Systems of Inequalities

Goals:

Given system of inequalities of the form $Ax \leq b$

- determine if system has an integer solution
- enumerate all integer solutions

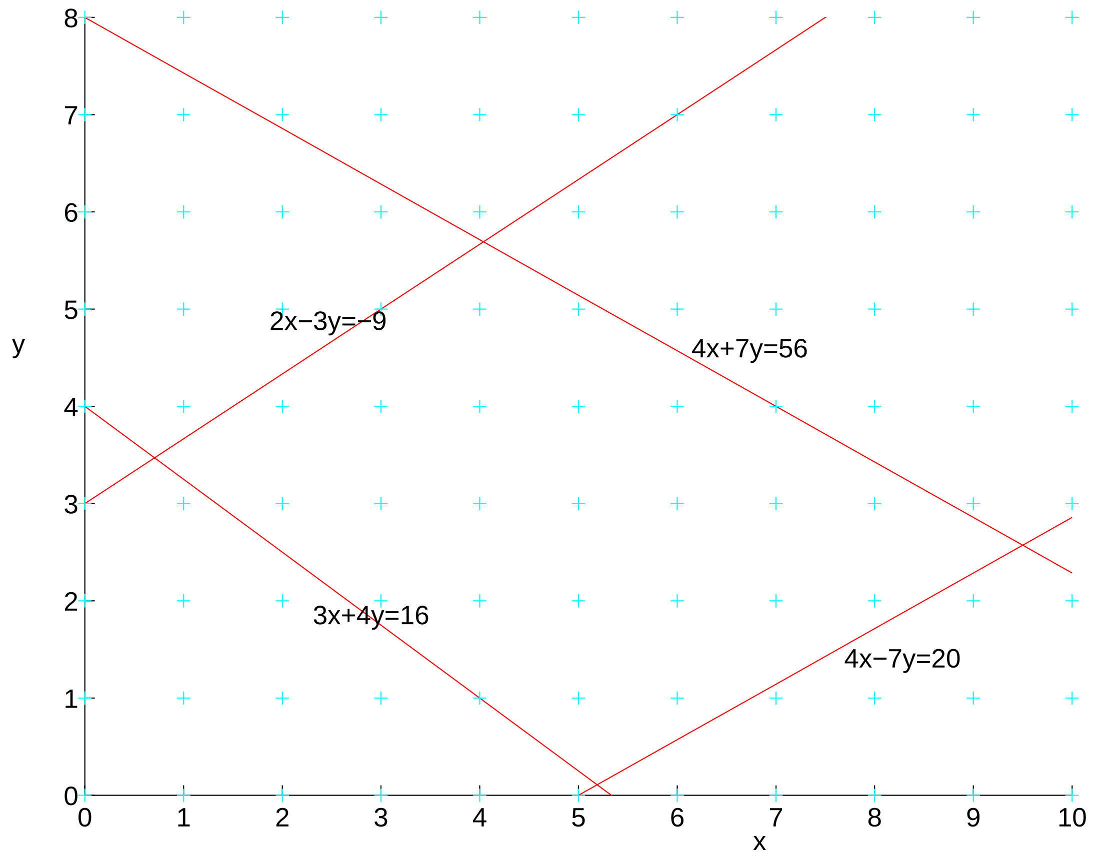
Running example:

$$3x + 4y \geq 16 \tag{1}$$
$$4x + 7y \leq 56 \tag{2}$$
$$4x - 7y \leq 20 \tag{3}$$
$$2x - 3y \geq -9 \tag{4}$$

Upper bounds for x: (2) and (3)
Lower bounds for x: (1) and (4)

Upper bounds for y: (2) and (4)
Lower bounds for y: (1) and (3)

MATLAB graphs:



Code for enumerating integer points in polyhedron: (see graph)

Outer loop: Y, Inner loop: X

DO Y=$\lceil 4/37 \rceil$, $\lfloor 74/13 \rfloor$

  DO X=$\lceil max(16/3 - 4y/3, -9/2 + 3y/2) \rceil$, $\lfloor min(5 + 7y/4, 14 - 7y/4) \rfloor$

   . . . . .

Outer loop: X, Inner loop: Y

DO X=1, 9

  DO Y=$\lceil max(4 - 3y/4, (4x - 20)/7) \rceil$, $\lfloor (min(8 - 4x/5, (2x + 9)/3) \rfloor$

   . . . . .

How do we can determine loop bounds?

Fourier-Motzkin elimination: variable elimination technique for inequalities

$$3x + 4y \geq 16 \tag{5}$$
$$4x + 7y \leq 56 \tag{6}$$
$$4x - 7y \leq 20 \tag{7}$$
$$2x - 3y \geq -9 \tag{8}$$

Let us project out x.
First, express all inequalities as upper or lower bounds on x.

$$x \geq 16/3 - 4y/3 \tag{9}$$
$$x \leq 14 - 7y/4 \tag{10}$$
$$x \leq 5 + 7y/4 \tag{11}$$
$$x \geq -9/2 + 3y/2 \tag{12}$$

For any y, if there is an x that satisfies all inequalities, then every lower bound on x must be less than or equal to every upper bound on x.

Generate a new system of inequalities from each pair (upper,lower) bounds.

$$\begin{aligned}
5 + 7y/4 &\geq 16/3 - 4y/3 (\text{Inequalities} 3, 1) \\
5 + 7y/4 &\geq -9/2 + 3y/2 (\text{Inequalities} 3, 4) \\
14 - 7y/4 &\geq 16/3 - 4y/3 (\text{Inequalities} 2, 1) \\
14 - 7y/4 &\geq -9/2 + 3y/2 (\text{Inequalities} 2, 4)
\end{aligned}$$

Simplify:

$$
\begin{aligned}
y &\geq 4/37 \\
y &\geq -38 \\
y &\leq 104/5 \\
y &\leq 74/13
\end{aligned}
$$

=>

$$
\begin{aligned}
max(4/37, -38) &\leq y \leq min(104/5, 74/13) \\
&=> \\
4/37 \leq y &\leq 74/13
\end{aligned}
$$

This means there are rational solutions to original system of inequalities.

We can now express solutions in closed form as follows:

$$
\begin{aligned}
4/37 &\leq y \leq 4/37 \\
max(16/3 - 4y/3, -9/2 + 3y/2) &\leq x \leq min(5 + 7y/4, 14 - 7y/4)
\end{aligned}
$$

Fourier-Motzkin elimination: iterative algorithm

Iterative step:

- obtain reduced system by projecting out a variable
- if reduced system has a rational solution, so does the original

Termination: no variables left

Projection along variable x: Divide inequalities into three categories

$$a_1 * y + a_2 * z + ... \leq c_1 (no\ x)$$
$$b_1 * x \leq c_2 + b_2 * y + b_3 * z + ...(upper\ bound)$$
$$d_1 * x \geq c_3 + d_2 * y + d_3 * z + ...(lower\ bound)$$

New system of inequalities:

- All inequalities that do not involve x
- Each pair (lower,upper) bounds gives rise to one inequality:

$$b_1[c_3 + d_2 * y + d_3 * z + ...] \leq d_1[c_2 + b_2 * y + b_3 * z + ...]$$

Theorem: If $(y_1, z_1, ...)$ satisfies the reduced system, then $(x_1, y_1, z_1 ...)$ satisfies the original system, where x_1 is a rational number between

$min(1/b_1(c_2 + b_2 y_1 + b_3 z_1 + ...), ...)$ (over all upper bounds)

and

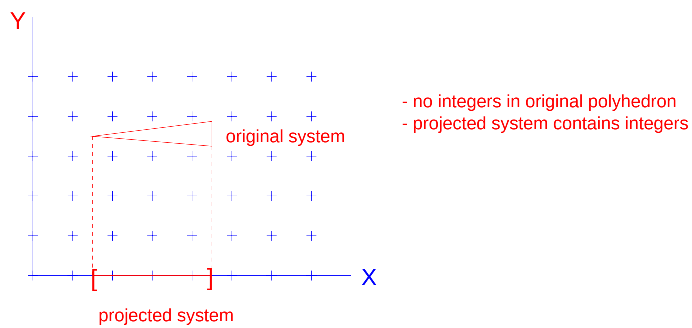
$max(1/d_1(c_3 + d_2 y_1 + d_3 z_1 + ...), ...)$ (over all lower bounds)

Proof: trivial

What can we conclude about integer solutions?

Corollary: If reduced system has no integer solutions, neither does the original system.

Not true: Reduced system has integer solutions => original system does too.



Key problem: Multiplying one inequality by b_1 and other by d_1 is not guaranteed to preserve "integrality" (cf. equalities)

Exact projection: If all upper bound coefficients b_i or all lower bound coefficients d_i happen to be 1, then integer solution to reduced system implies integer solution to original system.

Theorem: If $(y_1, z_1, ...)$ is an integer vector that satisfies the reduced system in FM elimination, then $(x_1, y_1, z_1 ...)$ satisfies the original system if there exists an integer x_1 between

$\lceil max(1/d_1(c_3 + d_2 y_1 + d_3 z_1 + ...), ...) \rceil$ (over all lower bounds)

and

$\lfloor min(1/b_1(c_2 + b_2 y_1 + b_3 z_1 + ...), ...) \rfloor$ (over all upper bounds).

Proof: trivial

Enumeration: Given a system $Ax \leq b$, we can use Fourier-Motzkin elimination to generate a loop nest to enumerate all integer points that satisfy system as follows:

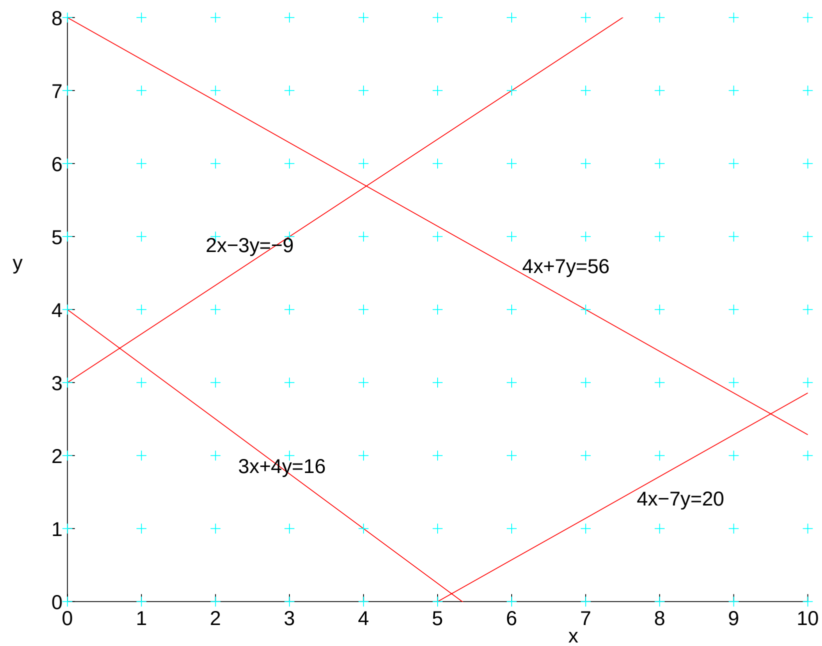
- pick an order to eliminate variables (this will be the order of variables from innermost loop to outermost loop)
- eliminate variables in that order to generate upper and lower bounds for loops as shown in theorem in previous slide

Remark: if polyhedron has no integer points, then the lower bound of some loop in the loop nest will be bigger than the upper bound of that loop

Existence: Given a system $Ax \leq b$, we can use Fourier-Motzkin elimination to project down to a single variable.

- If the reduced system has no integer solutions, then original system has no integer solutions either.
- If the reduced system has integer solutions and all projections were exact, then original system has integer solutions too.
- If reduced system has integer solutions and some projections were no exact, be conservative and assume that original system has integer solutions.

More accurate algorithm for determining existence



Just because there are integers between 4/37 and 74/13, we cannot assume there are integers in feasible region.

However, if gap between lower and upper bounds is greater than or equal to 1 for some integer value of y, there must be an integer in feasible region.

Dark shadow: region of y for which gap between upper and lower bounds of x is guaranteed to be greater than or equal to 1.

Determining dark shadow region:

Ordinary FM elimination:

$x \leq u$, $x \geq l \Rightarrow u \geq l$

Dark shadow:

$x \leq u$, $x \geq l \Rightarrow u \geq l + 1$

For our example, dark shadow projection along x gives system

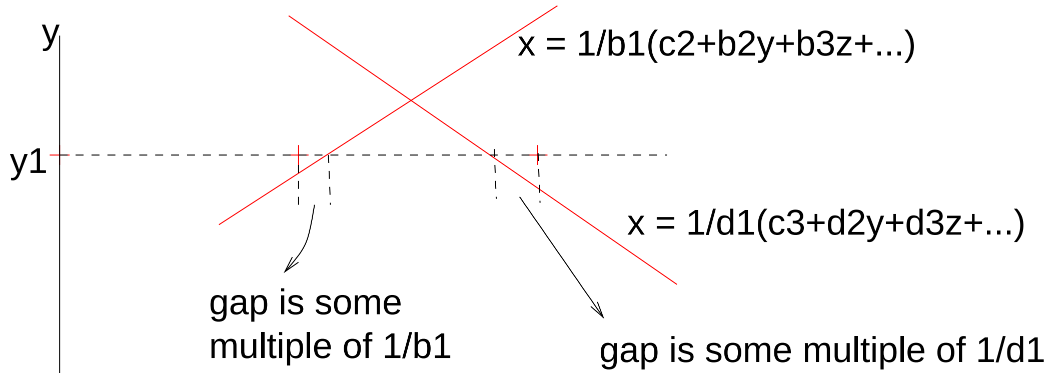
$$
\begin{aligned}
5 + 7y/4 &\geq 16/3 - 4y/3 + 1 (\text{Inequalities} 3, 1) \\
5 + 7y/4 &\geq -9/2 + 3y/2 + 1 (\text{Inequalities} 3, 4) \\
14 - 7y/4 &\geq 16/3 - 4y/3 + 1 (\text{Inequalities} 2, 1) \\
14 - 7y/4 &\geq -9/2 + 3y/2 + 1 (\text{Inequalities} 2, 4)
\end{aligned}
$$

=>

$66/13 \geq$ y $\geq 16/37$

There is an integer value of y in this range => integer in polyhedron.

More accurate estimate of dark shadow



For integer values of y1,z1,...., there is no integer value x1 between lower and upper bounds if

$$1/d1(c3+d2y1+d3z1+...) - 1/b1(c2+b2y1+b3z1+...) +1/b1+1/d1 <= 1$$

This means there is an integer between upper and lower bounds if

$$1/d1(c3+d2y1+d3z1+...) - 1/b1(c2+b2y1+b3z1+...) +1/b1+1/d1 > 1$$

To convert this to >=, notice that smallest change of lhs value is 1/b1d1.

So the inequality is

$$1/d1(c3+d2y1+d3z1+...) - 1/b1(c2+b2y1+b3z1+...) +1/b1+1/d1 >= 1 + 1/b1d1$$

=>

$$1/d1(c3+d2y1+d3z1+...) - 1/b1(c2+b2y1+b3z1+...) >= (1 - 1/b1)(1 - 1/d1)$$

Note: If $(b_1 = 1)$ or $(d_1 = 1)$, dark shadow constraint = real shadow constraint

Example:

$$3x \geq 16 - 4y$$
$$4x \leq 20 + 7y$$

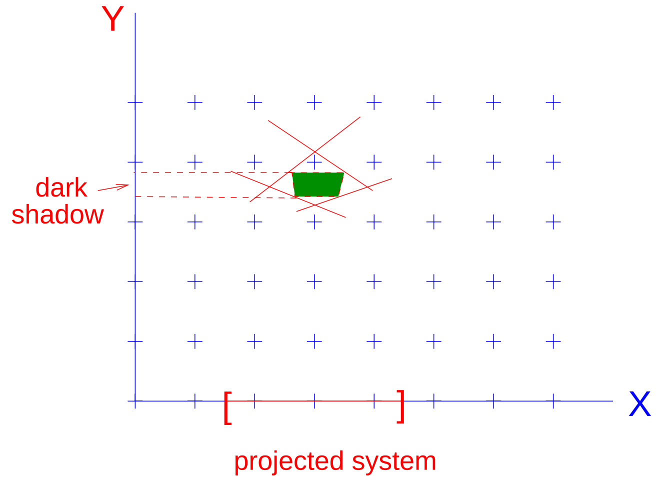
Real shadow: $(20 + 7y) * 3 \geq 4(16 - 4y)$

Dark shadow: $(20 + 7y) * 3 - 4(16 - 4y) \geq 12$

Dark shadow (improved): $(20 + 7y) * 3 - 4(16 - 4y) \geq 6$

What if dark shadow has no integers?

There may still be integer points nestled closely between an upper and lower bound.

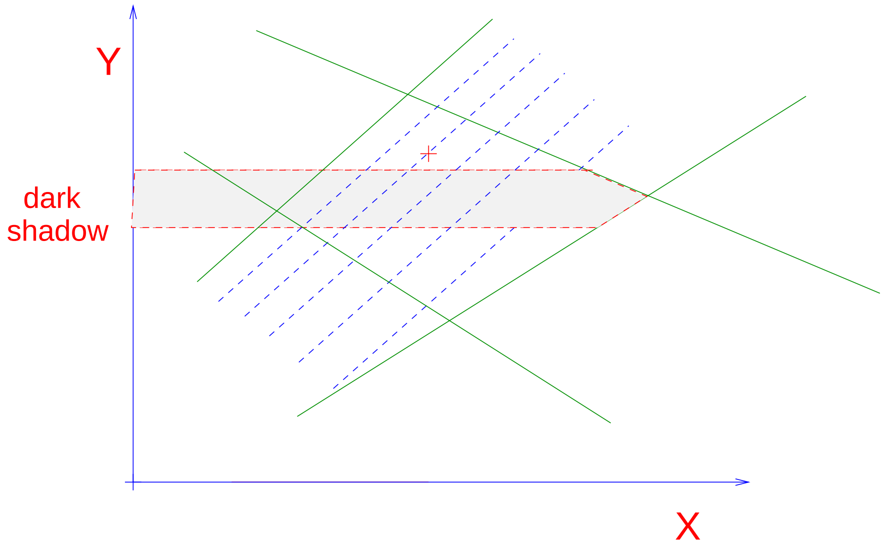


Conservative approach:

- if dark shadow has integer points, deduce correctly that original system has integer solutions
- if dark shadow has no integer points, declare conservatively that original system may have integer solutions

Another alternative: if dark shadow has no integer points, try enumeration

One enumeration idea: splintering



Scan the corners with hyperplanes, looking for integer points.

Generate a succession of problems in which each lower bound is replaced with a sequence of hyperplanes. How many hyperplanes are needed?

Equation for lower bound: x = 1/b1(c2+b2y+b3z+....)

Hyperplanes:

x = 1/b1(c2+b2y+b3z+....)

x = 1/b1(c2+b2y+b3z+....)+ 1/b1

x = 1/b1(c2+b2y+b3z+....)+ 2/b1

x = 1/b1(c2+b2y+b3z+....)+ 3/b1

....

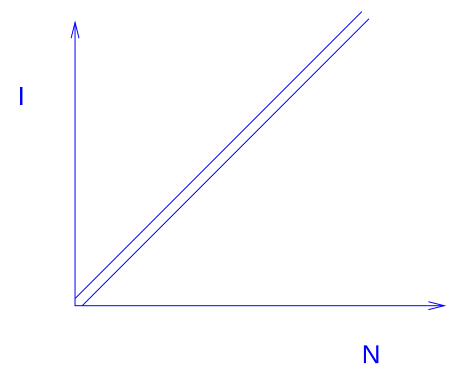
x = 1/b1(c2+b2y+b3z+....)+ 1 (in dark shadow region; if this is integer, so is)

Something to think about for assignment:

Can we just generate loop bounds using Fourier-Motzkin elimination and execute that loop to find integer points?

Issues:

1. What if there are variables in the system of inequalities which are not bounded (like an unknown upper bound 'N' in a loop in source code)? Region of interest is unbounded, so what does enumeration mean?



I >= 0
N >= 0
N <= I + 1/3
I <= N - 1/3

real shadow is infinite
dark shadow is empty

=> loop bounds are infinite :-(

2. If this idea can be made to work, is it as efficient as splintering?

Engineering

- Use matrices and vectors to represent inequalities.

$$\begin{pmatrix} -3 & -4 \\ 4 & 7 \\ 4 & -7 \\ -2 & 3 \end{pmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \leq \begin{bmatrix} -16 \\ 56 \\ 20 \\ 9 \end{bmatrix}$$

  - lower bounds and upper bounds for a variable can be determined by inspecting signs of entries in column for that variable
  - easy to tell if exact projection is being carried out

Engineering (cont.)

- Use matrices and vectors ...
    - Fourier-Motzkin elimination is carried out by row operations on pairs of lower and upper bounds. For example, eliminating x:

$$\begin{pmatrix} 0 & 5 \\ 0 & -37 \\ 0 & 13 \\ 0 & -1 \end{pmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \leq \begin{bmatrix} 104 \\ -4 \\ 74 \\ 38 \end{bmatrix}$$

- Dark shadow and real shadow computations should be carried out simultaneously to share work (only vector on rhs is different)

Engineering (cont.)

- Handle equalities first to reduce number of equations. Find (parameterized) solution to equalities and substitute solution into inequalities.
- Keep co-efficients small by dividing an inequality by gcd of co-efficients if gcd is not 1.
- Check for redundant and contradictory constraints.
- Do exact projections wherever possible.

Engineering (cont.)

- Eliminate equations with semi-constrained variables (no upper or no lower bound).

```
DO 10 I = 1, N
   X(I) = ...X(I-1)...
```

```
Flow dependence:
Iw = Ir - 1
1 <= Iw <= Ir <= N
```

`N` only has an lower bound (`N >= Ir`) which can always be satisfied given any values of `(Ir,Iw)`. So eliminate the constraint from consideration.