Announcements

Recitation change form closes tonight!

Assignments:

- P2: Optimization
  - Due Thurs 2/23, 10pm
- HW4 (online)
  - Covers LP, IP
  - Due Tues 2/14, 10 pm (Happy Valentine's Day)

EXAM 1 2/16!!

Plan

Last Time

- Linear programming formulation
  - Problem description
  - Graphical representation
  - Optimization representation

Today

- Solving linear programs
- Higher dimensions than just 2
- Integer programs

From last time…

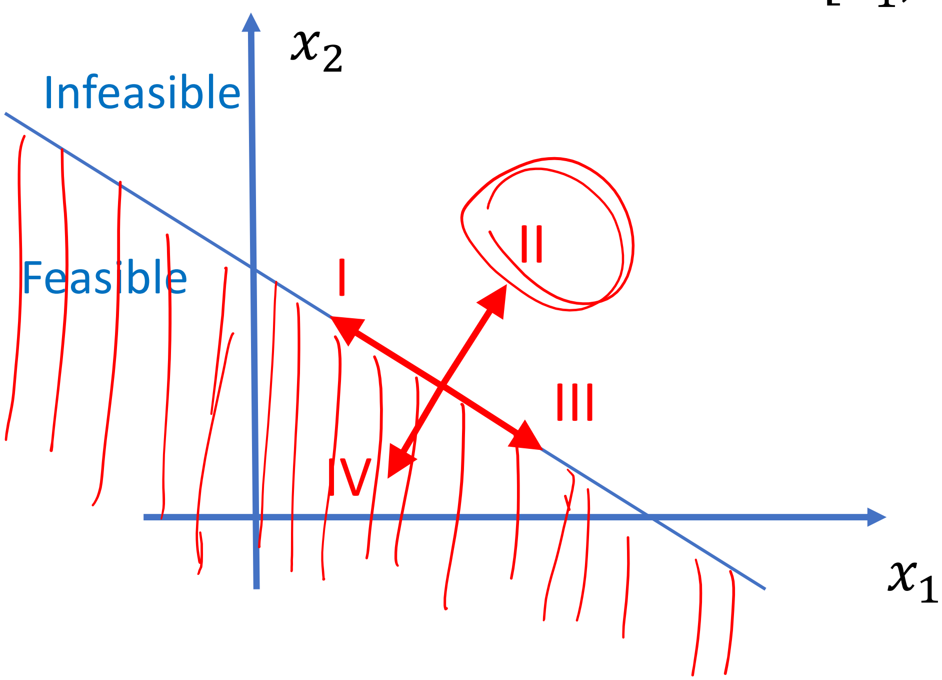
Poll 4 (already completed)

What is the relationship between the half plane:

$$a_1\, x_1 + a_2\, x_2 \leq b_1$$

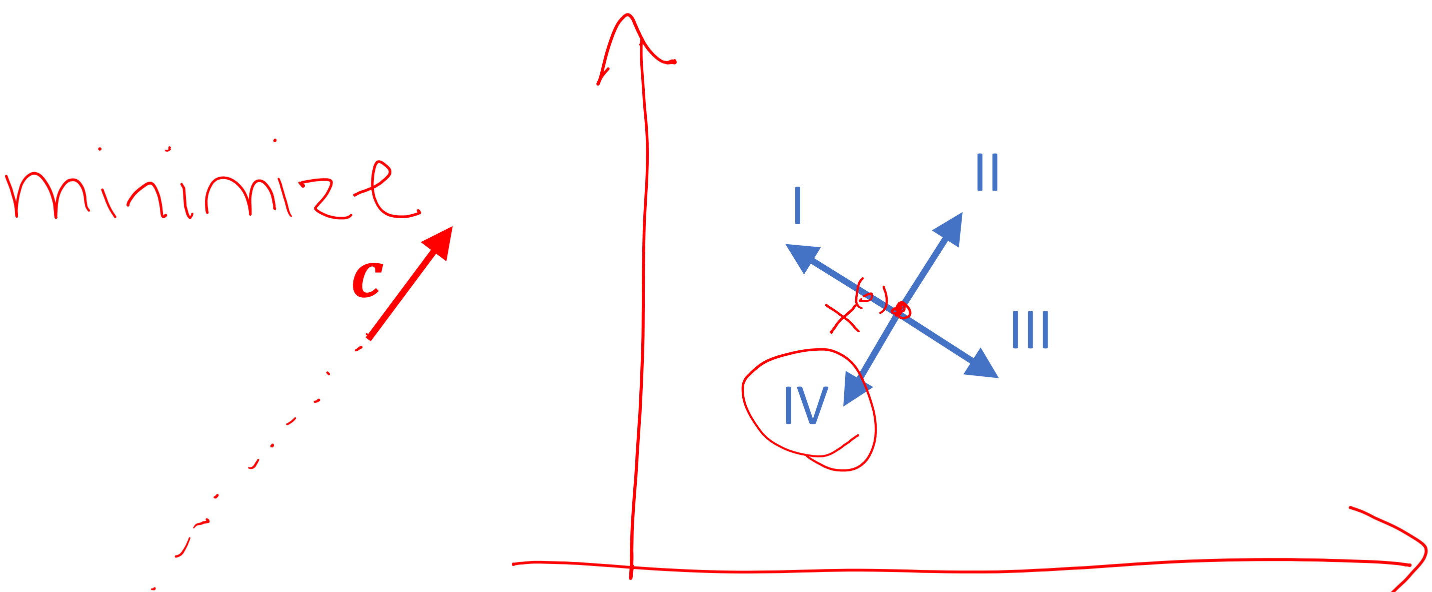
and the vector:

$$[a_1, a_2]^T$$



Question

Given the cost vector $[c_1, c_2]^T$ and initial point $\boldsymbol{x}^{(0)}$,
Which unit vector step $\triangle\ \boldsymbol{x}$ will cause $\boldsymbol{x}^{(1)} = \boldsymbol{x}^{(0)} + \triangle\ \boldsymbol{x}$
to have the lowest cost $\boldsymbol{c}^T \boldsymbol{x}^{(1)}$?



Notation Alert!

Cost Contours

Given the cost vector $[c_1, c_2]^T$ where will

$\boldsymbol{c}^T\boldsymbol{x}$ = 0 ?

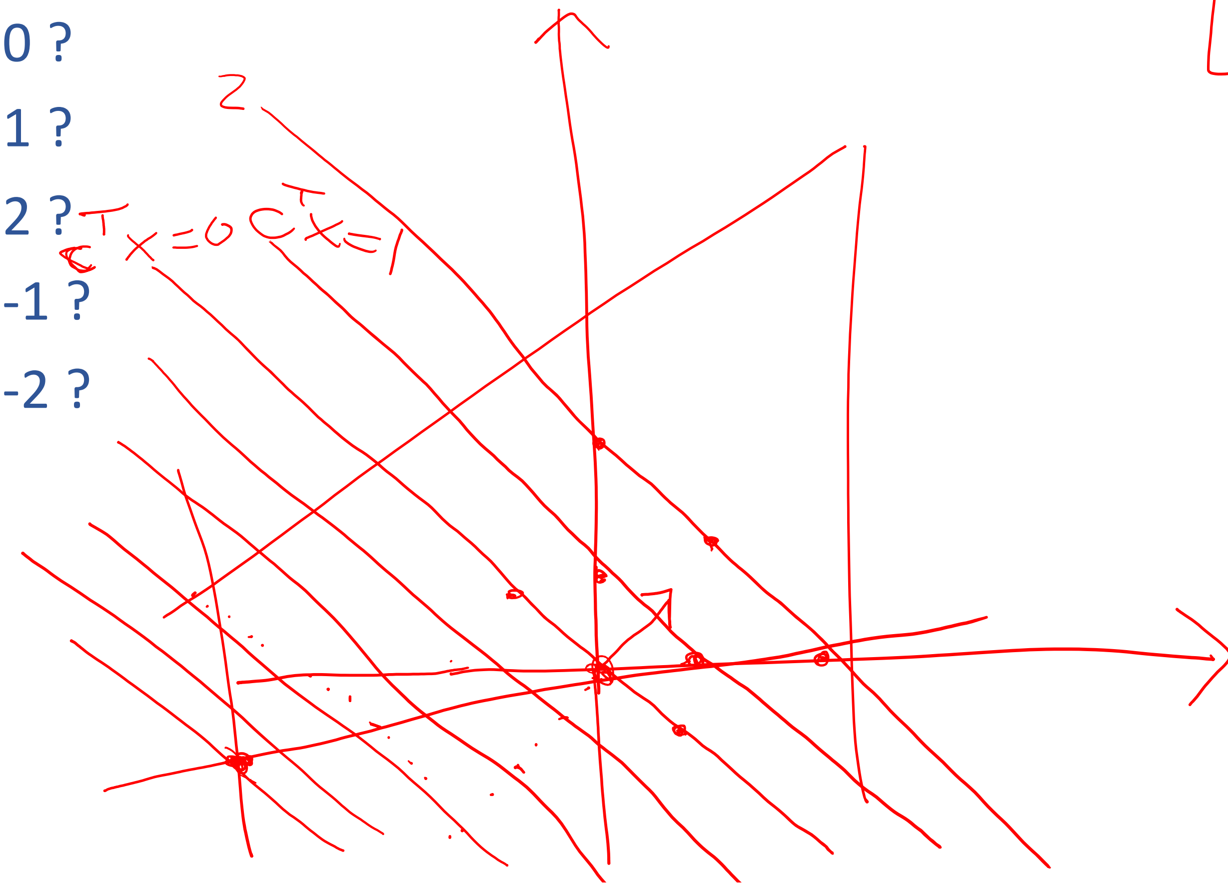
$\boldsymbol{c}^T\boldsymbol{x}$ = 1 ?

$\boldsymbol{c}^T\boldsymbol{x}$ = 2 ?

$\boldsymbol{c}^T\boldsymbol{x}$ = -1 ?

$\boldsymbol{c}^T\boldsymbol{x}$ = -2 ?

$[1, 1]\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$

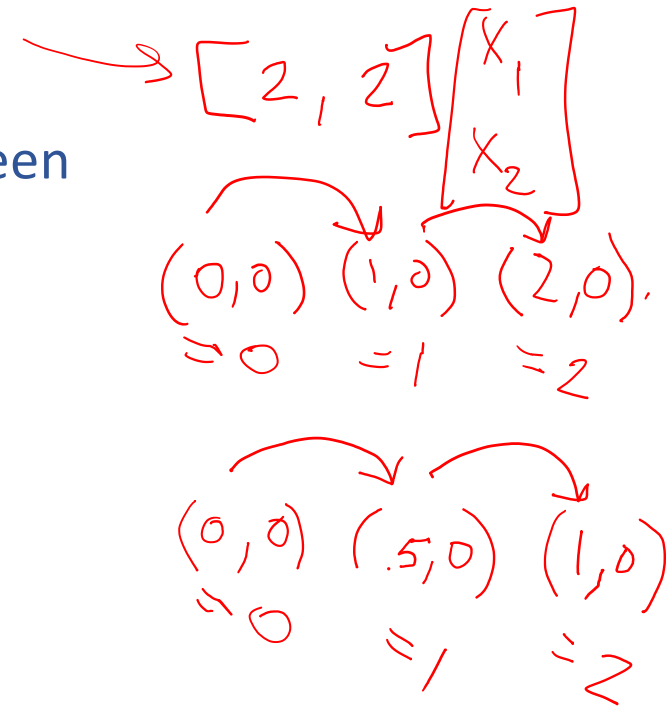


Question

As the magnitude of $\boldsymbol{c}$ increases, the distance between the contours lines of the objective $\boldsymbol{c}^T\boldsymbol{x}$:

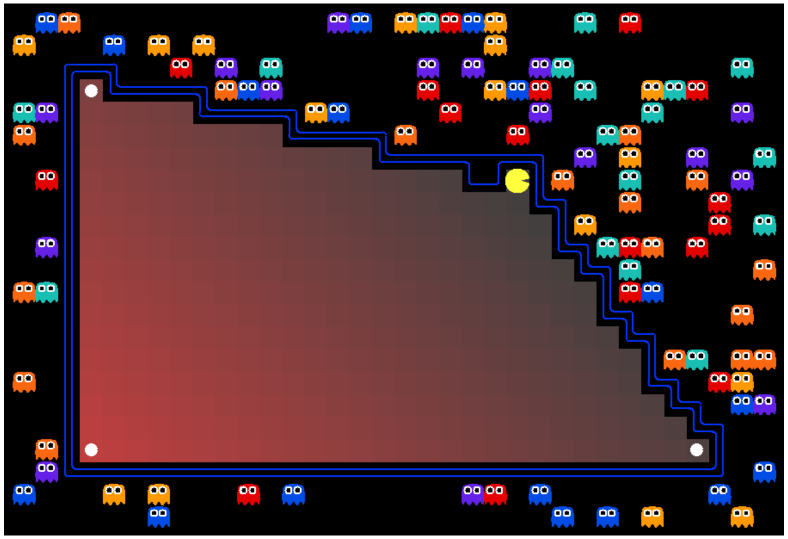
A) Increases

B) Decreases



AI: Representation and Problem Solving

Integer Programming



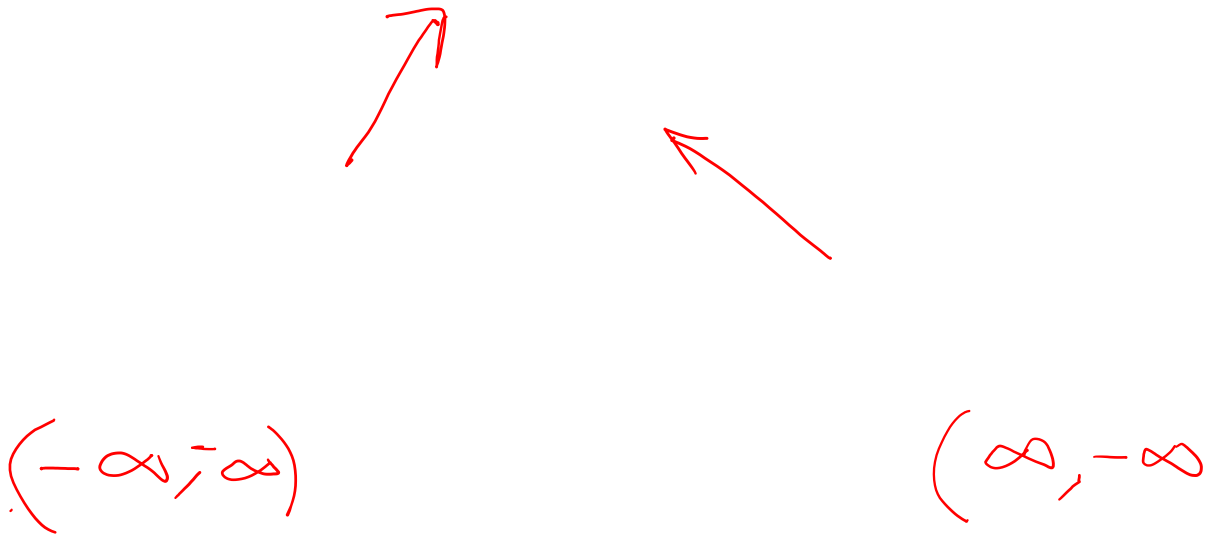
Instructor: Stephanie Rosenthal

Slide credits: CMU AI with drawings from http://ai.berkeley.edu

Solving a Linear Program

Inequality form, with no constraints

$$\min_{\boldsymbol{x}}. \quad \boldsymbol{c}^T \boldsymbol{x}$$



Solving a Linear Program

Inequality form, with one constraint

$$\begin{aligned} \min_{\boldsymbol{x}}. \quad & \boldsymbol{c}^T \boldsymbol{x} \\ \text{s.t.} \quad & a_1 x_1 + a_2 x_2 \leq b \end{aligned}$$

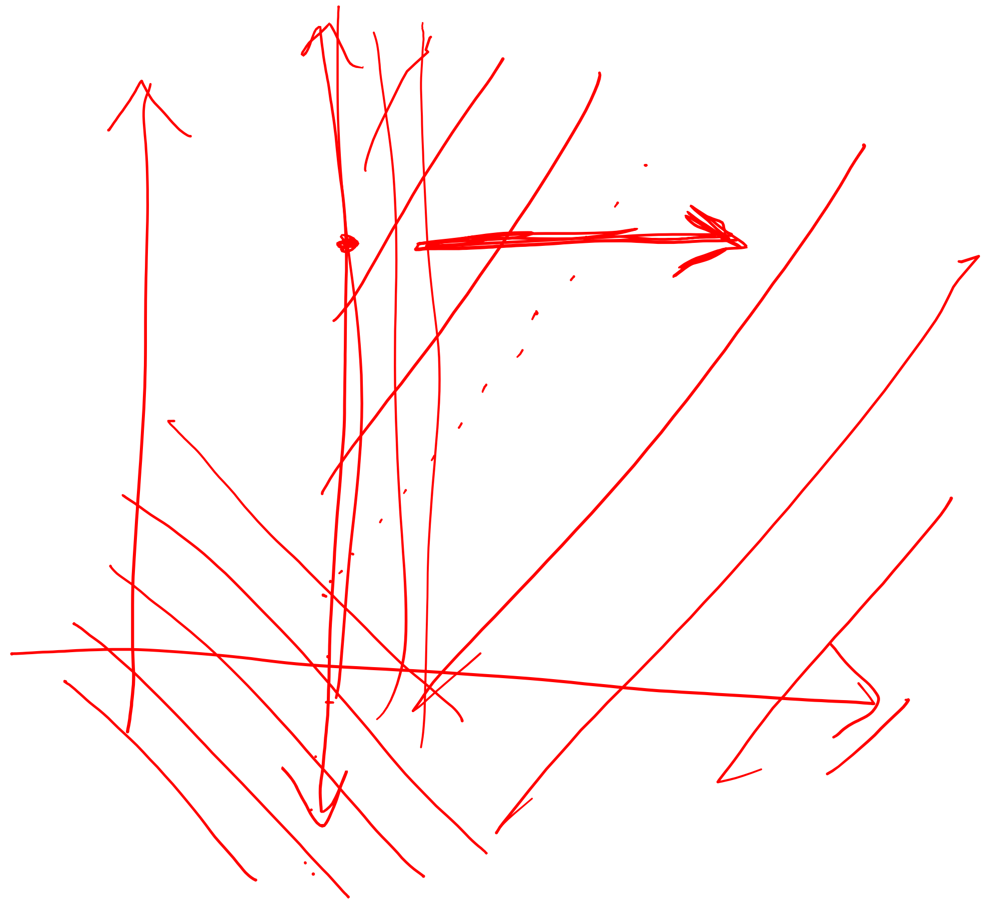
Poll 1

True or False: A minimizing LP with exactly one constraint, will always have a minimum objective at $-\infty$.

$= c^T x$

$$\min_{\boldsymbol{x}} \quad \boldsymbol{c}^T \boldsymbol{x}$$
$$\text{s.t.} \quad a_1 x_1 + a_2 x_2 \le b$$

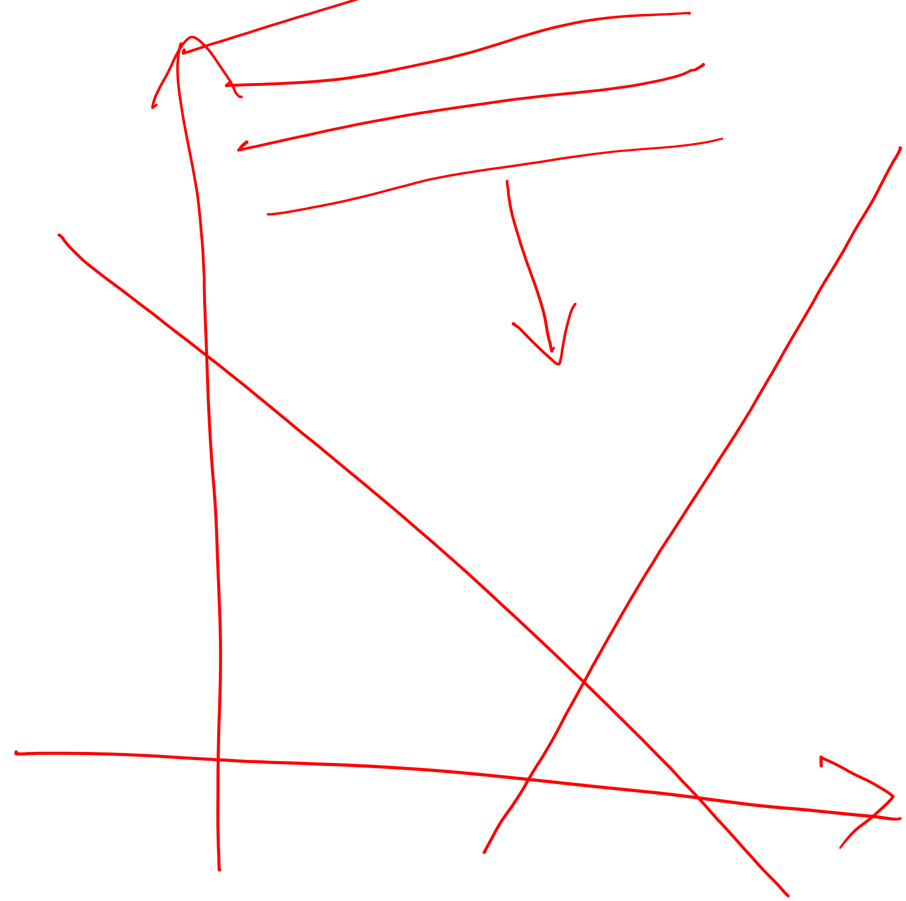
False



Question

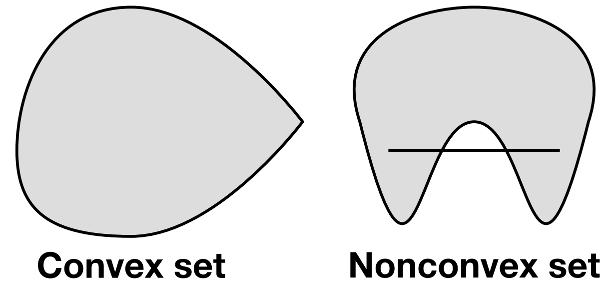
True or False: A minimizing LP with exactly two constraints, will always have a minimum objective $> -\infty$.

$$\min_{\boldsymbol{x}} \quad \boldsymbol{c}^T \boldsymbol{x}$$
$$\text{s.t.} \quad a_{11}x_1 + a_{12}x_2 \leq b_1$$
$$a_{21}x_1 + a_{22}x_2 \leq b_2$$

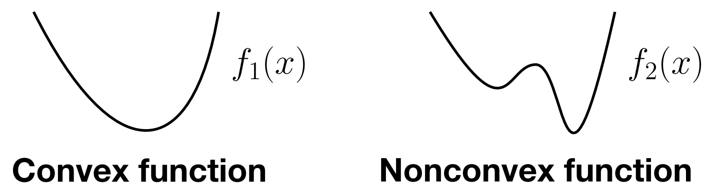


Convexity

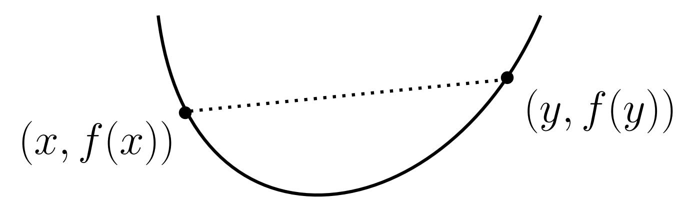
Convex sets are those in which you can draw a line between two points and all the points between them are also in the set



Convex optimization problems are ones in which the local minimum is also the global minimum



Convex functions have the property that for any point between two points x and y in a convex set:
$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta)f(y)$



Linear functions (like our costs) are convex!

Convexity and LPs

LPs are constrained convex problems.

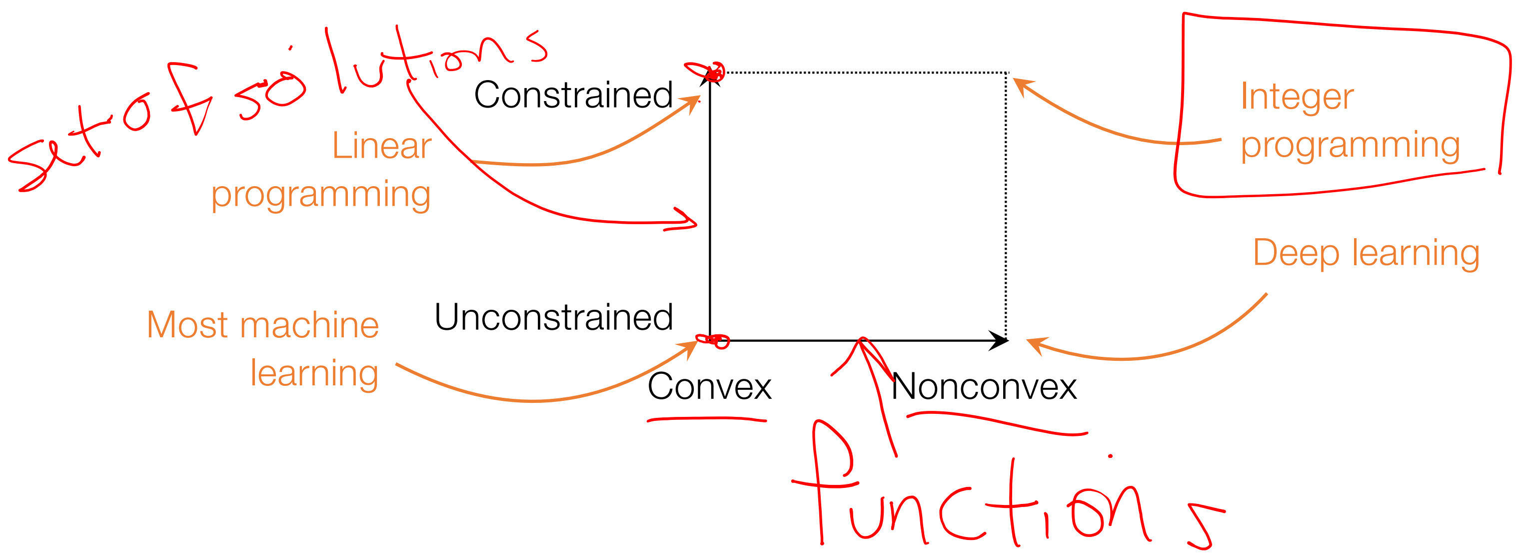
The constraints form a convex set

The objective function is convex

What does this tell us about the costs at the corners of a constrained polygon?

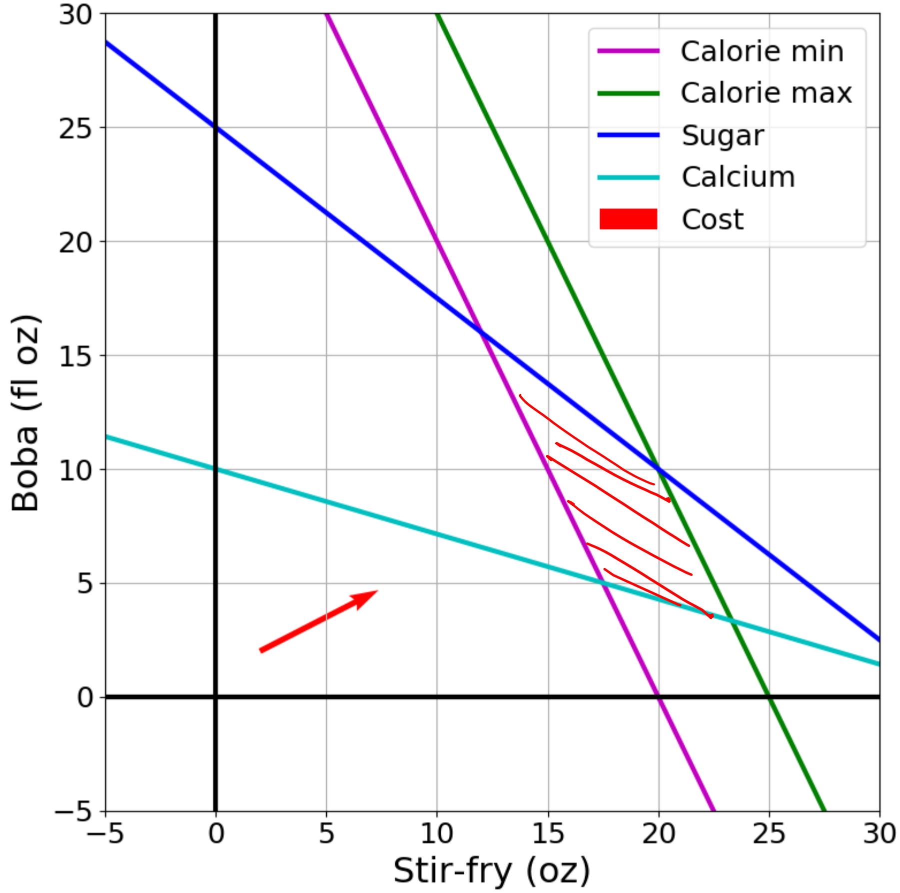
exactly the solutions

Bigger Picture



Convexity and LP Solutions

Solutions are at feasible intersections of constraint boundaries!!



Solving an LP

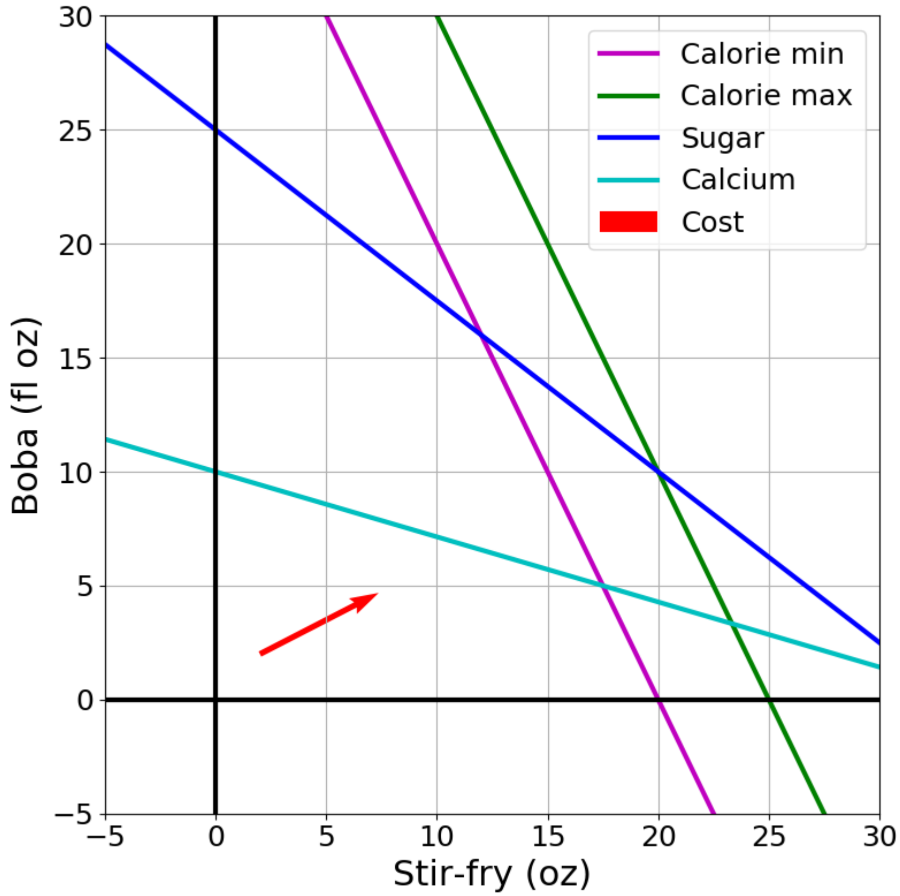
Solutions are at feasible intersections of constraint boundaries!!

Algorithm

- Check objective at all feasible intersections

In more detail:

1. Enumerate all intersections
2. Keep only those that are feasible (satisfy *all* inequalities)
3. Return feasible intersection with the lowest objective value



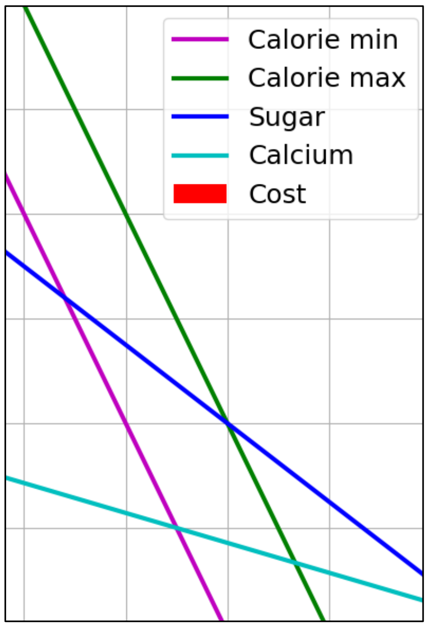
Solving an LP

But, how do we find the intersection between boundaries?

$$\min_{\boldsymbol{x}} \quad \boldsymbol{c}^T \boldsymbol{x}$$
$$\text{s.t.} \quad A\boldsymbol{x} \preceq \boldsymbol{b}$$

$$A = \begin{bmatrix} -100 & -50 \\ 100 & 50 \\ 3 & 4 \\ -20 & -70 \end{bmatrix} \qquad \boldsymbol{b} = \begin{bmatrix} -2000 \\ 2500 \\ 100 \\ -700 \end{bmatrix}$$

Calorie min
Calorie max
Sugar
Calcium



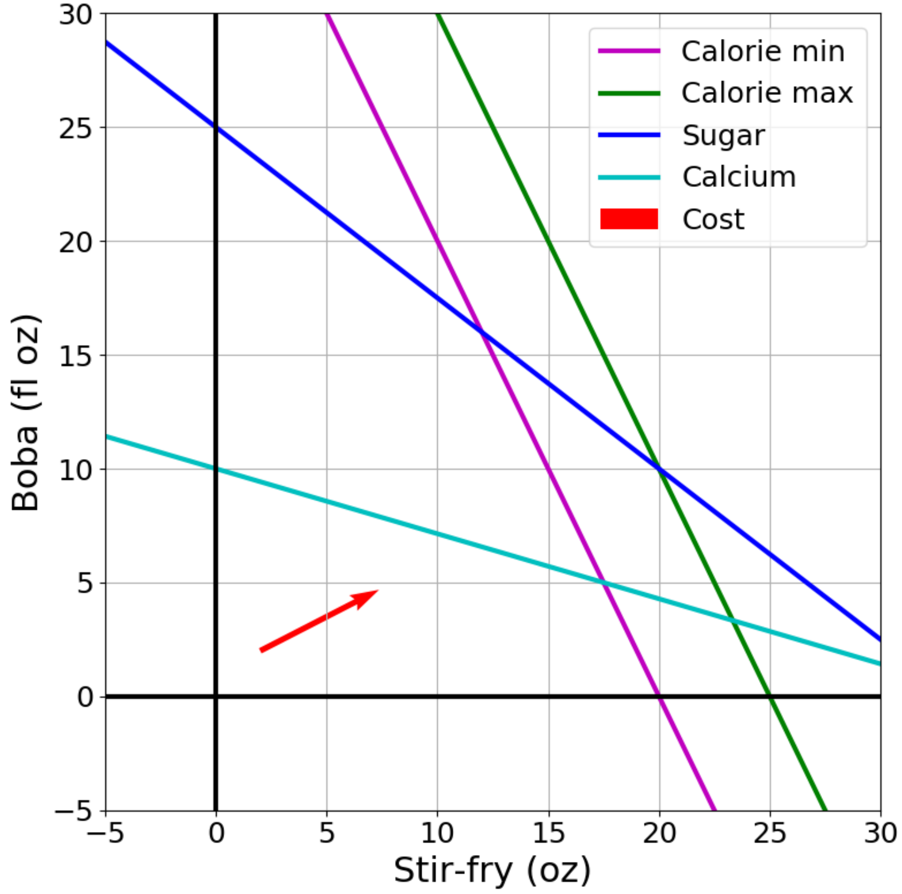
all pairs of constraints

Solving an LP

Solutions are at feasible intersections of constraint boundaries!!

Algorithms

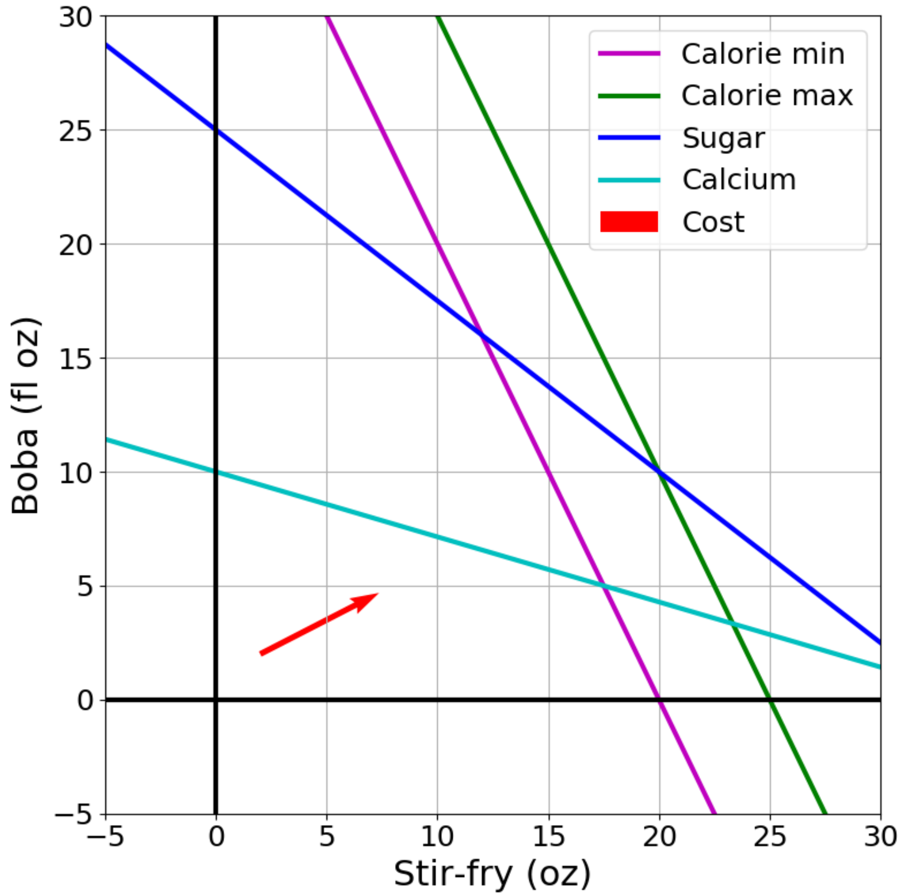
- Check objective at all feasible intersections
- Simplex hill climbing



Solving an LP

Simplex algorithm

- Start at a feasible intersection (if not trivial, can solve another LP to find one)
- Define successors as “neighbors” of current intersection
  - i.e., remove one row from our square subset of A, and add another row not in the subset; then check feasibility
- Move to any successor with lower objective than current intersection
  - If no such successors, we are done



Greedy local hill-climbing search! … but always finds *optimal* solution

Solving an LP

Solutions are at feasible intersections of constraint boundaries!!

Algorithms

- Check objective at all feasible intersections
- Simplex
- Interior Point

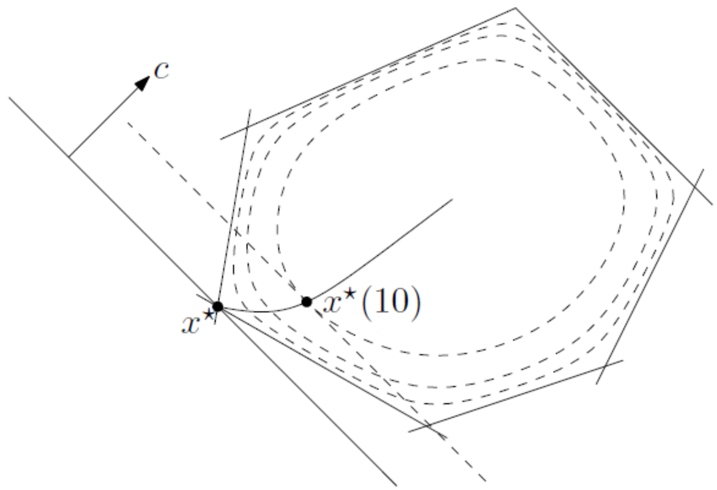


Figure 11.2 from Boyd and Vandenberghe, *Convex Optimization*

What about higher dimensions?

Problem Description

m constraints

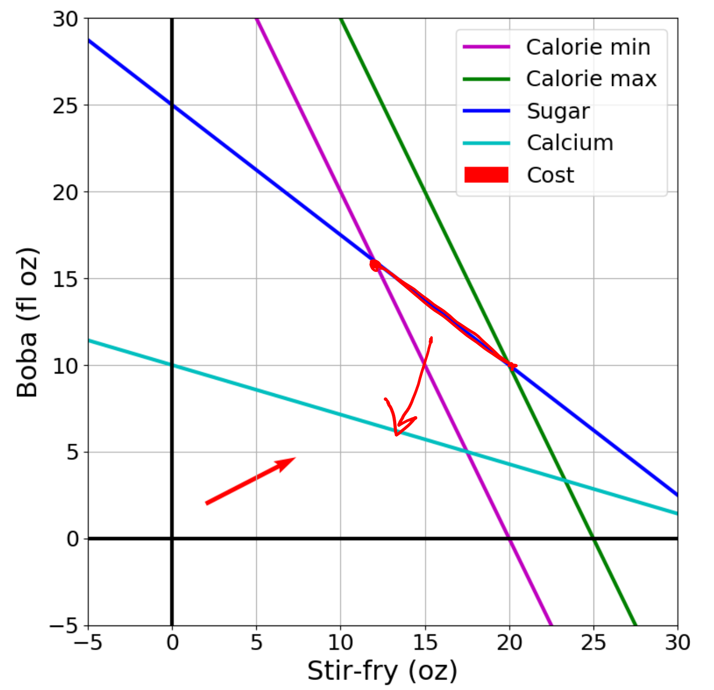
n dimensional ✗

$\binom{m}{n}$

Optimization Representation

$$\min_{\boldsymbol{x}} \quad \boldsymbol{c}^T \boldsymbol{x}$$
$$\text{s.t.} \quad A\boldsymbol{x} \preceq \boldsymbol{b}$$

Graphical Representation



“Marty, you’re not thinking fourth-dimensionally”

https://www.youtube.com/watch?v=CUcNM7OsdsY

Shapes in higher dimensions

How do these linear shapes extend to 3-D, N-D?

$$a_1\ x_1 + a_2\ x_2 = b_1$$

$$a_1\ x_1 + a_2\ x_2 \leq b_1$$

$$a_{1,1}\ x_1 + a_{1,2}\ x_2 \leq b_1$$
$$a_{2,1}\ x_1 + a_{2,2}\ x_2 \leq b_2$$
$$a_{3,1}\ x_1 + a_{3,2}\ x_2 \leq b_3$$
$$a_{4,1}\ x_1 + a_{4,2}\ x_2 \leq b_4$$

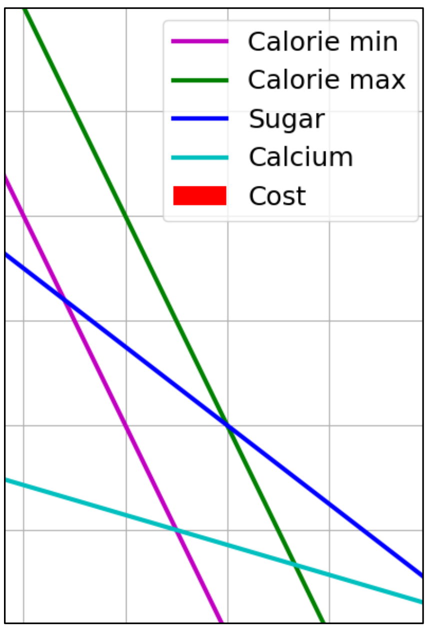
What are intersections in higher dimensions?

How do these linear shapes extend to 3-D, N-D?

$$\min_{\boldsymbol{x}} \quad \boldsymbol{c}^T \boldsymbol{x}$$
$$\text{s.t.} \quad A\boldsymbol{x} \preceq \boldsymbol{b}$$

$$A = \begin{bmatrix} -100 & -50 & \\ 100 & 50 & \\ 3 & 4 & \\ -20 & -70 & \end{bmatrix} \quad \boldsymbol{b} = \begin{bmatrix} -2000 \\ 2500 \\ 100 \\ -700 \end{bmatrix}$$

Calorie min
Calorie max
Sugar
Calcium

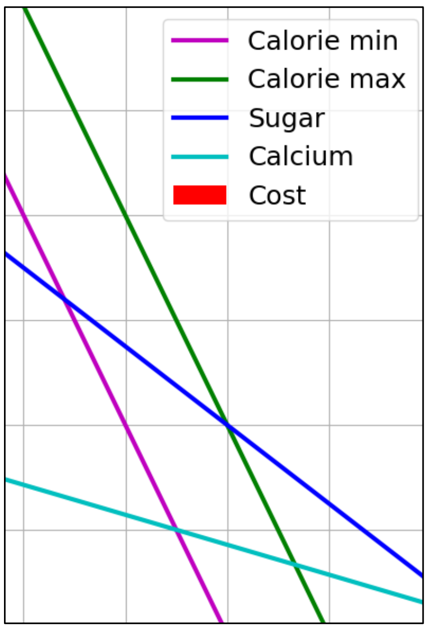


How do we find intersections in higher dimensions?

Still looking at subsets of A matrix

$$\min_{\boldsymbol{x}} \quad \boldsymbol{c}^T\boldsymbol{x}$$
$$\text{s.t.} \quad A\boldsymbol{x} \preceq \boldsymbol{b}$$

$$A = \begin{bmatrix} -100 & -50 & \\ 100 & 50 & \\ 3 & 4 & \\ -20 & -70 & \end{bmatrix} \quad \boldsymbol{b} = \begin{bmatrix} -2000 \\ 2500 \\ 100 \\ -700 \end{bmatrix} \begin{matrix} \text{Calorie min} \\ \text{Calorie max} \\ \text{Sugar} \\ \text{Calcium} \end{matrix}$$



Linear Programming

We are trying to stay healthy by finding the optimal food to purchase.

We can choose the amount of stir-fry (ounce) and boba (fluid ounces).

Healthy Squad Goals

- $2000 \leq \text{Calories} \leq 2500$
- $\text{Sugar} \leq 100$ g
- $\text{Calcium} \geq 700$ mg

Food	Cost	Calories	Sugar	Calcium
Stir-fry (per oz)	1	100	3	20
Boba (per fl oz)	0.5	50	4	70

What is the cheapest way to stay "healthy" with this menu?

How much stir-fry (ounce) and boba (fluid ounces) should we buy?

Linear Programming → Integer Programming

We are trying healthy by finding the optimal amount of food to purchase.

We can choose the amount of stir-fry (bowls) and boba (glasses).

Healthy Squad Goals

- $2000 \leq$ Calories ≤ 2500
- Sugar ≤ 100 g
- Calcium ≥ 700 mg

Food	Cost	Calories	Sugar	Calcium
Stir-fry (per bowl)	1	100	3	20
Boba (per glass)	0.5	50	4	70

What is the cheapest way to stay "healthy" with this menu?

How much stir-fry (ounce) and boba (fluid ounces) should we buy?

Linear Programming vs Integer Programming

Linear objective with linear constraints, but now with additional constraint that all values in $\boldsymbol{x}$ must be integers

$$\min_{\boldsymbol{x}} \quad \boldsymbol{c}^T \boldsymbol{x} \qquad \text{s.t.} \quad A\boldsymbol{x} \preceq \boldsymbol{b}$$

$$\min_{\boldsymbol{x}} \quad \boldsymbol{c}^T \boldsymbol{x} \qquad \text{s.t.} \quad A\boldsymbol{x} \preceq \boldsymbol{b}, \quad \boldsymbol{x} \in \mathbb{Z}^N$$

integer

We could also do:

- Even more constrained: Binary Integer Programming
- A hybrid: Mixed Integer Linear Programming

Notation Alert!

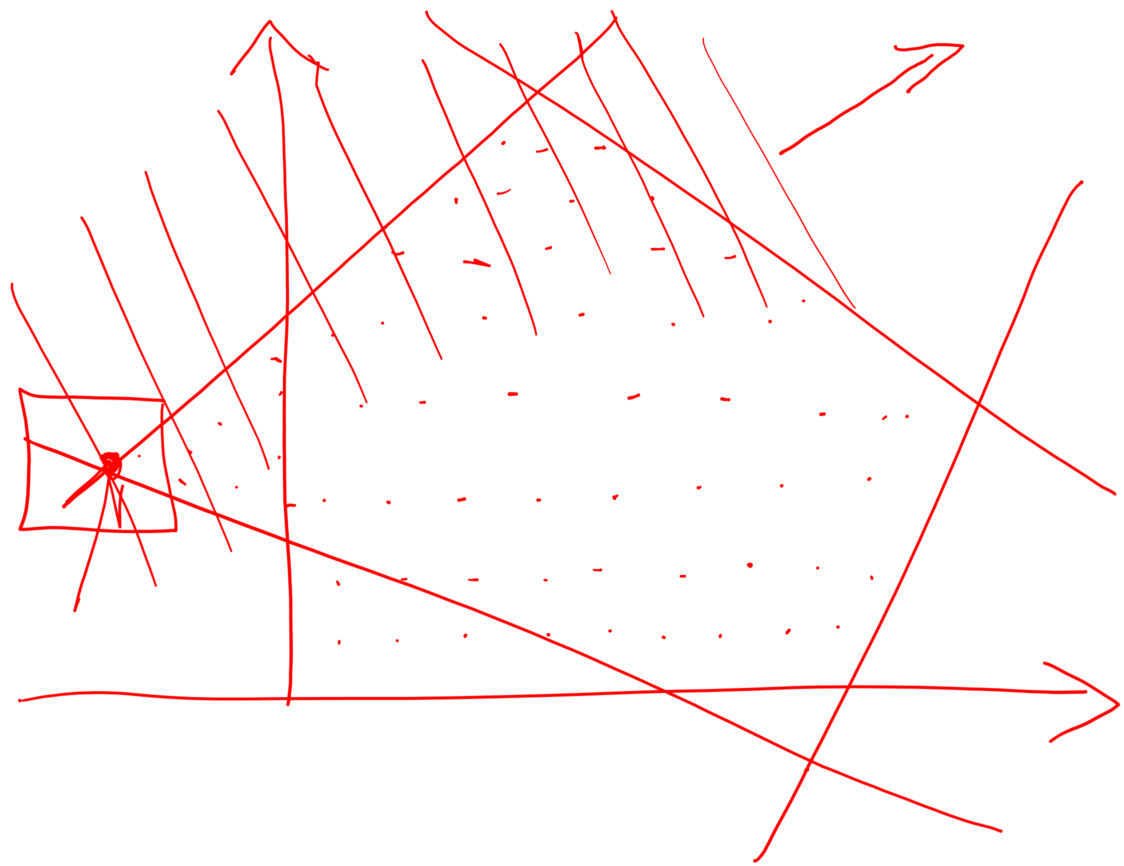
Integer Programming: Graphical Representation

Just add a grid of integer points onto our LP representation

$$\min_{\boldsymbol{x}} \quad \boldsymbol{c}^T \boldsymbol{x}$$

$$\text{s.t.} \quad A\boldsymbol{x} \preceq \boldsymbol{b}$$

$$\boldsymbol{x} \in \mathbb{Z}^N$$



Integer Programming: Scheduling

How would we formulate our CSP as an integer program?

How would we could we solve it?

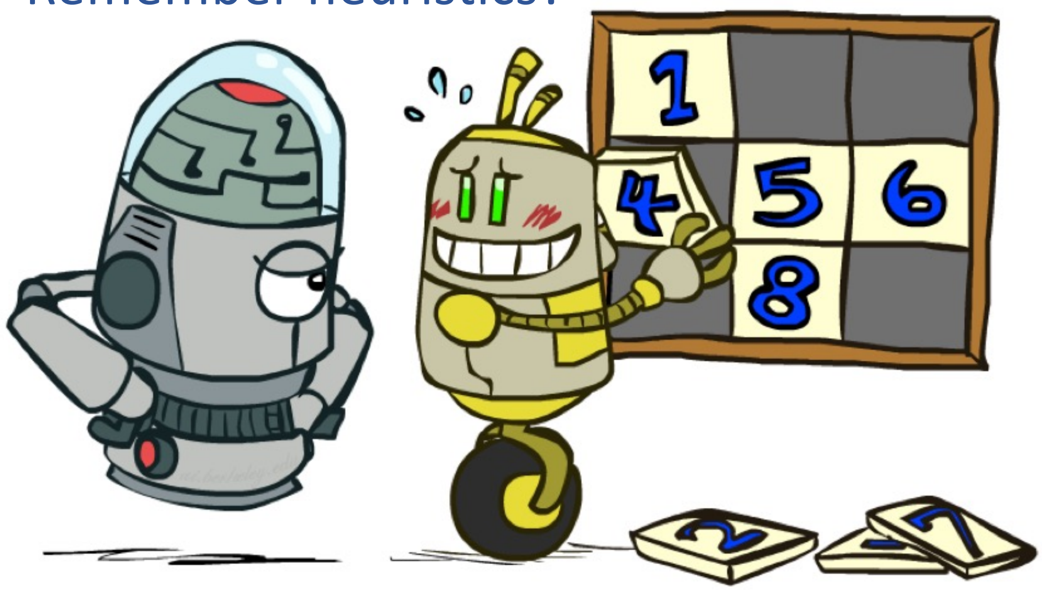
Convexity and IPs

Integer programs are not convex, but perhaps we can use the LP solvers to find solutions to integer programs?

Relax IP to LP by dropping integer constraints

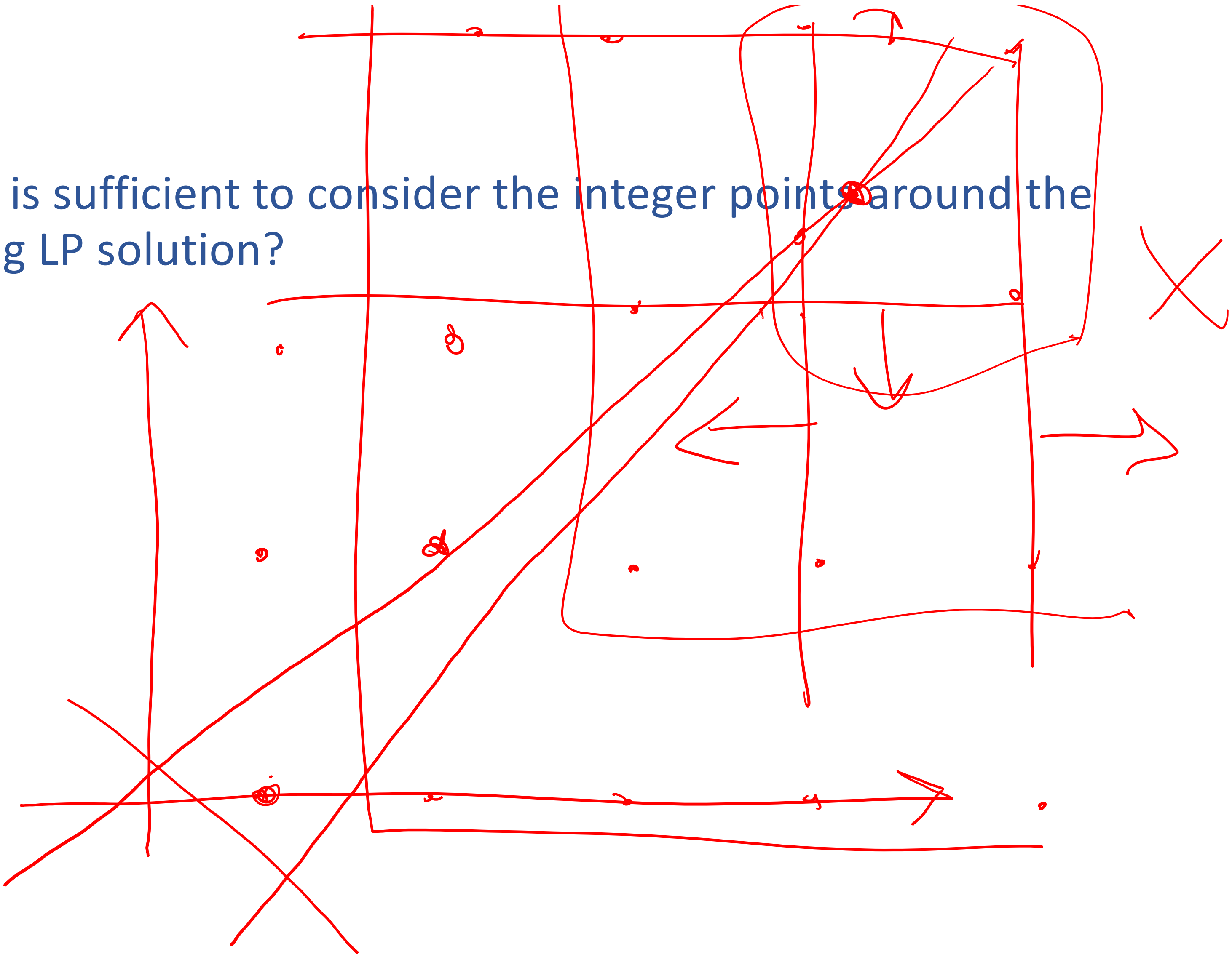
$$\begin{aligned} \min_{\boldsymbol{x}}. \quad & \boldsymbol{c}^T \boldsymbol{x} \\ \text{s.t.} \quad & A\boldsymbol{x} \leq \boldsymbol{b} \\ & \cancel{\boldsymbol{x} \in \mathbb{Z}^N} \end{aligned}$$

Remember heuristics?



Poll 2:

True/False: It is sufficient to consider the integer points around the corresponding LP solution?



Poll 3:

Let y_{IP}^* be the optimal objective of an integer program P.

Let $\boldsymbol{x}_{IP}^*$ be an optimal point of an integer program P.

Let y_{LP}^* be the optimal objective of the LP-relaxed version of P.

Let $\boldsymbol{x}_{LP}^*$ be an optimal point of the LP-relaxed version of P.

Assume that P is a minimization problem.

$$y_{IP}^* = \min_{\boldsymbol{x}} \quad \boldsymbol{c}^T\boldsymbol{x}$$
$$\text{s.t.} \quad A\boldsymbol{x} \preceq \boldsymbol{b}$$
$$\boldsymbol{x} \in \mathbb{Z}^N$$

Which of the following are true? Select all that apply.

A) $\boldsymbol{x}_{IP}^* = \boldsymbol{x}_{LP}^*$

B) $y_{IP}^* \le y_{LP}^*$

C) $y_{IP}^* \ge y_{LP}^*$

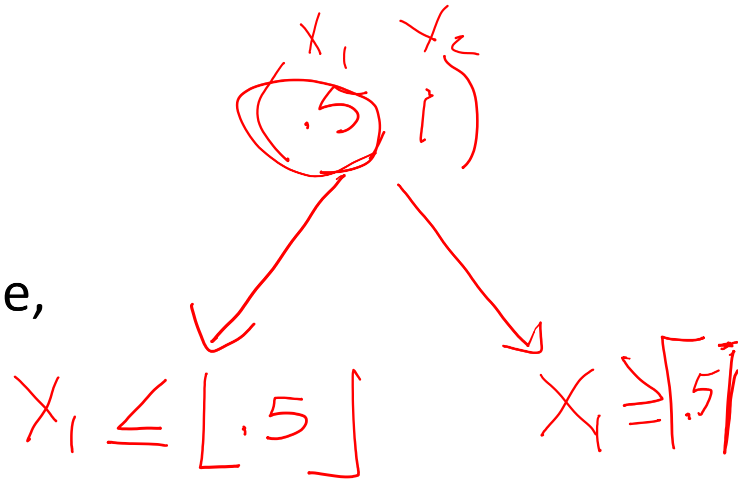
$$y_{LP}^* = \min_{\boldsymbol{x}} \quad \boldsymbol{c}^T\boldsymbol{x}$$
$$\text{s.t.} \quad A\boldsymbol{x} \preceq \boldsymbol{b}$$

Solving an IP

Branch and Bound algorithm

1. Push LP solution of problem into priority queue,

   ordered by objective value of LP solution



2. Repeat:
   - If queue is empty, return IP is infeasible
   - Pop candidate solution $\boldsymbol{x}_{LP}^{\star}$ from priority queue
   - If $\boldsymbol{x}_{LP}^{\star}$ is all integer valued, we are done; return solution
   - Otherwise, select a coordinate x_i that is not integer valued, and add two additional LPs to the priority queue:

     Left branch: Added constraint $x_i \leq floor(x_i)$

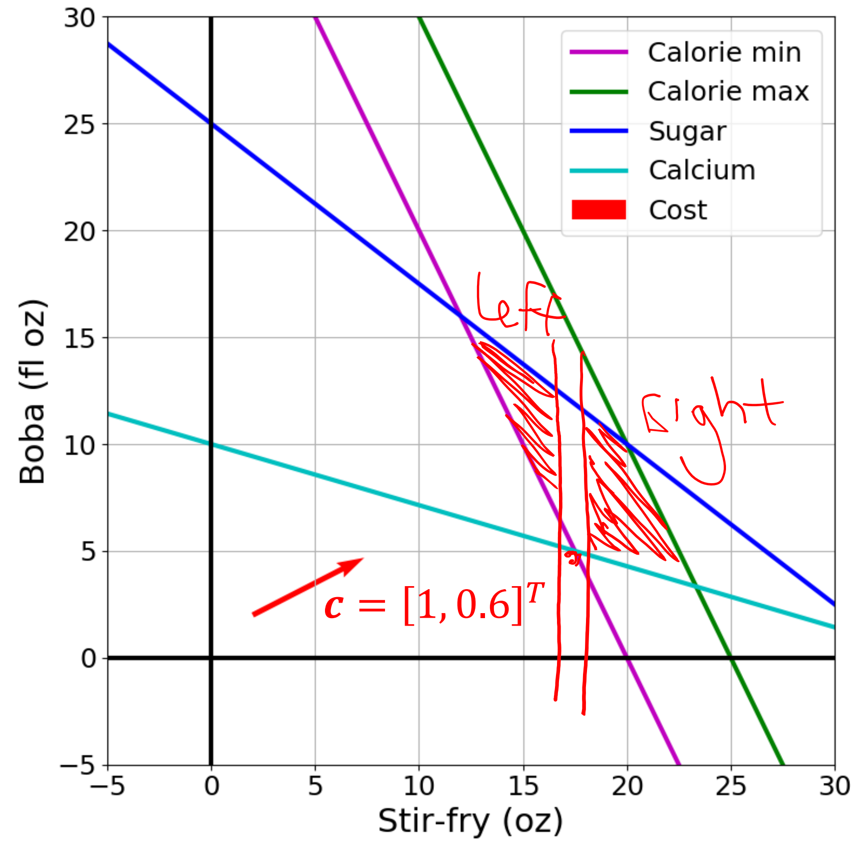
     Right branch: Added constraint $x_i \geq ceil(x_i)$

Note: Only add LPs to the queue if they are feasible

Branch and Bound Example

$$\boldsymbol{x}^\star = (17.5,5)$$
$$y^\star = 20.5$$

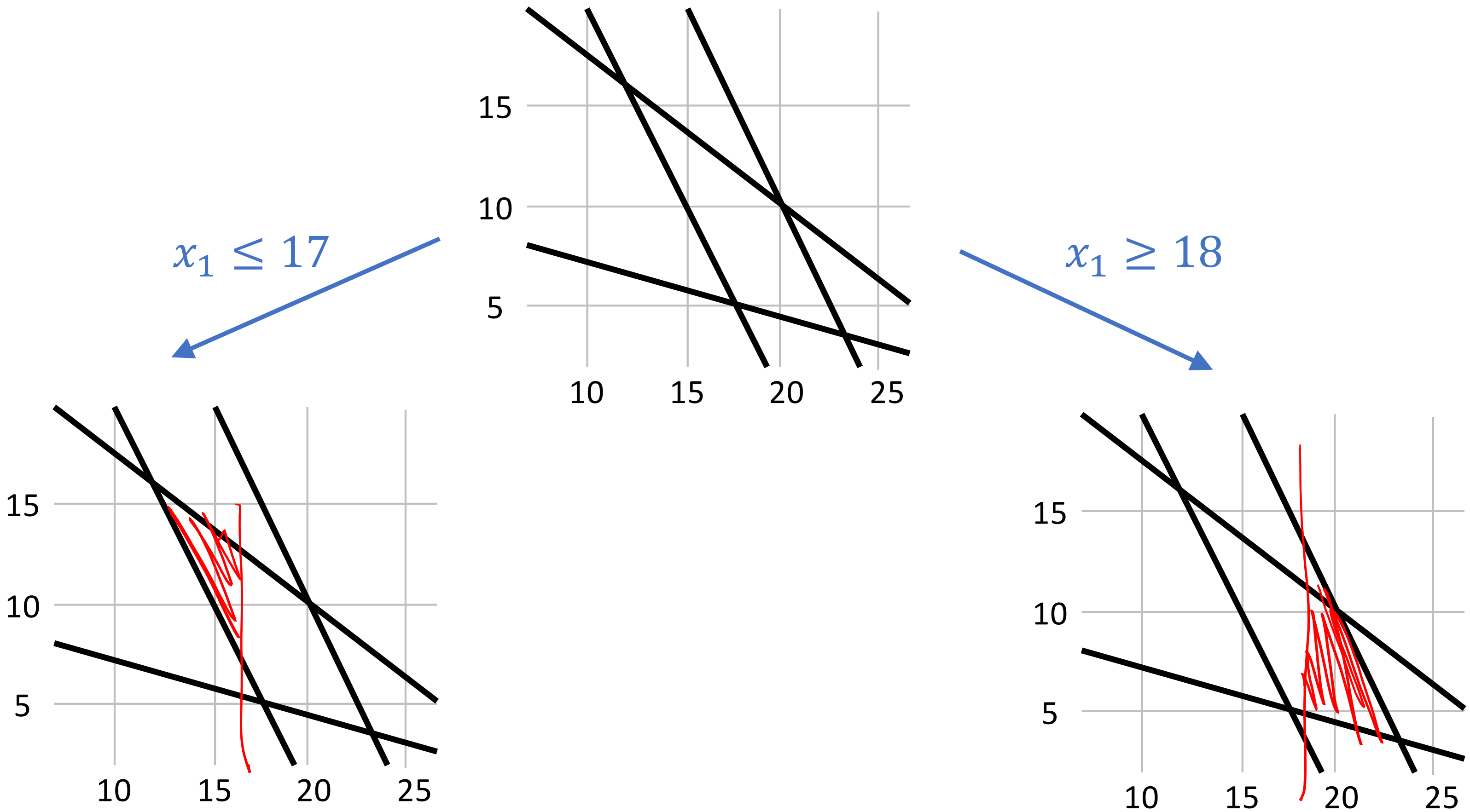
$x_1 \leq 17$ | $x_1 \geq 18$



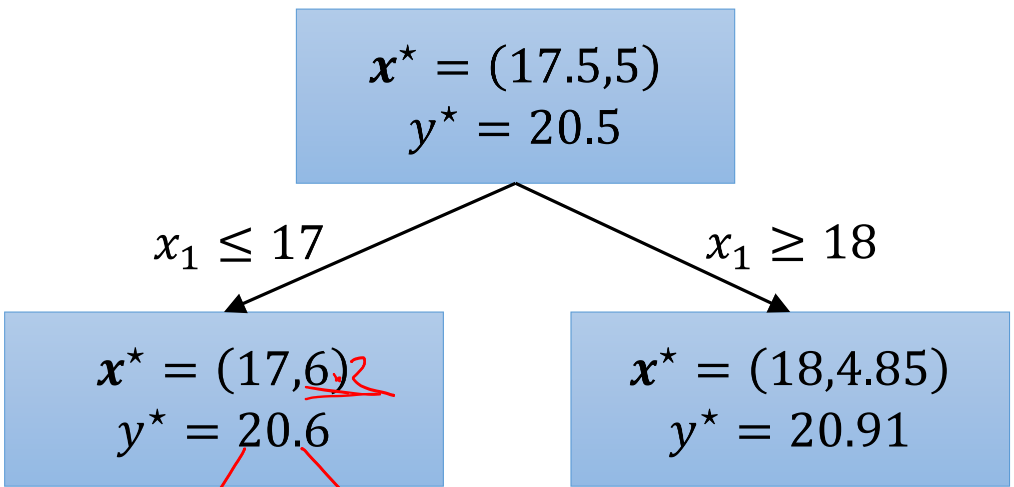
Priority Queue:

1. $\boldsymbol{x}^\star = (17.5,5),\ y^\star = 20.5$

Branch and Bound Example

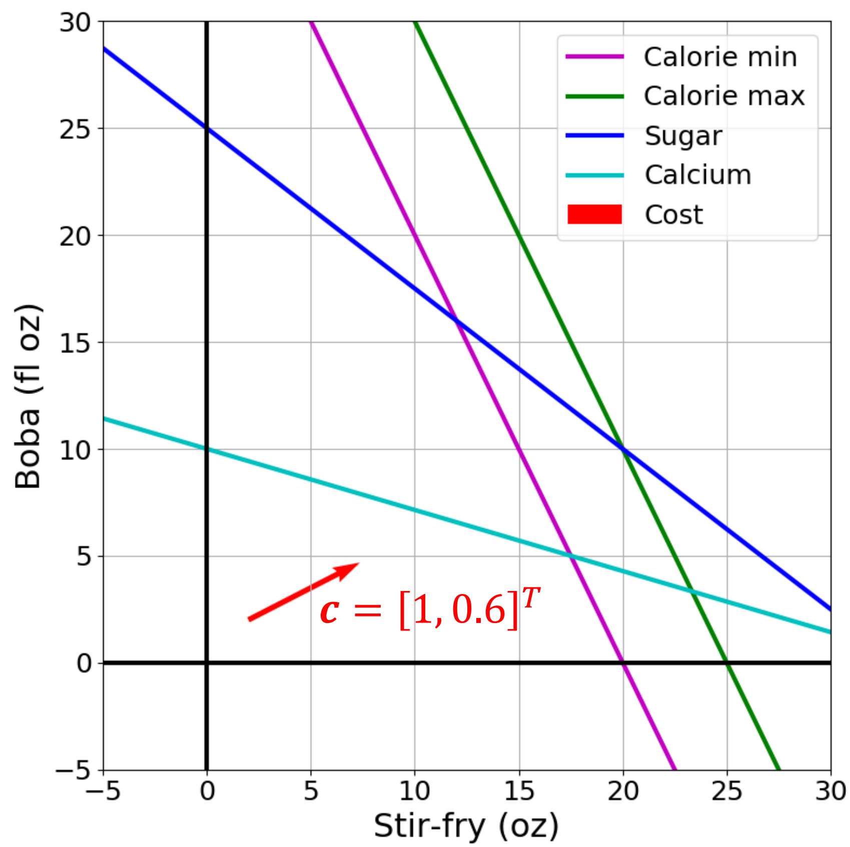


Branch and Bound Example



Priority Queue:

1. $x^\star = (17,6)$, $y^\star = 20.6$
2. $x^\star = (18,4.85)$, $y^\star = 20.91$



Activity + Poll

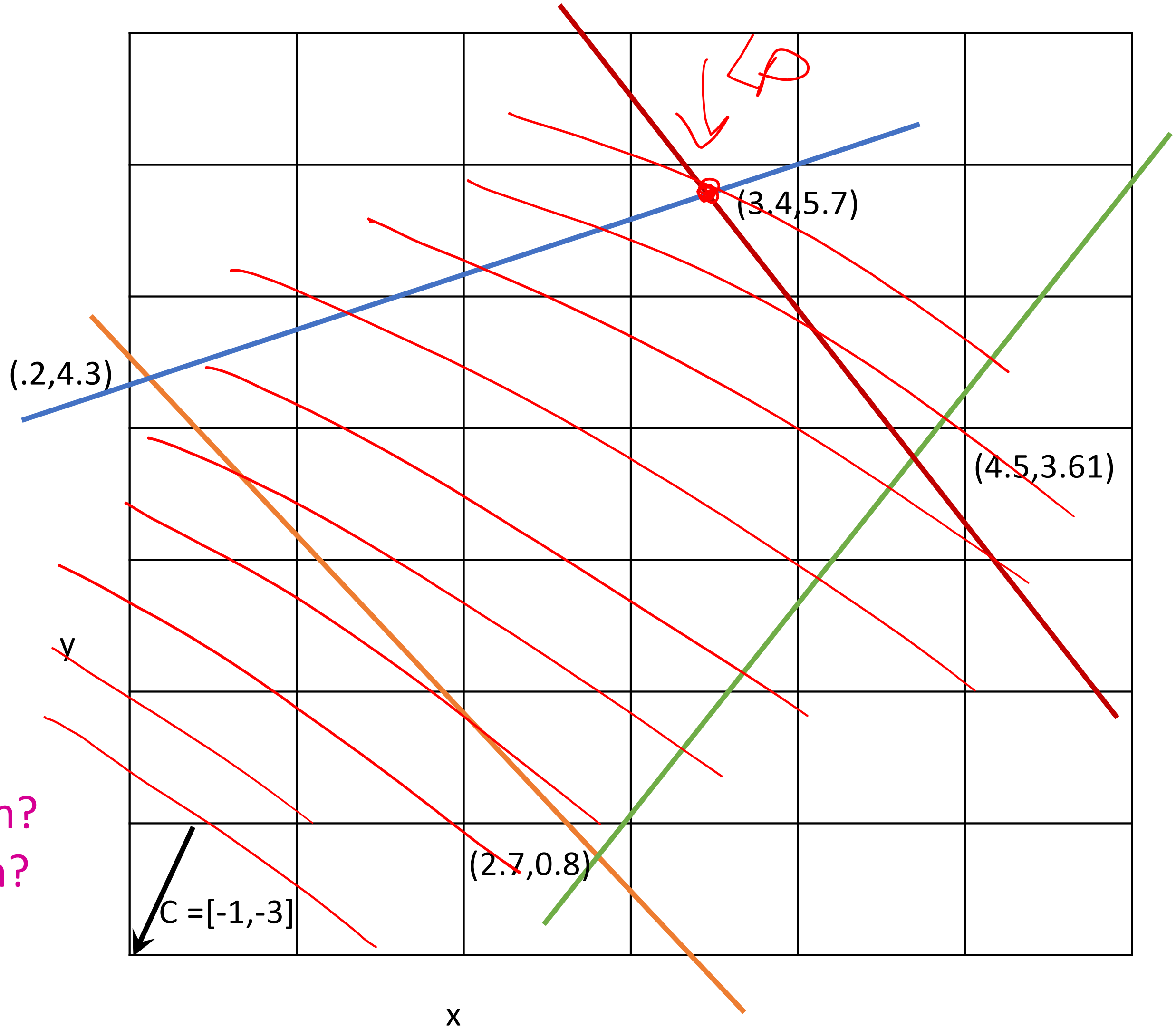
Constraints :

$y = -1.4x + 4.58$

$y = 1.56x + 3.41$

$y = -1.9x + 12.16$

$y = .44x + 4.21$



Priority Queue:

Poll 4: What is the LP solution?

Poll 5: What is the IP solution?

Activity

Constraints :

$y = -1.4x + 4.58$

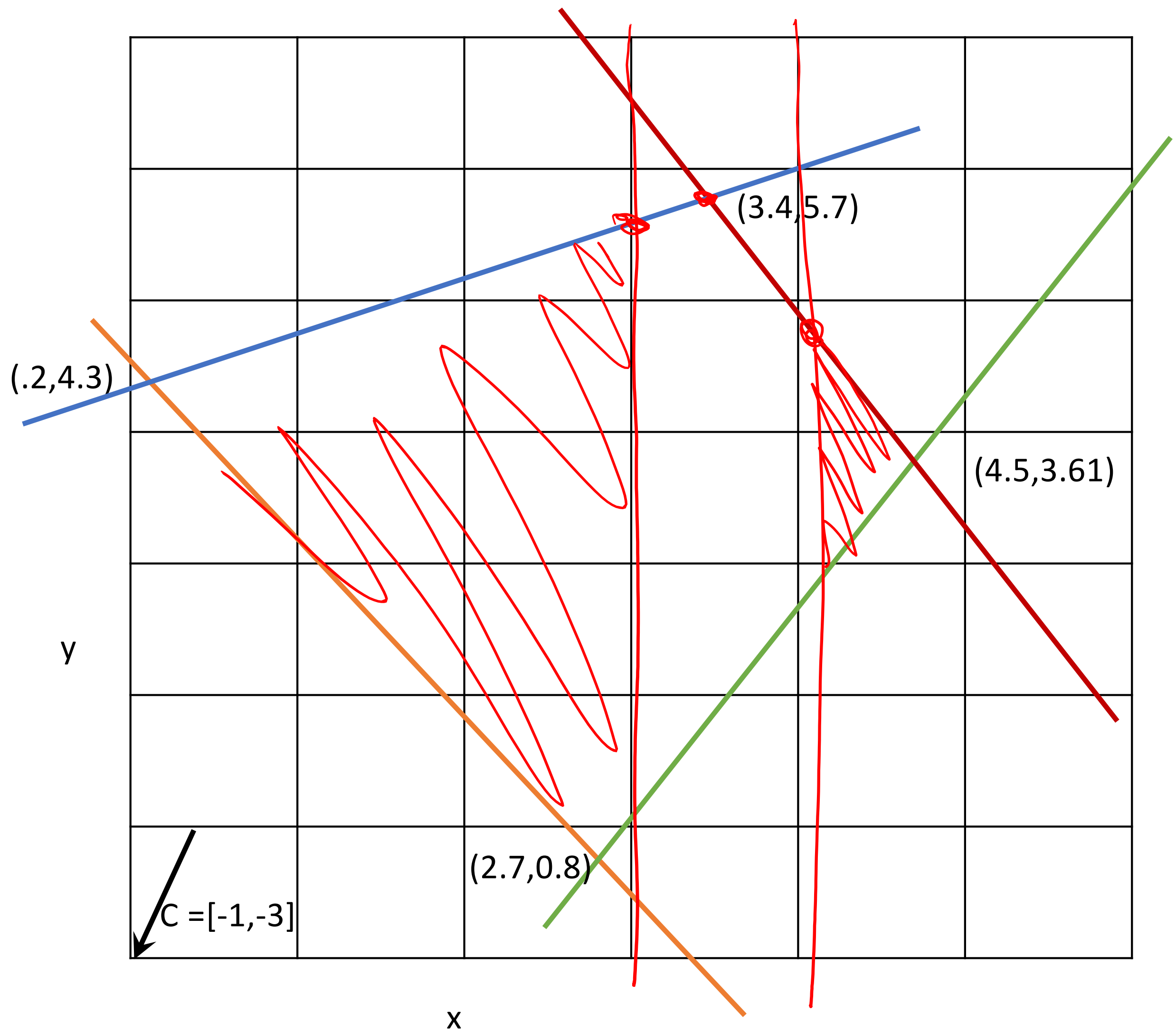
$y = 1.56x + 3.41$

$y = -1.9x + 12.16$

$y = .44x + 4.21$

<u>Priority Queue</u>:

-20.5: (3.4,5.7)



Activity

Constraints :

$y = -1.4x + 4.58$

$y = 1.56x + 3.41$

$y = -1.9x + 12.16$

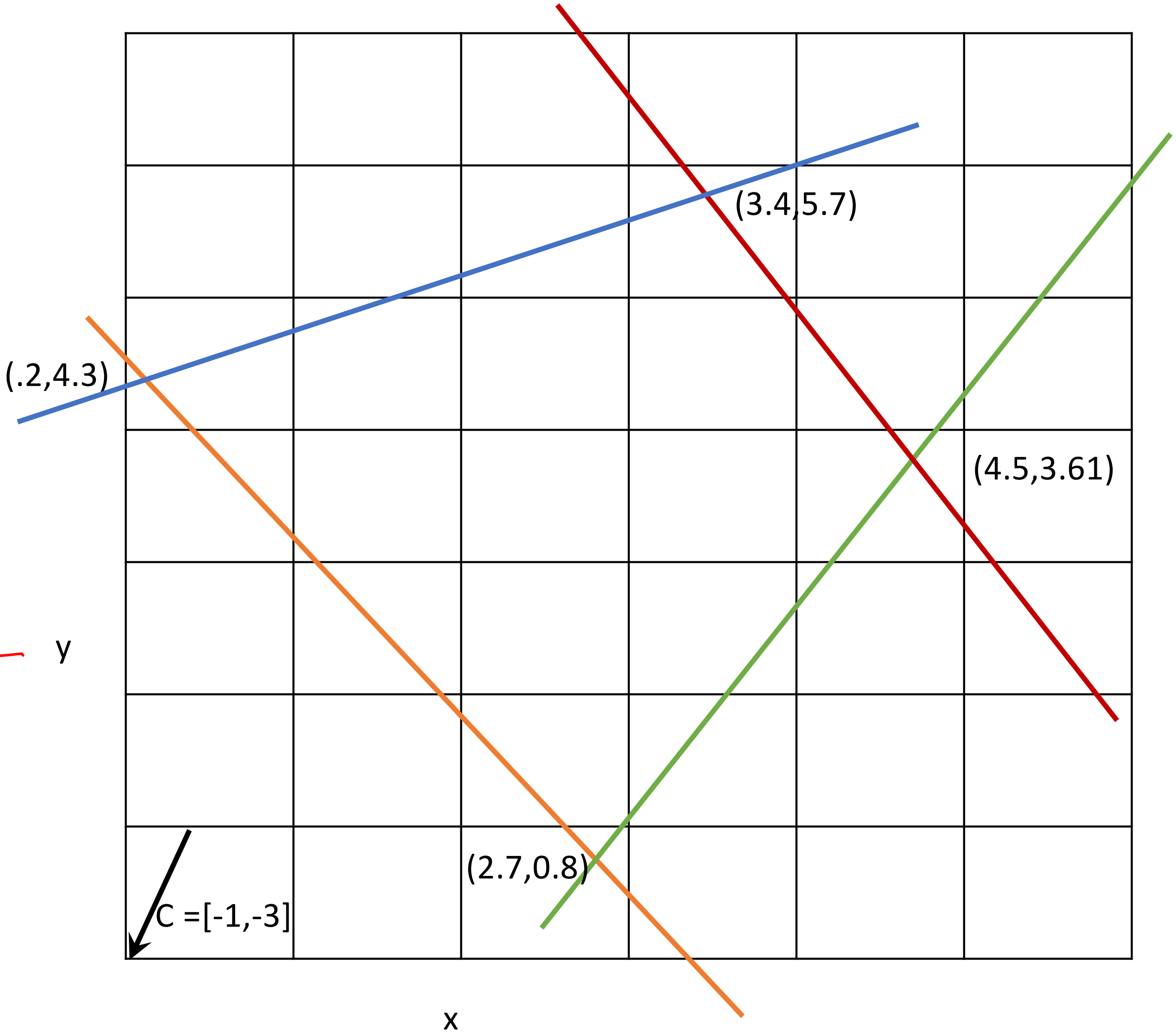
$y = .44x + 4.21$

<u>Priority Queue</u>:

~~-20.5: (3.4,5.7)~~

-19.6: (3,5.53) (x <= 3) ← y

-17.7: (4,4.56) (x >=4)



Activity

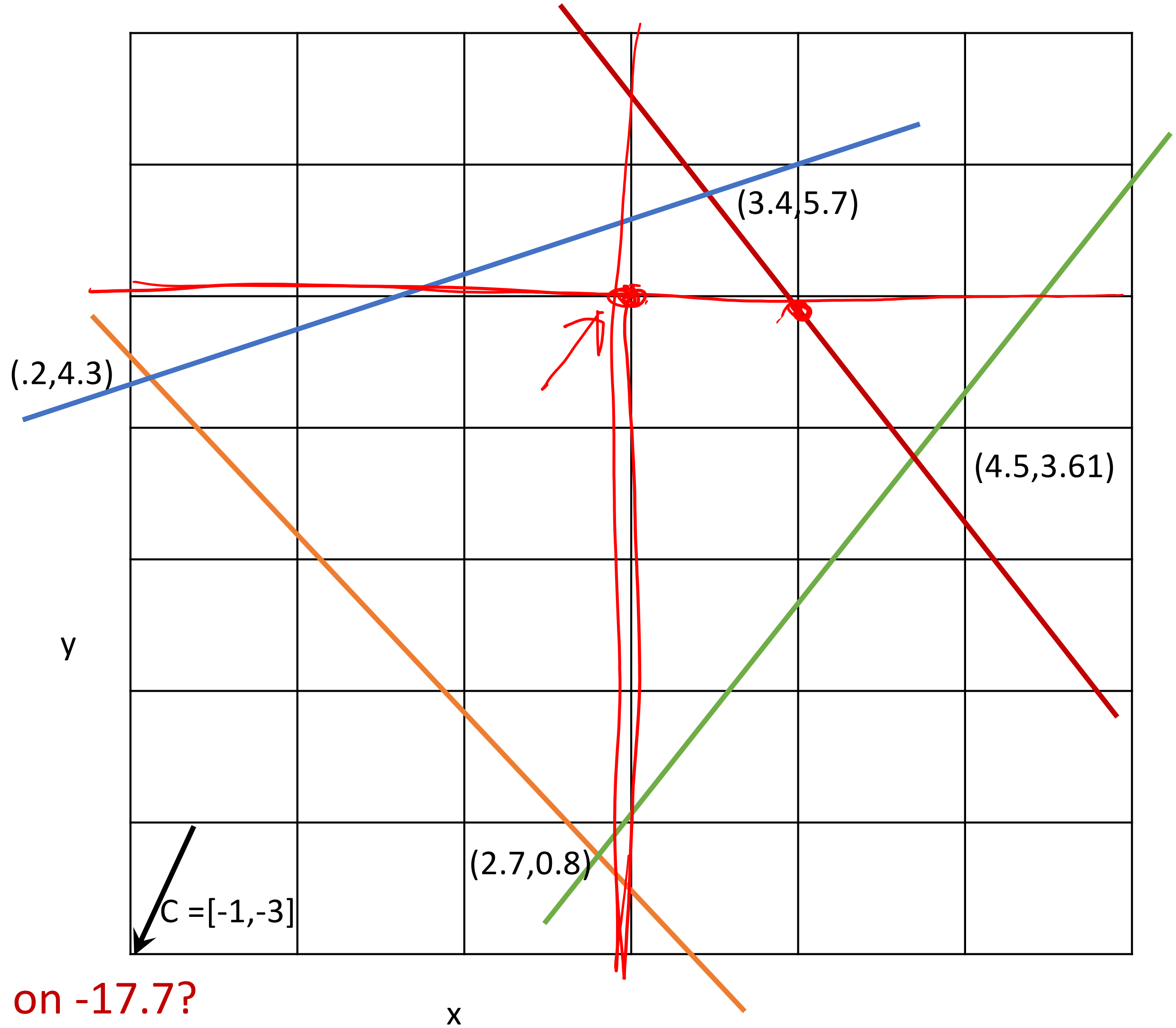
Constraints :

$y = -1.4x + 4.58$

$y = 1.56x + 3.41$

$y = -1.9x + 12.16$

$y = .44x + 4.21$



Priority Queue:

~~-20.5: (3.4,5.7)~~

~~-19.6: (3,5.53) (x <= 3)~~

-17.7: (4,4.56) (x >=4)

-18.0: (3,5) (x<=3,y<=5)

Inf: (x<=3,y>=6)

Why do we not need to recurse on -17.7?