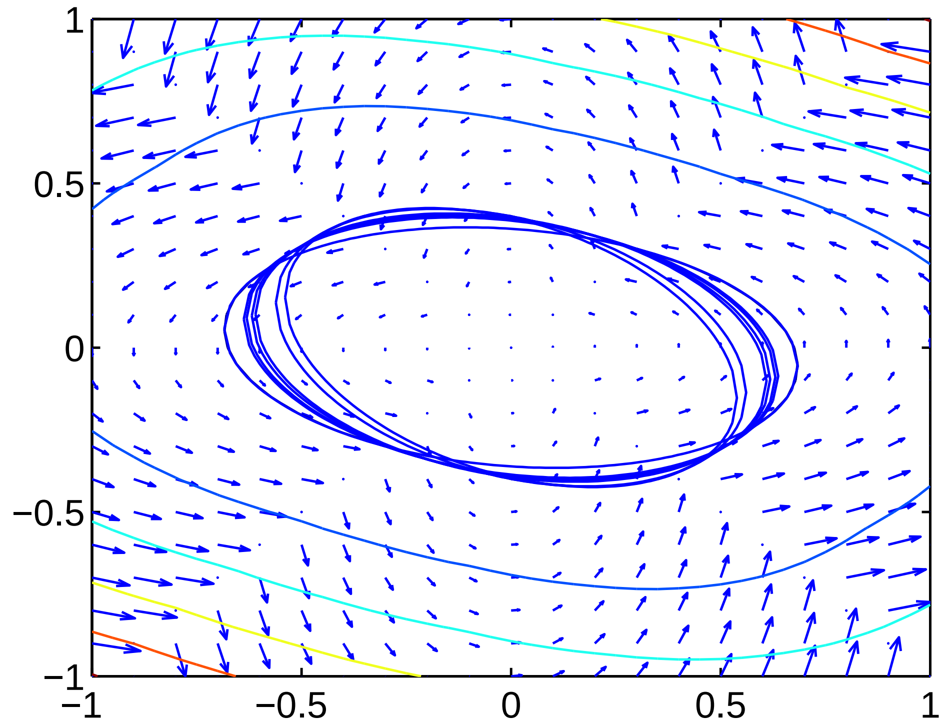
Example 1: 2D Flower



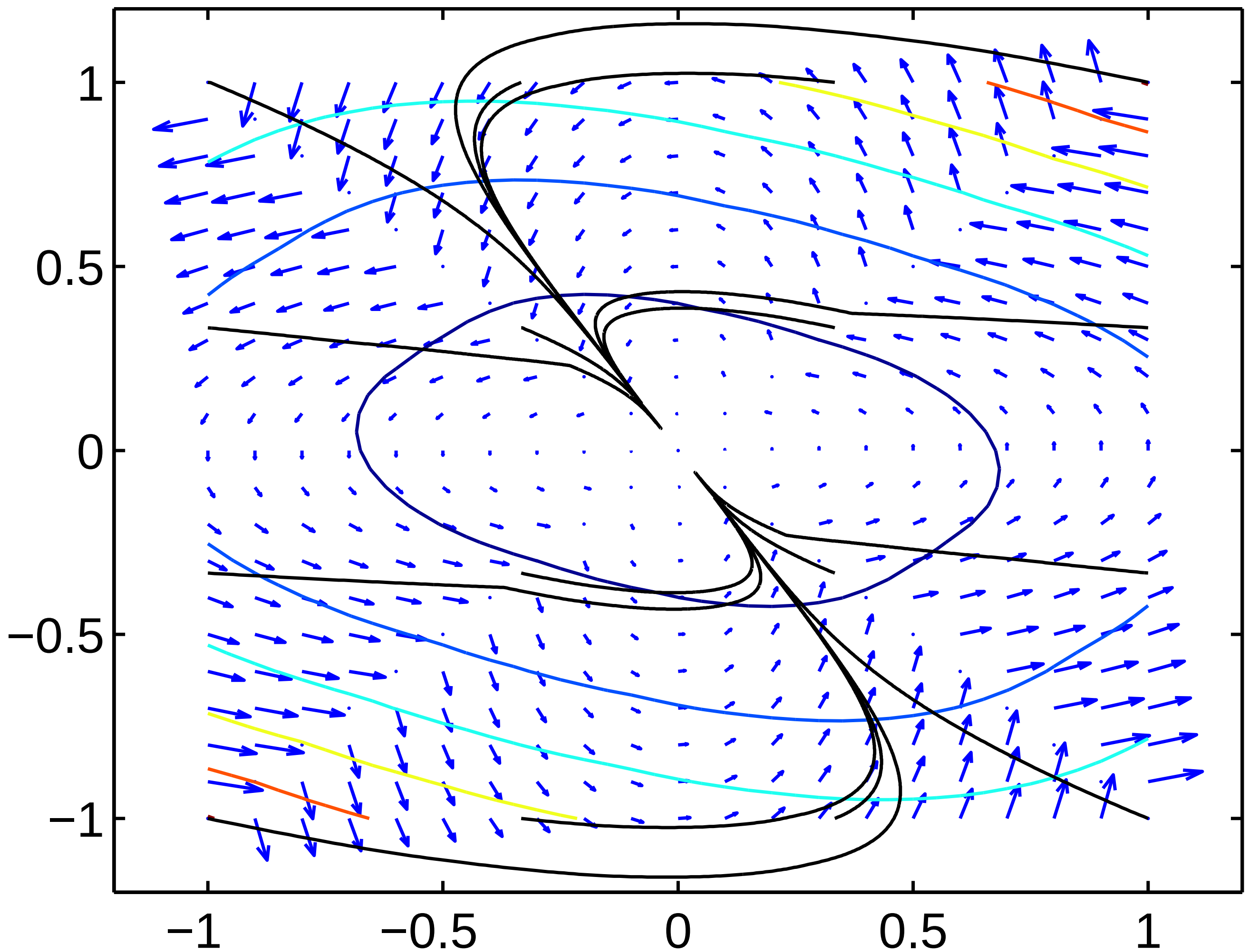
Example 1: 2D Flower

Using the relaxation $\overline{\alpha} = 2$ and $\underline{\alpha} = \frac{1}{2}$, a steady state solution $V(x)$ is found within 100 iterations.

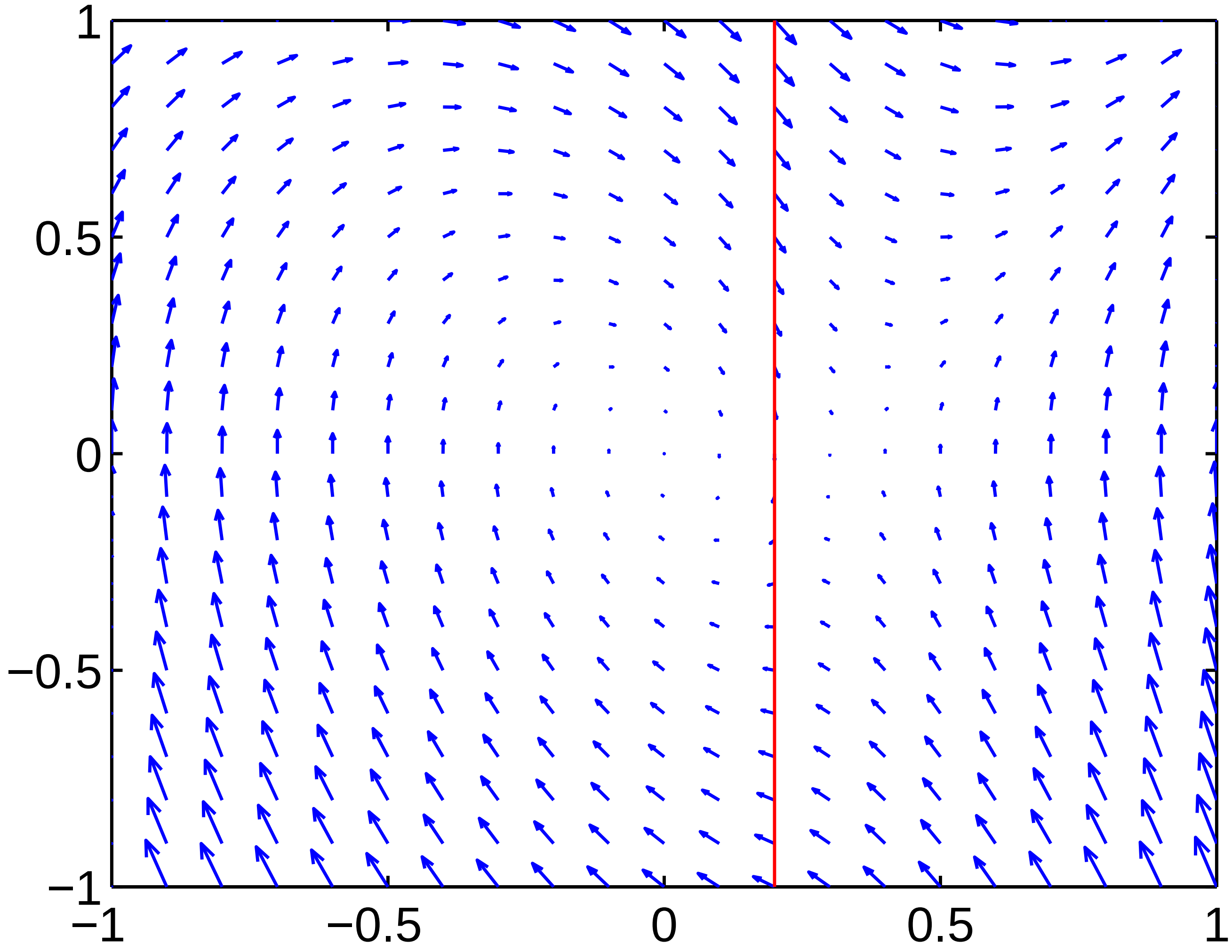
$V(x)$ consists of seven quadratic functions.



Example 1: 2D Flower



Example 2: Affine System



Example 2: Affine System

Affine system created by letting $x = \begin{bmatrix} x_1 \\ x_2 \\ 1 \end{bmatrix}$, and viewing it as a 3D system.

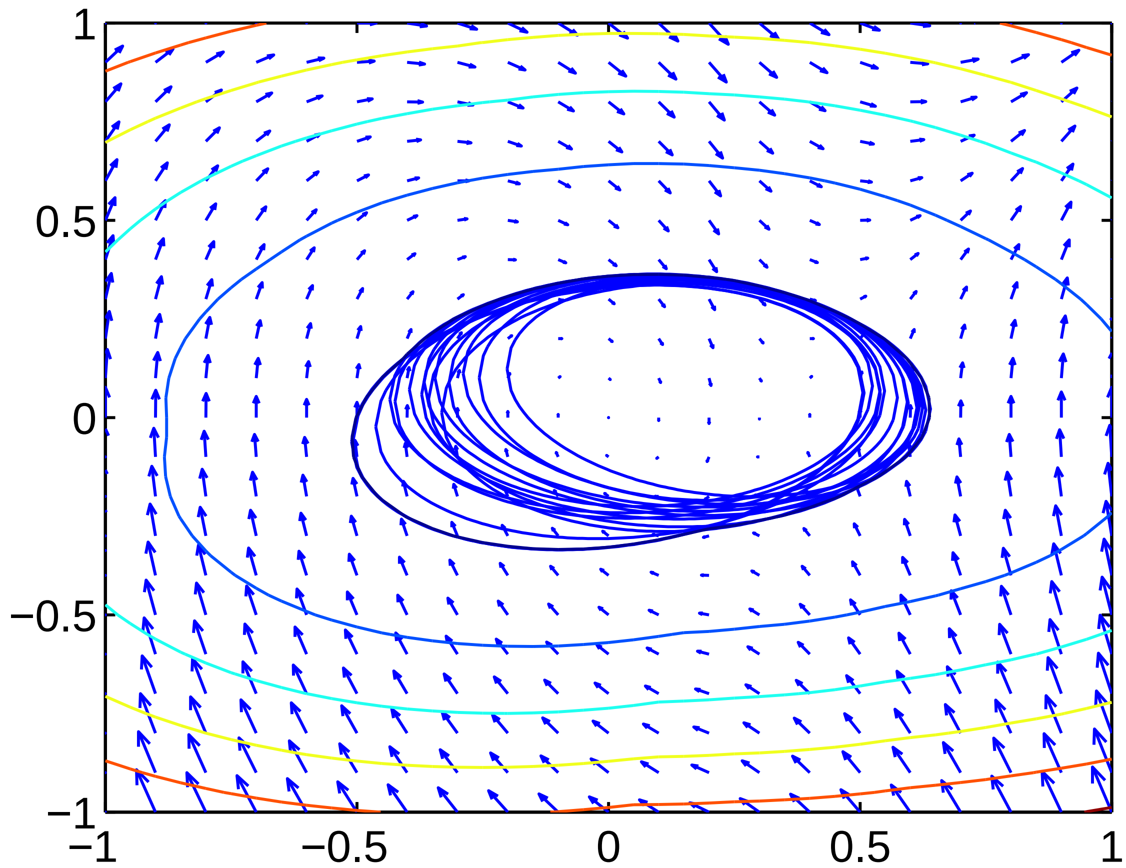
To obtain the $x_1 = 0.2$ dynamics switch, we define

$$x^T G x = (x + 5)^2 - (5 + 0.2)^2$$

State space is limited by $x_1^2 + x_2^2 \leq 9$.

Example 2: Affine System

Using the relaxation $\overline{\alpha} = 2$ and $\underline{\alpha} = \frac{1}{2}$, $V_{100}(x)$ contains 15 quadratic functions.



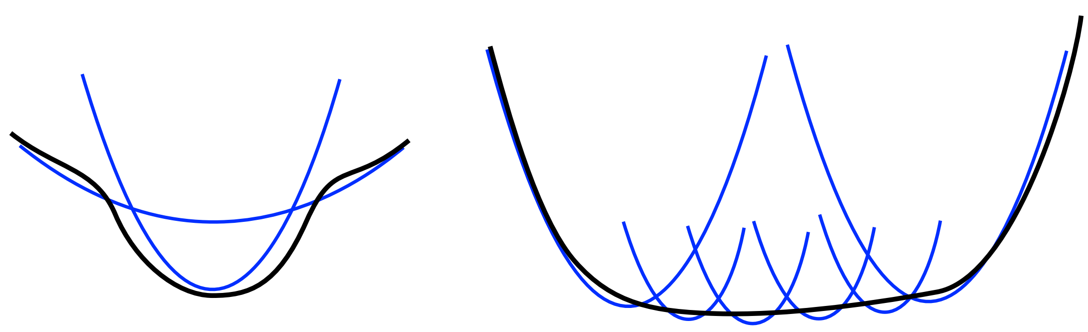
Pros and Cons

Cons:

- LMI and S-procedure sometimes conservative.
- Discontinuous dynamics can give impossible boundaries.
- The value function

$$V(x) = \min_{\Pi \in P} x^T \Pi x$$

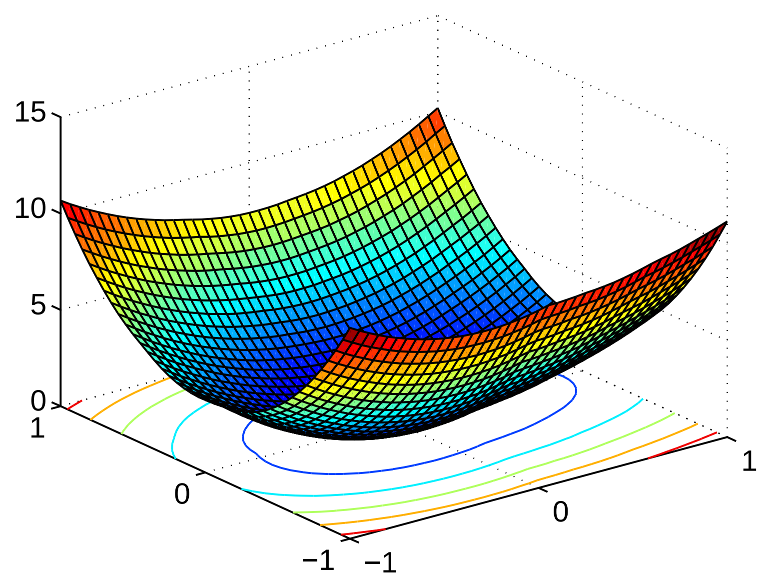
is not always a good way of approximating of $V^*(x)$



Pros and Cons

Pros:

- Very simple representation of non-convex cost functions.
- Can get close-to-optimal feedback controllers for piecewise linear systems.
- Easily implementable controller (regions of linear/affine controllers).



Conclusions

A new application of the relaxed DP method has been presented: Piecewise linear quadratic control . The application is in the early research stage.

- Relaxed DP enables use approximating value function parameterization.
- The value function repr. used in the application cannot represent $V^*(x)$, but works well as approximation.
- Key issue in using relaxed DP: Is there an algorithm to find a (simple) cost function in between two bounds?