### Mixed Integer Convex Programming Computational Intelligence, Lecture 11

by Sergei Savin

Spring 2022

#### CONTENT

- Mixed Integer Linear Programming (MILP)
- Mixed Integer Quadratic Programming (MIQP)
- Remarks
- Example: Footstep planning
- Big-M method relaxation
  - ▶ Basic idea
  - Systems of inequalities
  - ▶ Illustration, part 1
  - ▶ Illustration, part 2
  - Multiple variables
- Example: Footstep planning
  - Formulation as MIQP
  - Evenly spaced steps
  - ► Code, part 1
  - ▶ Code, part 2
  - ► Code, part 3
- Homework

## MIXED INTEGER LINEAR PROGRAMMING (MILP) General form

A general form of a mixed-integer linear program is:

minimize 
$$\mathbf{f}^{\top}\mathbf{x}$$
,

subject to 
$$\begin{cases} \mathbf{A}\mathbf{x} \leq \mathbf{b}, \\ \mathbf{C}\mathbf{x} = \mathbf{d}, \\ \mathbf{x}_{1}, ..., \mathbf{x}_{m} \in \mathbb{R}, \\ \mathbf{x}_{m+1}, ..., \mathbf{x}_{n} \in \mathbb{N}. \end{cases}$$
(1)

In other words, the only difference is that some of the variables (denoted above as  $\mathbf{y}$ ) are only allowed to assume pure integer values.

# MIXED INTEGER QUADRATIC PROGRAMMING (MIQP)

General form

A general form of a mixed-integer quadratic program is:

minimize 
$$\mathbf{x}^{\top} \mathbf{H} \mathbf{x} + \mathbf{f}^{\top} \mathbf{x}$$
,  
subject to 
$$\begin{cases} \mathbf{A} \mathbf{x} \leq \mathbf{b}, \\ \mathbf{C} \mathbf{x} = \mathbf{d}, \\ \mathbf{x}_{1}, ..., \mathbf{x}_{m} \in \mathbb{R}, \\ \mathbf{x}_{m+1}, ..., \mathbf{x}_{n} \in \mathbb{N}. \end{cases}$$
(2)

Other mixed-integer convex programs can be constructed likewise from their pure convex counterparts.

#### REMARKS

- Mixed-integer convex programs are not convex, even if the name seems to suggest otherwise.
- The "convex program" in the phrase "mixed-integer convex program" should be understood as "if we fix values of the integer variables, or relax them into reals, the result will be a convex program".
- Mixed integer programs are usually solved using branch and bound algorithms; in worse case scenario, the algorithms performs exhaustive search.
- In robotics applications, integer variables in mixed integer programs are often restricted to binary values, i.e.  $\mathbf{y} \in \{0,1\}^n$ . It is still called "mixed-integer" in the literature, although phrases such as "mixed-binary" or "binary constraints" are not rare.

#### EXAMPLE: FOOTSTEP PLANNING

#### Problem statement

Given N convex regions defined by linear inequalities  $\{\mathbf{x}: \mathbf{A}_i\mathbf{x} \leq \mathbf{b}_i\}$  (which is called H-polytope representation), find a sequence of K points (footsteps) from the given starting point to the given goal point, such that all footsteps lie in one of the convex regions.

minimize 
$$||\mathbf{x}_1 - \mathbf{x}_{\text{start}}|| + ||\mathbf{x}_K - \mathbf{x}_{\text{goal}}||,$$
  
subject to  $\exists \{\mathbf{A}_j, \mathbf{b}_j\} \in \Omega \text{ s.t. } \mathbf{A}_j \mathbf{x}_i \leq \mathbf{b}_j$  (3)

where  $\Omega = \{\{\mathbf{A}_1, \mathbf{b}_1\}, \{\mathbf{A}_2, \mathbf{b}_2\}, ..., \{\mathbf{A}_N, \mathbf{b}_N\}\}$ . This is not a convex program.

## BIG-M METHOD RELAXATION Basic idea

One of the key methods associated with the use of mixed-integer programming in robotics is the big-M method.

Assume you have two inequalities,  $\mathbf{a}_1^{\top} \mathbf{x} \leq b_1$  and  $\mathbf{a}_2^{\top} \mathbf{x} \leq b_2$ , and you are happy if at least one of them holds. Define a new binary variables  $c_1, c_2 \in \{0, 1\}$  and find a big enough constant M, such that  $\mathbf{a}_1^{\top} \mathbf{x} \leq b_1 + M$  and  $\mathbf{a}_2^{\top} \mathbf{x} \leq b_2 + M$  holds for all of your domain (of the part of it you are interested in). Then you write your constraints as:

$$\begin{cases}
\mathbf{a}_{1}^{\top} \mathbf{x} \leq b_{1} + M \cdot c_{1} \\
\mathbf{a}_{2}^{\top} \mathbf{x} \leq b_{2} + M \cdot c_{2} \\
c_{1} + c_{2} = 1 \\
c_{i} \in \{0, 1\}
\end{cases} \tag{4}$$

#### BIG-M METHOD RELAXATION

#### Systems of inequalities

It works the same way for the case when you have three (or two or more) systems of inequalities  $A_1x \leq b_1$ ,  $A_2x \leq b_2$ ,  $A_3x \leq b_3$  and are happy if at least one holds:

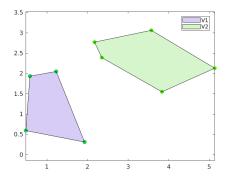
$$\begin{cases}
\mathbf{A}_{1}\mathbf{x} \leq \mathbf{b}_{1} + M \cdot \mathbf{1} \cdot (1 - c_{1}) \\
\mathbf{A}_{2}\mathbf{x} \leq \mathbf{b}_{2} + M \cdot \mathbf{1} \cdot (1 - c_{2}) \\
\mathbf{A}_{3}\mathbf{x} \leq \mathbf{b}_{3} + M \cdot \mathbf{1} \cdot (1 - c_{3}) \\
c_{1} + c_{2} + c_{3} = 1 \\
c_{i} \in \{0, 1\}
\end{cases}$$
(5)

where **1** is a vector of all ones. Notice that constraint  $c_1 + c_2 + c_3 = 1$  can be replaced with  $c_1 + c_2 + c_3 >= 1$ , allowing avoid relaxing more than one region.

### BIG-M METHOD RELAXATION

Illustration, part 1

Below are two H-polytops (convex regions represented by systems of inequalities):

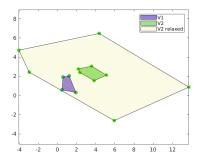


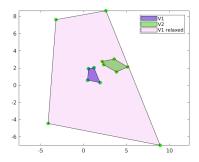
As we can see, their union represents a non-convex domain, and they have no intersection.

#### BIG-M METHOD RELAXATION

#### Illustration, part 2

Now, one of them is relaxed as described above. Notice that the intersection of the two polytopes is non-relaxed polytope.





## BIG-M METHOD RELAXATION Multiple variables

If you have multiple variables  $\mathbf{x}_1, ..., \mathbf{x}_K$ , and each should belong to at least one of the H-polytopes  $\{\mathbf{A}_i, \mathbf{b}_i\}$ , this can also be represented using big-M method:

$$\begin{cases}
\mathbf{A}_{1}\mathbf{x}_{k} \leq \mathbf{b}_{1} + M \cdot \mathbf{1} \cdot (1 - c_{1,k}) \\
\mathbf{A}_{2}\mathbf{x}_{k} \leq \mathbf{b}_{2} + M \cdot \mathbf{1} \cdot (1 - c_{2,k}) \\
\mathbf{A}_{3}\mathbf{x}_{k} \leq \mathbf{b}_{3} + M \cdot \mathbf{1} \cdot (1 - c_{3,k}) \\
c_{1,k} + c_{2,k} + c_{3,k} = 1 \\
c_{i,k} \in \{0, 1\} \\
k = 1, ..., K
\end{cases}$$
(6)

Notice that the only difference from the previous example is that now we have K sets of binary variables  $c_{1,k}$ ,  $c_{2,k}$  and  $c_{3,k}$ .

Formulation as MIQP

Using big-M relaxation we can now formulate the problem as follows:

minimize 
$$\mathbf{x}_{1, \dots, \mathbf{x}_{K}} = ||\mathbf{x}_{1} - \mathbf{x}_{\text{start}}|| + ||\mathbf{x}_{K} - \mathbf{x}_{\text{goal}}||,$$
subject to 
$$\begin{cases} \mathbf{A}_{i}\mathbf{x}_{k} \leq \mathbf{b}_{i} + M \cdot \mathbf{1} \cdot (1 - c_{i,k}), & i = 1, ..., N \\ \sum_{i=1}^{N} c_{i,k} = 1 \\ c \in \{0, 1\}^{N, K} \\ k = 1, ..., K \end{cases}$$

$$(7)$$

#### Evenly spaced steps

In order to make the footsteps evenly spaced we add cost on the distance between consequent steps:

minimize 
$$||\mathbf{x}_{1} - \mathbf{x}_{\text{start}}|| + ||\mathbf{x}_{K} - \mathbf{x}_{\text{goal}}|| + w \cdot \sum_{k=1}^{K-1} ||\mathbf{x}_{i+1} - \mathbf{x}_{i}||,$$

subject to 
$$\begin{cases} \mathbf{A}_{i}\mathbf{x}_{k} \leq \mathbf{b}_{i} + M \cdot \mathbf{1} \cdot (1 - c_{i,k}), & i = 1, ..., N \\ \sum_{i=1}^{N} c_{i,k} = 1 \\ c \in \{0, 1\}^{N, K} \\ k = 1, ..., K \end{cases}$$
(8)

where w is a weight, a coefficient we can tune to adjust relative importance of our primary objective (starting from the point  $\mathbf{x}_{\text{start}}$  and finishing at the point  $\mathbf{x}_{\text{goal}}$  and our secondary objective (making the footsteps evenly spaced).

Code, part 1

```
0 \mid n = 2;
  shift_1 = 1*rand(n, 1);
|| shift_2 | = 1*rand(n, 1);
 V1 = randn(n, 6);
4 | V2 = randn(n, 6) + shift_1;
 V3 = randn(n, 6) + shift_1 + shift_2;
  indices_convhull = convhull(V1');
|V1 = V1(:, indices\_convhull);
  indices_convhull = convhull(V2');
10 \mid V2 = V2(:, indices\_convhull);
  indices_convhull = convhull(V3');
12 \mid V3 = V3(:, indices\_convhull);
|A|[A1, b1] = vert2con(V1');
  [A2, b2] = vert2con(V2');
[A3, b3] = vert2con(V3');
```

Code, part 2

```
0 \mid \text{number\_of\_steps} = 7;
 start_point = sum(V1, 2) / size(V1, 2);
2 | finish_point = sum(V3, 2) / size(V3, 2);
 weight\_goal = 5;
_{4}| \text{bigM} = 15;
6 cvx_begin
      variable x(n, number_of_steps)
      binary variable c(3, number_of_steps);
      cost = 0;
      for i = 1:(number\_of\_steps - 1)
          cost = cost + norm(x(:, i) - x(:, i+1));
      end
      cost = cost + norm(x(:, 1) - start_point)*
      weight_goal;
      cost = cost + norm(x(:, number_of_steps) -
      finish_point) * weight_goal;
      minimize (cost)
```

Code, part 3

```
subject to for i = 1:number\_of\_steps A1*x(:, i) <= b1 + (1 - c(1, i))*bigM; A2*x(:, i) <= b2 + (1 - c(2, i))*bigM; A3*x(:, i) <= b3 + (1 - c(3, i))*bigM; c(1, i) + c(2, i) + c(3, i) == 1; end cvx\_end plot(x(1, :)', x(2, :)', '^', 'MarkerEdgeColor', 'k', 'MarkerSize', 10, 'LineWidth', 2); hold on;
```

### Example: switching control

Multiple variables

Consider the case when you have a control system  $\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}_1\mathbf{u}_1 + \mathbf{B}_2\mathbf{u}_2$ , but you are required to use either  $\mathbf{B}_1\mathbf{u}_1$  or  $\mathbf{B}_2\mathbf{u}_2$  at any time, not both.

This can be cast as a mixed-integer constraint:

$$\begin{cases} ||\mathbf{B}_{1}\mathbf{u}_{1}|| \leq M(1-c_{1}) \\ ||\mathbf{B}_{2}\mathbf{u}_{2}|| \leq M(1-c_{2}) \\ c_{1}+c_{2}=1 \end{cases}$$
(9)

#### Homework

Implement footstep planning for a biped, making sure every pair of steps lands in the same H-polytope. Lecture slides are available via Moodle.

 $You\ can\ help\ improve\ these\ slides\ at:$  github.com/SergeiSa/Computational-Intelligence-Slides-Spring-2022



Check Moodle for additional links, videos, textbook suggestions.