18. Rounding and relaxation

- Decision problems
- Easy and hard examples
- Performance and the future
- Rounding
- Convex relaxation
- Convex hull

Decision problems

A **decision problem** is a yes/no question.

Examples

- Does the following sequence contain the pattern A, A, G ?
  $\{C, T, G, A, T, A, A, G, C, T\}$
- Is there a subset of these numbers that sums to zero?
  $\{-7, -3, -1, 5, 8\}$

NP decision problems

A decision problem is in the class **NP**[1] if instances where the answer is "yes" have efficiently verifiable proofs of the fact that the answer is indeed "yes". Here, "efficient" means polynomial-time in the length of the instance.

Examples

- Does the following sequence contain the pattern A, A, G ?
  $\{C, T, G, A, T, A, A, G, C, T\}$
- Is there a subset of these numbers that sums to zero?
  $\{-7, -3, -1, 5, 8\}$

The red elements prove that the answer is indeed "yes".

(1) "Nondeterministic Polynomial time"

NP decision problems

- The easiest problems are **P**; they can be *solved* efficiently.

  $\Longrightarrow \{C, T, G, A, T, A, A, G, C, T\}$.
  If the sequence has length n, we can determine whether the pattern occurs by using n comparisons (efficient).

- The hardest problems are **NP-complete**. It is not known whether any efficient algorithms exist that guarantee that we won't have to check every possible case...

  $\Longrightarrow \{-7, -3, -1, 5, 8\}$. Need to check all subsets! If the sequence has length n, there are 2^n subsets (exponential).

P = NP?

- It's not actually known whether there is such a thing as "hard" problems in NP! It could be possible that P=NP (all NP problems are solvable in polynomial time).
- This is perhaps the most famous unsolved problem in theoretical computer science.
- Most experts believe that P $\neq$ NP.

Examples in P

- **pattern-matching:** Given a string $x_1 x_2 \ldots x_n$, does it contain a substring $y_1 y_2 \ldots y_k$? (e.g. linear search)
- **sorting a list:** Given a set of numbers $\{x_1, \ldots, x_n\}$ sort it in ascending order. (e.g. bubble sort, quicksort)
- **linear equations:** Solve a system of n linear equations in n variables (e.g. Gaussian elimination)
- **linear programming:** Solve a linear program with n variables and n constraints. (e.g. ellipsoid method)

Note: these are not all decision problems. The decision version would be e.g. "is this list sorted" or "is this LP feasible"?

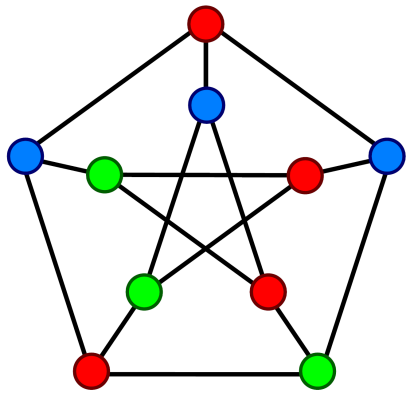
NP-complete problems

- **Traveling salesman (TSP):** Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city exactly once and returns to the origin city?
- **Boolean satisfiability (SAT)**: Given an expression using n boolean variables and the operators *AND*, *OR*, *NOT*, and parentheses, is there a choice of the variables that makes the expression true? Example:

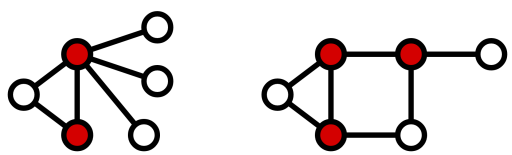
$$(x_1 \vee \neg x_2) \wedge (\neg x_1 \vee x_2 \vee x_3) \wedge \neg x_1$$

NP-complete problems

- ***k*-coloring:** Given a graph, can we color the vertices using k colors so that each edge connects two vertices of a different color? This is in **P** for $k = 2$ only.



- **vertex cover:** Given a graph, can we select k of the vertices so that every vertex is at most one edge away from a selected vertex?



NP-complete problems

- **Integer (linear) programs:** Solving a linear program with integer constraints on the variables.

Every NP problem can be represented as an integer program!

Easy instances

- All problems in NP can be written as IPs.
- (including problems in P)
- So some IPs must be easy to solve...

$$\begin{aligned} \underset{z}{\text{maximize}} \quad & a_1 z_1 + \cdots + a_n z_n \\ \text{subject to:} \quad & z_1 + \cdots + z_n = 1 \\ & z_i \in \{0, 1\} \end{aligned}$$

- Same as $\max\{a_1, \ldots, a_n\}$, which can efficiently be solved!

Bad news

Suppose you'd like to solve a SAT problem with n variables by brute force (checking all 2^n combinations), and you can check 10^9 combinations per second.

n	**time to check all combinations**
10	1 microsecond
30	1 second
50	13 days
70	374 centuries
100	2908 $\times$ (current age of the universe)

Good news

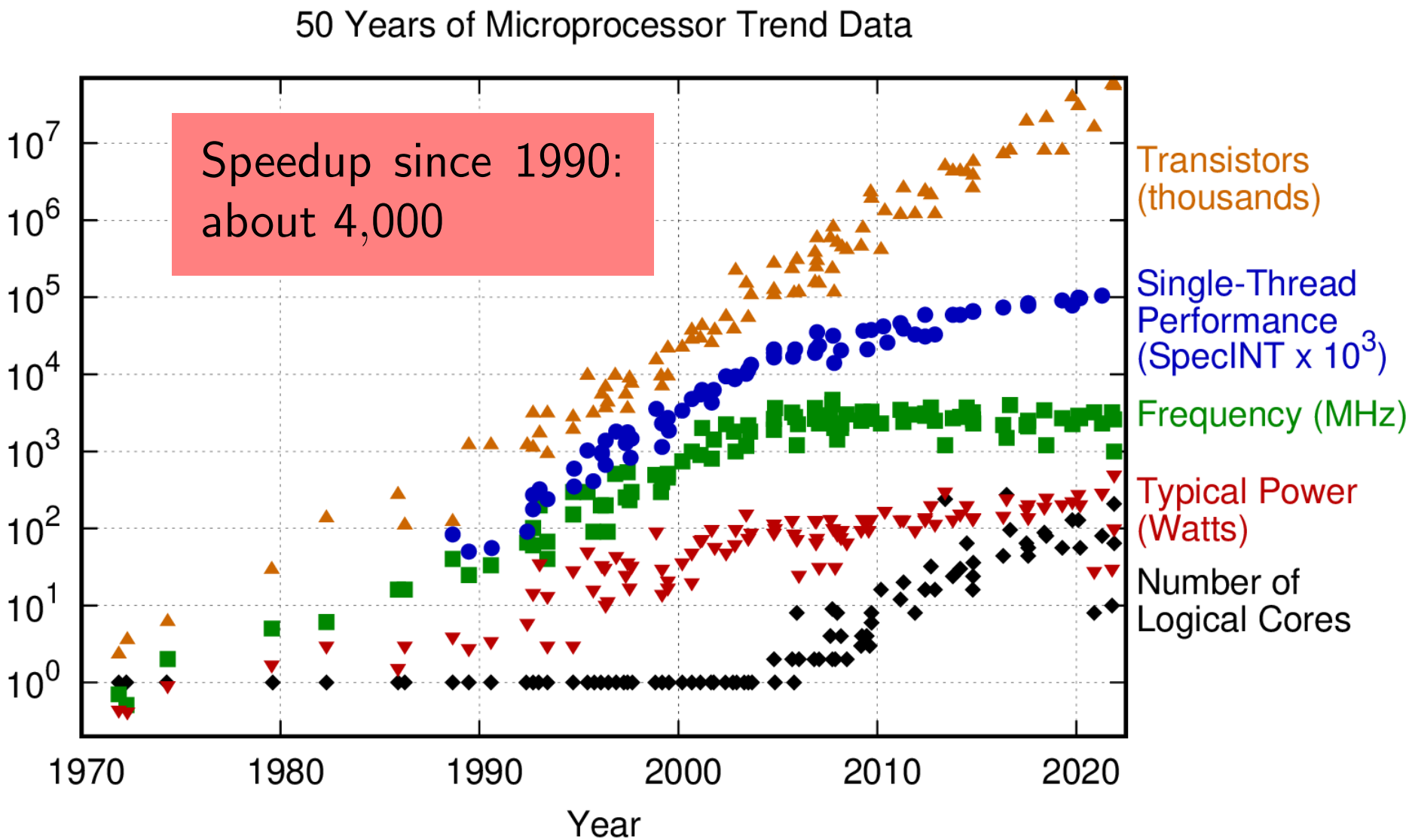
Very large NP-complete problems are solved in practice!

- SAT problems with a million variables
- TSP with a million variables (1000 cities)

How is this possible?

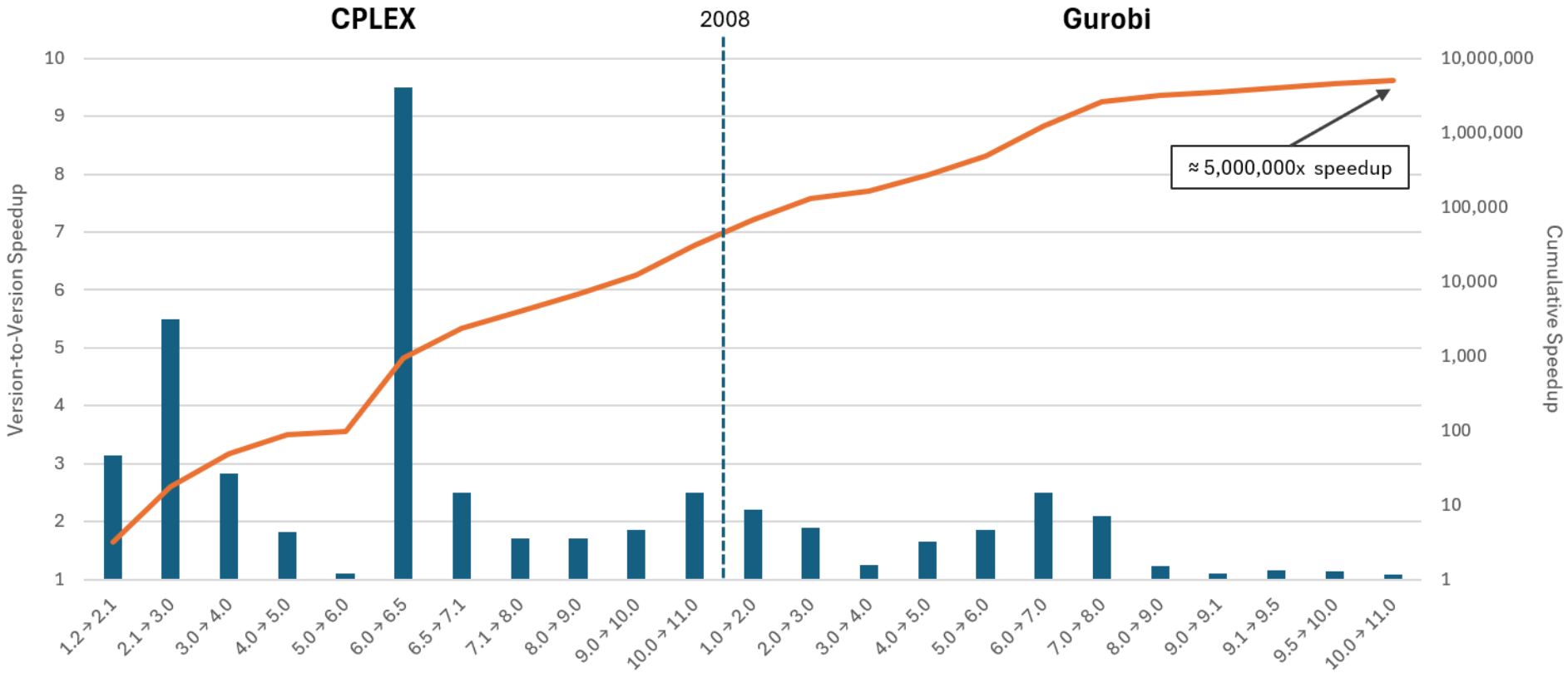
- Instances occurring in practice have special structures that can be exploited.
- Efficient approximation algorithms sometimes exist.
  Example: get within ε of optimal in polynomial time.
- Computers and solvers are both getting faster...

Moore good news: processors



Credit: Karl Rupp Gihub repo

Moore good news: MILP solvers



- Each bar is 1–2 years. Covers 1990–2023.
- Does not include computer speedup (solver only!)

Adapted from: solvermax.com blog

Summary

- Computers have improved steadily. Single-thread computing has stalled, but now we have cloud computing, GPUs, and more...
- Solvers have improved steadily at a much faster rate than computers, and continue to do so. So today's solver on yesterday's hardware would outperform yesterday's solver on todays hardware.
- Total speedup since 1990: about 20 billion times faster. A typical MILP that would have taken 400 years to solve in 1990 can be solved in 1 second today.
- Mileage may vary!

Leyffer–Linderoth–Luedtke complexity [1]

Sven Leyffer (Argonne National Lab)
Jeff Linderoth (UW–Madison)
Jim Luedtke (UW–Madison)



"How many decision variables (n) must a problem have before everyone would be willing to pay \$50 that a state-of-the-art solver would no longer be able to solve it?"

convex and continuous

LP	2×10^8
QP	2×10^6
SOCP	4×10^5
NLP	2×10^5

convex with mixed-integer

MILP	8×10^4
MIQP	4000
MISOCP	4000
MINLP	2000

nonconvex mixed or cont.

QP	600
MIQP	600
NLP	200
MINNLP	200

[1] Not meant to be taken (too) seriously

Rounding

Back to the standard IP formulation:

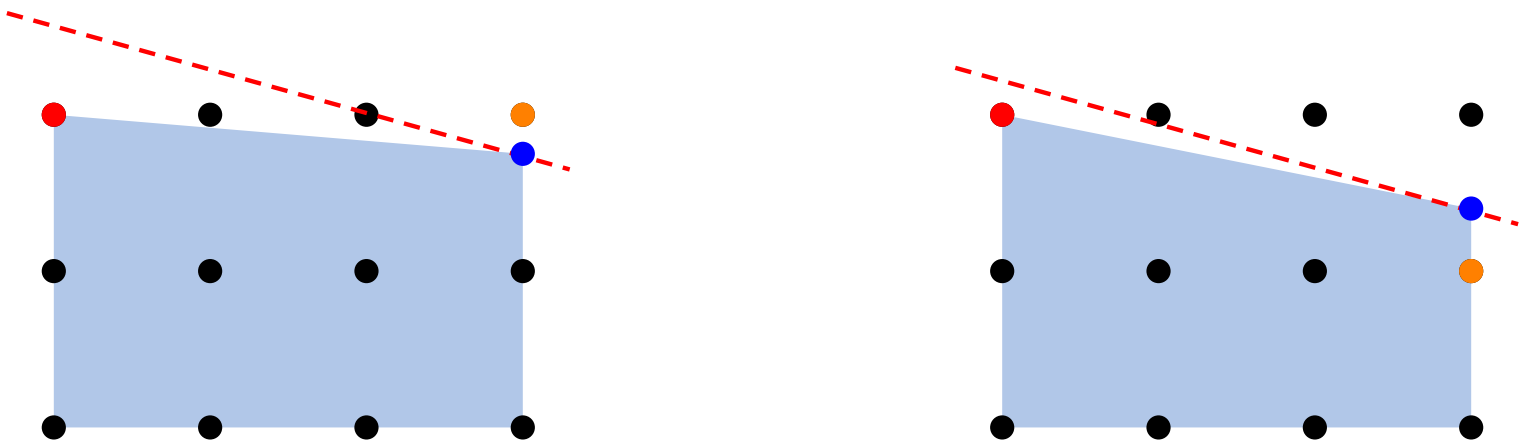
$$\begin{aligned} \underset{x}{\text{maximize}} \quad & c^T x \\ \text{subject to:} \quad & Ax \leq b \\ & x \in \mathbb{Z} \end{aligned}$$

Idea:

- Solve the problem for $x \in \mathbb{R}$ instead (a regular LP).
- Round each x_i in the solution to the nearest integer.
- This usually **does not** work!

Rounding

- If LP solution is already integral, then it is also the exact solution to the original IP. (e.g. min cost flow problems)
- Rounding can lead to an infeasible point
- Rounding can produce a point far from the optimal point



true optimum (●), relaxed optimum (●), rounded (●)

Convex relaxation

$$\underset{x \in S}{\text{minimize}} \quad f(x)$$

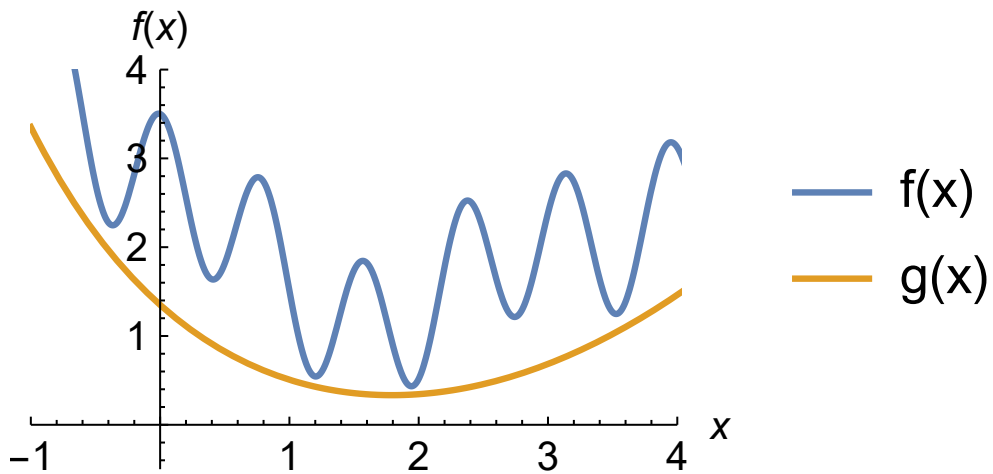
Two ideas we will discuss:

1. *Function relaxation:* if f is troublesome, bound it with a function that is easier to work with, e.g. a convex function.
2. *Constraint relaxation:* If S is troublesome, find a bigger set that is easier to work with, e.g. a convex set.

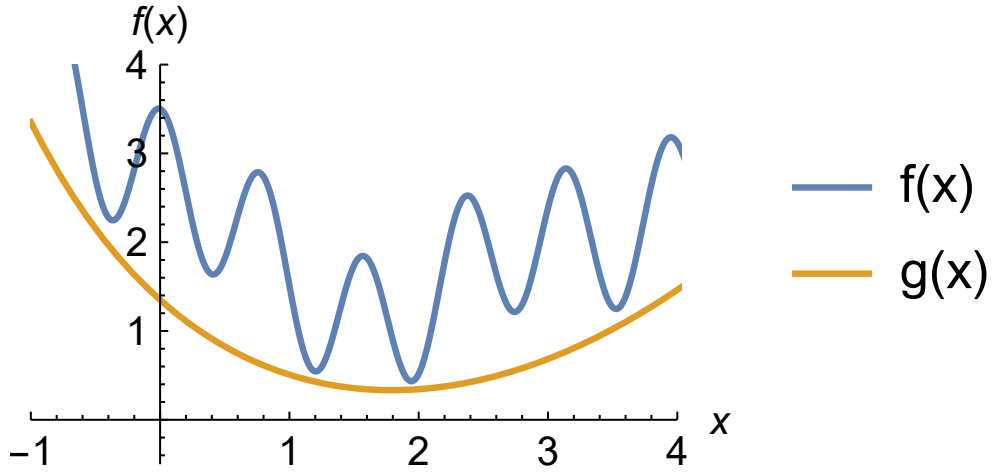
Function relaxation

$$f_{\text{opt}} = \underset{x \in S}{\text{minimize}} \;\; f(x)$$

Suppose we can find g such that $g(x) \leq f(x)$ for all x.
In other words g is a *lower bound* on f.



Function relaxation



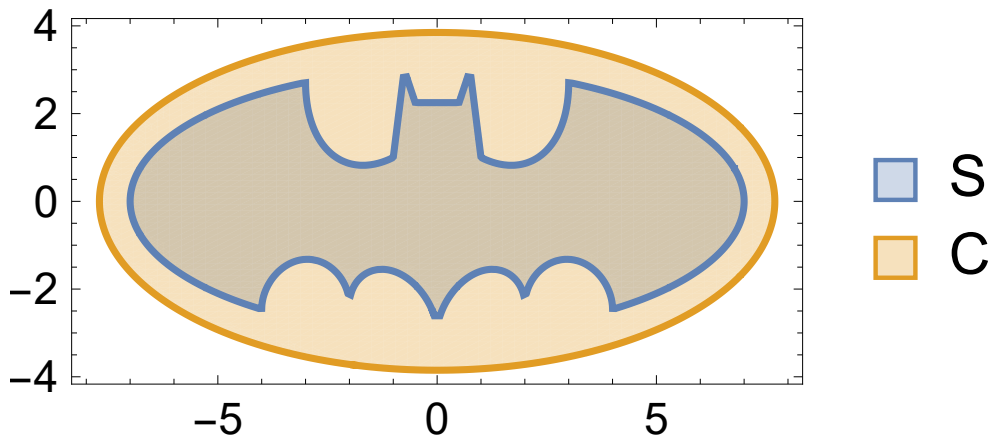
- Solve $g_{\text{opt}} = \min_{x \in S} g(x)$ and let $\hat{x}$ be the corresponding x.
- We have the bounds: $g_{\text{opt}} = g(\hat{x}) \leq f_{\text{opt}} \leq f(\hat{x})$.
- If $f(\hat{x}) = g_{\text{opt}}$ then the bound is tight and $f_{\text{opt}} = f(\hat{x})$.

Pick a convex g so that g_{opt} and $\hat{x}$ are easy to find!

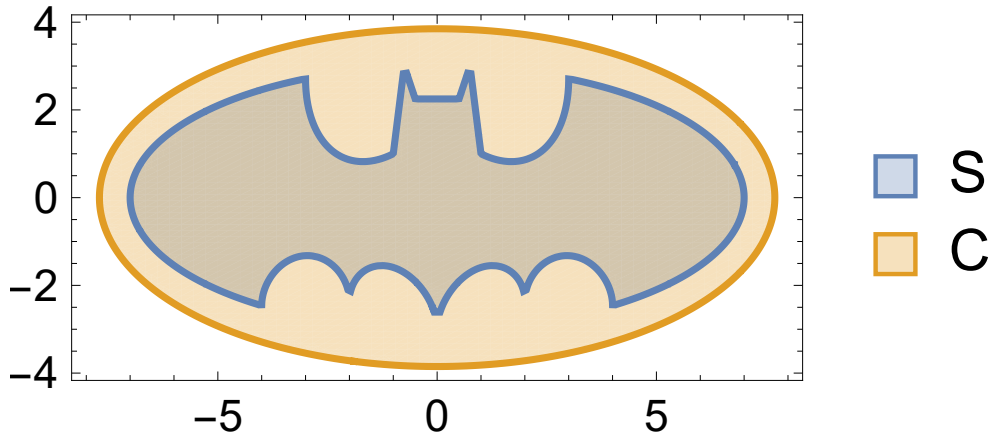
Constraint relaxation

$$f_{\text{opt}} = \underset{x \in S}{\text{minimize}} \;\; f(x)$$

Suppose we can find some set C such that $S \subseteq C$.
In other words, C is a *superset* of S.



Constraint relaxation



- Solve $h_{\text{opt}} = \min_{x \in C} f(x)$ and let $\tilde{x}$ be the optimal x.
- We have the bound: $h_{\text{opt}} = f(\tilde{x}) \leq f_{\text{opt}} \leq f(x)$ for $x \in S$.
- If $\tilde{x} \in S$ then the bound is tight and $f_{\text{opt}} = f(\tilde{x})$.

Pick a convex C so that h_{opt} and $\tilde{x}$ are easy to find!

Common relaxations

1. Boolean constraint:

$$x \in \{0, 1\} \quad \Longrightarrow \quad 0 \leq x \leq 1$$

If x_{opt} is 0 or 1, relaxation is exact.

2. Convex equality:

$$f(x) = 0 \quad \Longrightarrow \quad f(x) \leq 0$$

If $f(x_{\text{opt}}) = 0$, relaxation is exact.

3. A constraint you don't like:

$$x \neq 3 \quad \Longrightarrow \quad \text{just remove the constraint!}$$

If $x_{\text{opt}} \neq 3$, relaxation is exact.

Convex hull

The **convex hull** of a set S, written $\text{conv}(S)$ is the smallest convex set that contains S.

Equivalent definitions:

- The set of all affine combinations of all points in S
- The intersection of all convex sets containing S