# Optimization ... in 45 minutes

L. Nunes Vicente

Department of Mathematics
University of Coimbra

Slides written with the help of Rohollah Garmanjani (Nima)

# Presentation outline

1. Importance of Optimization (illustration by Linear Programming)
2. Application of Optimization (illustration by Classification)
3. Classes of Optimization Problems
4. Type of Mathematics used in Optimization (illustration by Integrality, Convexity, Non-smooth Calculus)

# Importance of Optimization

Optimization has many applications in different fields such as

- Pure and Applied Mathematics: auxiliary problems, bounds ...

# Importance of Optimization

Optimization has many applications in different fields such as

- Pure and Applied Mathematics: auxiliary problems, bounds ...
- Transportation and Management: assignment, scheduling, routing, supply chain ...

# Importance of Optimization

Optimization has many applications in different fields such as

- Pure and Applied Mathematics: auxiliary problems, bounds ...
- Transportation and Management: assignment, scheduling, routing, supply chain ...
- Finance and Economics: portfolio selection, game theory ...

# Importance of Optimization

Optimization has many applications in different fields such as

- Pure and Applied Mathematics: auxiliary problems, bounds ...
- Transportation and Management: assignment, scheduling, routing, supply chain ...
- Finance and Economics: portfolio selection, game theory ...
- Engineering and Computer science: information processing, telecommunication networks, robotics, process engineering ...

# Importance of Optimization

Optimization has many applications in different fields such as

- Pure and Applied Mathematics: auxiliary problems, bounds ...
- Transportation and Management: assignment, scheduling, routing, supply chain ...
- Finance and Economics: portfolio selection, game theory ...
- Engineering and Computer science: information processing, telecommunication networks, robotics, process engineering ...
- Medicine and Biology: medical imaging, diagnosing, radiation treatment ...

# Importance of Optimization

Optimization has many applications in different fields such as

- Pure and Applied Mathematics: auxiliary problems, bounds ...
- Transportation and Management: assignment, scheduling, routing, supply chain ...
- Finance and Economics: portfolio selection, game theory ...
- Engineering and Computer science: information processing, telecommunication networks, robotics, process engineering ...
- Medicine and Biology: medical imaging, diagnosing, radiation treatment ...

And many other applications in Physics, Chemistry, Geology ...

# Importance of Optimization

Roughly 5-10% of Math. journals are in Optimization and related fields. Here are some of Optimization journals:

| | | | |
|---|---|---|---|
| Mathematical Programming | Mathematical Programming Computation | SIAM Journal on Optimization | SIAM Journal on Control and Optimization |
| Mathematics of Operations Research | EURO Journal on Computational Optimization | Operations Research | INFORMS J. Computing |
| Computational Optimization and Applications | IIE Transactions | Journal of Combinatorial Optimization | Journal of Global Optimization |
| Optimization and Engineering | Optimization Methods and Software | Journal of Optimization Theory and Applications | Optimization |
| Optimization Letters | Journal of Combinatorial Optimization | Discrete Optimization | Annals of Operations Research |

# Importance of Optimization

Roughly 5-10% of Math. journals are in Optimization and related fields. Here are some of Optimization journals:

| | | | |
|---|---|---|---|
| Mathematical Programming | Mathematical Programming Computation | SIAM Journal on Optimization | SIAM Journal on Control and Optimization |
| Mathematics of Operations Research | EURO Journal on Computational Optimization | Operations Research | INFORMS J. Computing |
| Computational Optimization and Applications | IIE Transactions | Journal of Combinatorial Optimization | Journal of Global Optimization |
| Optimization and Engineering | Optimization Methods and Software | Journal of Optimization Theory and Applications | Optimization |
| Optimization Letters | Journal of Combinatorial Optimization | Discrete Optimization | Annals of Operations Research |

There are hundreds of software packages for solving different optimization problems. See, for instance:

http://plato.asu.edu/guide.html

# Importance of Optimization

Roughly 5-10% of Math. journals are in Optimization and related fields. Here are some of Optimization journals:

| | | | |
|---|---|---|---|
| Mathematical Programming | Mathematical Programming Computation | SIAM Journal on Optimization | SIAM Journal on Control and Optimization |
| Mathematics of Operations Research | EURO Journal on Computational Optimization | Operations Research | INFORMS J. Computing |
| Computational Optimization and Applications | IIE Transactions | Journal of Combinatorial Optimization | Journal of Global Optimization |
| Optimization and Engineering | Optimization Methods and Software | Journal of Optimization Theory and Applications | Optimization |
| Optimization Letters | Journal of Combinatorial Optimization | Discrete Optimization | Annals of Operations Research |

There are hundreds of software packages for solving different optimization problems. See, for instance:

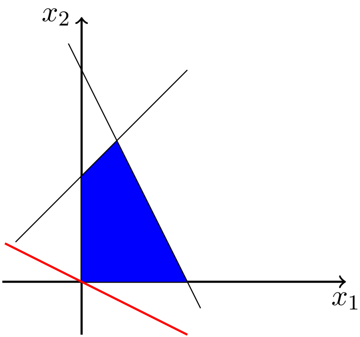
http://plato.asu.edu/guide.html

Optimization is broadly classified by AMS (under 90XX, 49XX, 65XX).

# Linear Programming (LP)

Optimizing a linear objective function subject to a number of linear equality or inequality constraints.

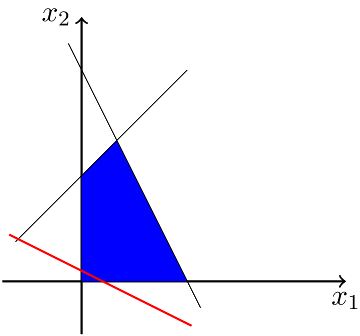
$$\begin{aligned} \text{maximize} \quad & x_1 + 2x_2 \\ \text{subject to} \quad & -x_1 + x_2 \le 2 \\ & 2x_1 + x_2 \le 4 \\ & x_1, x_2 \ge 0 \end{aligned}$$



# Linear Programming (LP)

Optimizing a linear objective function subject to a number of linear equality or inequality constraints.

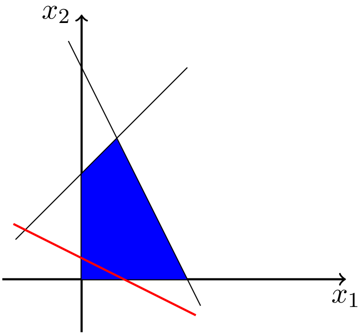
$$\begin{aligned} \text{maximize} \quad & x_1 + 2x_2 \\ \text{subject to} \quad & -x_1 + x_2 \leq 2 \\ & 2x_1 + x_2 \leq 4 \\ & x_1, x_2 \geq 0 \end{aligned}$$



# Linear Programming (LP)

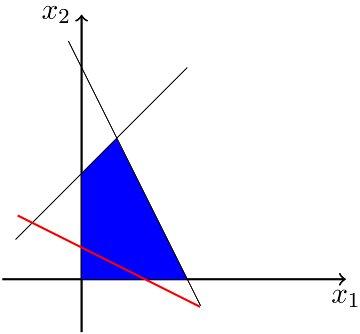
Optimizing a linear objective function subject to a number of linear equality or inequality constraints.

$$\begin{aligned} \text{maximize} \quad & x_1 + 2x_2 \\ \text{subject to} \quad & -x_1 + x_2 \le 2 \\ & 2x_1 + x_2 \le 4 \\ & x_1, x_2 \ge 0 \end{aligned}$$



Optimizing a linear objective function subject to a number of linear equality or inequality constraints.

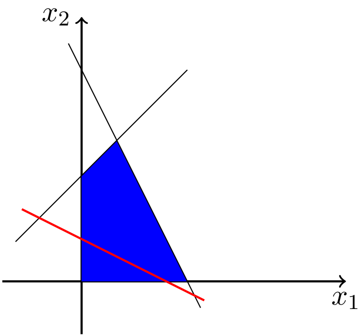
$$\begin{aligned} \text{maximize} \quad & x_1 + 2x_2 \\ \text{subject to} \quad & -x_1 + x_2 \le 2 \\ & 2x_1 + x_2 \le 4 \\ & x_1, x_2 \ge 0 \end{aligned}$$



# Linear Programming (LP)

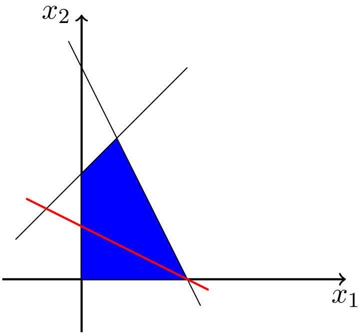
Optimizing a linear objective function subject to a number of linear equality or inequality constraints.

$$\begin{aligned} \text{maximize} \quad & x_1 + 2x_2 \\ \text{subject to} \quad & -x_1 + x_2 \le 2 \\ & 2x_1 + x_2 \le 4 \\ & x_1, x_2 \ge 0 \end{aligned}$$



Optimizing a linear objective function subject to a number of linear equality or inequality constraints.

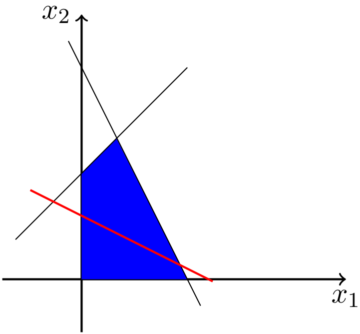
$$\begin{aligned} \text{maximize} \quad & x_1 + 2x_2 \\ \text{subject to} \quad & -x_1 + x_2 \leq 2 \\ & 2x_1 + x_2 \leq 4 \\ & x_1, x_2 \geq 0 \end{aligned}$$



# Linear Programming (LP)

Optimizing a linear objective function subject to a number of linear equality or inequality constraints.

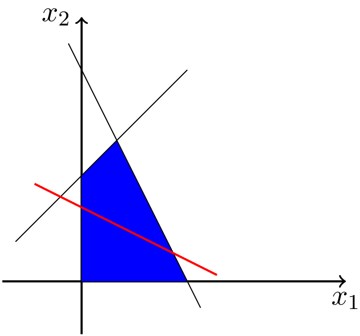
$$
\begin{aligned}
\text{maximize} \quad & x_1 + 2x_2 \\
\text{subject to} \quad & -x_1 + x_2 \leq 2 \\
& 2x_1 + x_2 \leq 4 \\
& x_1, x_2 \geq 0
\end{aligned}
$$



# Linear Programming (LP)

Optimizing a linear objective function subject to a number of linear equality or inequality constraints.

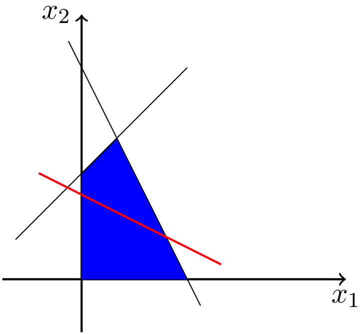
$$\begin{aligned} \text{maximize} \quad & x_1 + 2x_2 \\ \text{subject to} \quad & -x_1 + x_2 \leq 2 \\ & 2x_1 + x_2 \leq 4 \\ & x_1, x_2 \geq 0 \end{aligned}$$



# Linear Programming (LP)

Optimizing a linear objective function subject to a number of linear equality or inequality constraints.

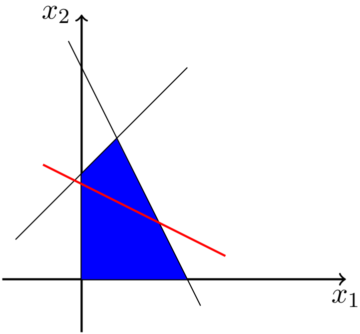
$$\begin{aligned} \text{maximize} \quad & \color{red}{x_1 + 2x_2} \\ \text{subject to} \quad & -x_1 + x_2 \leq 2 \\ & 2x_1 + x_2 \leq 4 \\ & x_1, x_2 \geq 0 \end{aligned}$$



# Linear Programming (LP)

Optimizing a linear objective function subject to a number of linear equality or inequality constraints.

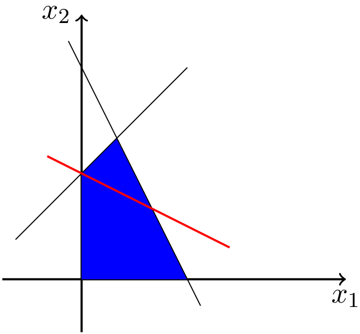
$$\begin{aligned} \text{maximize} \quad & x_1 + 2x_2 \\ \text{subject to} \quad & -x_1 + x_2 \le 2 \\ & 2x_1 + x_2 \le 4 \\ & x_1, x_2 \ge 0 \end{aligned}$$



# Linear Programming (LP)

Optimizing a linear objective function subject to a number of linear equality or inequality constraints.

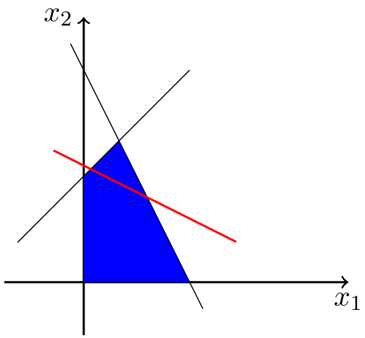
$$\begin{array}{ll} \text{maximize} & {\color{red} x_1 + 2x_2} \\ \text{subject to} & -x_1 + x_2 \ \leq \ 2 \\ & 2x_1 + x_2 \ \leq \ 4 \\ & x_1, x_2 \ \geq \ 0 \end{array}$$



# Linear Programming (LP)

Optimizing a linear objective function subject to a number of linear equality or inequality constraints.

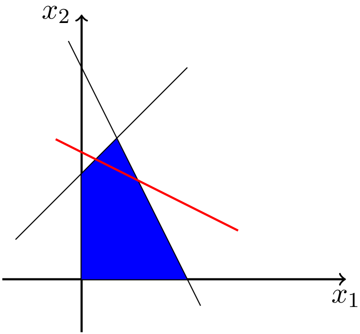
$$\begin{array}{ll} \text{maximize} & x_1 + 2x_2 \\ \text{subject to} & -x_1 + x_2 \leq 2 \\ & 2x_1 + x_2 \leq 4 \\ & x_1, x_2 \geq 0 \end{array}$$



# Linear Programming (LP)

Optimizing a linear objective function subject to a number of linear equality or inequality constraints.

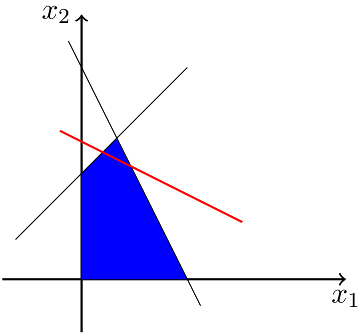
$$\begin{aligned} \text{maximize} \quad & x_1 + 2x_2 \\ \text{subject to} \quad & -x_1 + x_2 \leq 2 \\ & 2x_1 + x_2 \leq 4 \\ & x_1, x_2 \geq 0 \end{aligned}$$



# Linear Programming (LP)

Optimizing a linear objective function subject to a number of linear equality or inequality constraints.

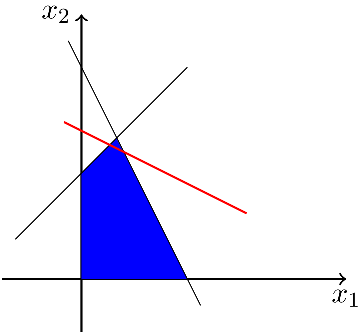
$$\begin{aligned} \text{maximize} \quad & x_1 + 2x_2 \\ \text{subject to} \quad & -x_1 + x_2 \le 2 \\ & 2x_1 + x_2 \le 4 \\ & x_1, x_2 \ge 0 \end{aligned}$$



# Linear Programming (LP)

Optimizing a linear objective function subject to a number of linear equality or inequality constraints.

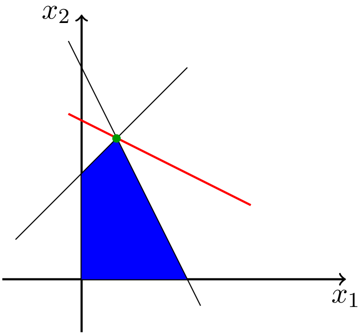
$$
\begin{array}{ll}
\text{maximize} & x_1 + 2x_2 \\
\text{subject to} & -x_1 + x_2 \leq 2 \\
& 2x_1 + x_2 \leq 4 \\
& x_1, x_2 \geq 0
\end{array}
$$



# Linear Programming (LP)

Optimizing a linear objective function subject to a number of linear equality or inequality constraints.

$$
\begin{aligned}
\text{maximize} \quad & x_1 + 2x_2 \\
\text{subject to} \quad & -x_1 + x_2 \leq 2 \\
& 2x_1 + x_2 \leq 4 \\
& x_1, x_2 \geq 0
\end{aligned}
$$



# Simplex algorithm

The simplex algorithm was developed by George Dantzig in 1947.

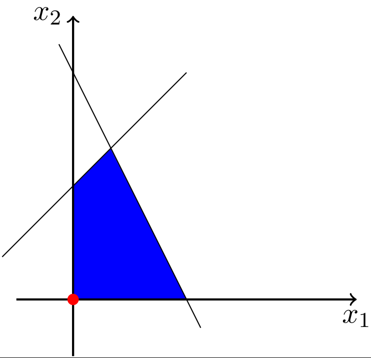
$$
\begin{aligned}
\text{maximize} \quad & x_1 + 2x_2 \\
\text{subject to} \quad & -x_1 + x_2 + s_1 = 2 \\
& 2x_1 + x_2 + s_2 = 4 \\
& x_1, x_2, s_1, s_2 \geq 0
\end{aligned}
$$

# Simplex algorithm

The simplex algorithm was developed by George Dantzig in 1947.

$$\begin{aligned} \text{maximize} \quad & x_1 + 2x_2 \\ \text{subject to} \quad & -x_1 + x_2 + s_1 = 2 \\ & 2x_1 + x_2 + s_2 = 4 \\ & x_1, x_2, s_1, s_2 \geq 0 \end{aligned}$$

| BV | $x_1$ | $x_2$ | $s_1$ | $s_2$ | RHS |
|---|---|---|---|---|---|
| $s_1$ | $-1$ | $\textcircled{1}$ | 1 | 0 | $2 \leftarrow$ |
| $s_2$ | 2 | 1 | 0 | 1 | 4 |
| $z$ | $-1$ | $-2\uparrow$ | 0 | 0 | 0 |

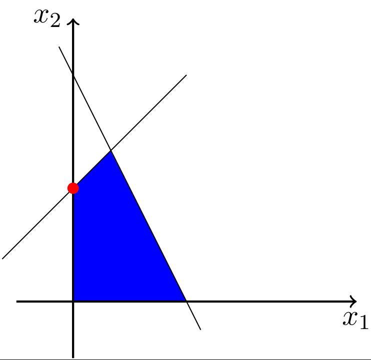


# Simplex algorithm

The simplex algorithm was developed by George Dantzig in 1947.

$$\begin{aligned} \text{maximize} \quad & x_1 + 2x_2 \\ \text{subject to} \quad & -x_1 + x_2 + s_1 = 2 \\ & 2x_1 + x_2 + s_2 = 4 \\ & x_1, x_2, s_1, s_2 \geq 0 \end{aligned}$$

| BV | $x_1$ | $x_2$ | $s_1$ | $s_2$ | RHS |
|---|---|---|---|---|---|
| $x_2$ | $-1$ | 1 | 1 | 0 | 2 |
| $s_2$ | (3) | 0 | $-1$ | 1 | 2 ← |
| $z$ | $-3$ ↑ | 0 | 2 | 0 | 4 |

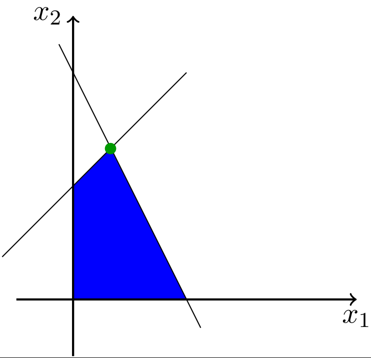


# Simplex algorithm

The simplex algorithm was developed by George Dantzig in 1947.

$$\begin{aligned} \text{maximize} \quad & {\color{red} x_1 + 2x_2} \\ \text{subject to} \quad & -x_1 + x_2 + s_1 = 2 \\ & 2x_1 + x_2 + s_2 = 4 \\ & x_1, x_2, s_1, s_2 \geq 0 \end{aligned}$$

| BV | $x_1$ | $x_2$ | $s_1$ | $s_2$ | RHS |
|---|---|---|---|---|---|
| $x_2$ | $0$ | $1$ | $2/3$ | $1/3$ | $8/3$ |
| $x_1$ | $1$ | $0$ | $-1/3$ | $1/3$ | $2/3$ |
| $z$ | $0$ | $0$ | $1$ | $3$ | $6$ |



# Worst case complexity of simplex algorithm

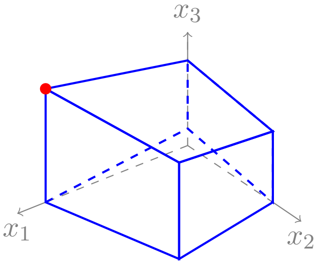
In 1973, Klee and Minty showed that the simplex algorithm performs badly when applied to a perturbed cube.

# Worst case complexity of simplex algorithm

In 1973, Klee and Minty showed that the simplex algorithm performs badly when applied to a perturbed cube.

Assume we want to minimize $x_3$ in the following region.

The simplex algorithm meets $2^3 - 1$ corner points before reaching the optimal one:

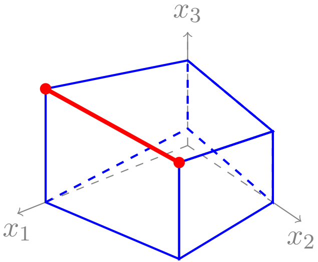


# Worst case complexity of simplex algorithm

In 1973, Klee and Minty showed that the simplex algorithm performs badly when applied to a perturbed cube.

Assume we want to minimize $x_3$ in the following region.

The simplex algorithm meets $2^3 - 1$ corner points before reaching the optimal one:

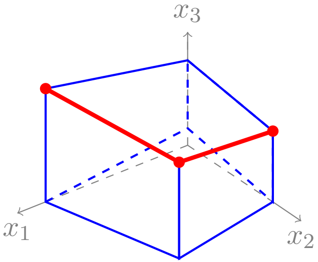


# Worst case complexity of simplex algorithm

In 1973, Klee and Minty showed that the simplex algorithm performs badly when applied to a perturbed cube.

Assume we want to minimize $x_3$ in the following region.

The simplex algorithm meets $2^3 - 1$ corner points before reaching the optimal one:

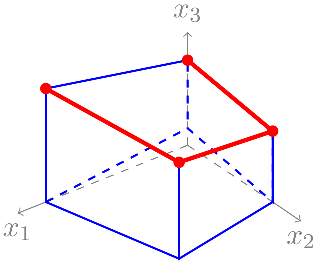


# Worst case complexity of simplex algorithm

In 1973, Klee and Minty showed that the simplex algorithm performs badly when applied to a perturbed cube.

Assume we want to minimize $x_3$ in the following region.

The simplex algorithm meets $2^3 - 1$ corner points before reaching the optimal one:

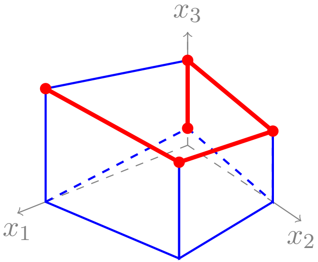


# Worst case complexity of simplex algorithm

In 1973, Klee and Minty showed that the simplex algorithm performs badly when applied to a perturbed cube.

Assume we want to minimize $x_3$ in the following region.

The simplex algorithm meets $2^3 - 1$ corner points before reaching the optimal one:

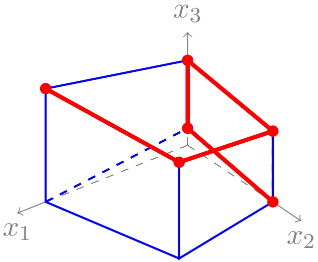


# Worst case complexity of simplex algorithm

In 1973, Klee and Minty showed that the simplex algorithm performs badly when applied to a perturbed cube.

Assume we want to minimize $x_3$ in the following region.

The simplex algorithm meets $2^3 - 1$ corner points before reaching the optimal one:

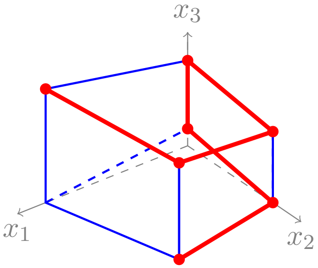


# Worst case complexity of simplex algorithm

In 1973, Klee and Minty showed that the simplex algorithm performs badly when applied to a perturbed cube.

Assume we want to minimize $x_3$ in the following region.

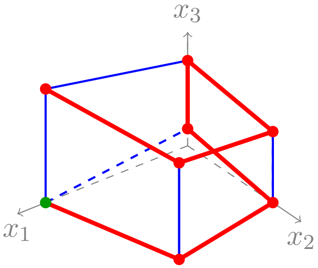
The simplex algorithm meets $2^3 - 1$ corner points before reaching the optimal one:



In 1973, Klee and Minty showed that the simplex algorithm performs badly when applied to a perturbed cube.

Assume we want to minimize $x_3$ in the following region.

The simplex algorithm meets $2^3 - 1$ corner points before reaching the optimal one:

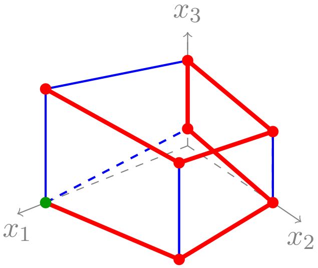


# Worst case complexity of simplex algorithm

In 1973, Klee and Minty showed that the simplex algorithm performs badly when applied to a perturbed cube.

Assume we want to minimize $x_3$ in the following region.

The simplex algorithm meets $2^3 - 1$ corner points before reaching the optimal one:




In $n$ dimensions, the cost is $2^n - 1$, which shows an exponential-time complexity.

# The breakthrough

In 1984, Narendra Karmarkar introduced a polynomial-time algorithm for solving LP problems.

# The breakthrough

In 1984, Narendra Karmarkar introduced a polynomial-time algorithm for solving LP problems.

His discovery received a huge media coverage. The news appeared in the front page of the New York Times:

### *Breakthrough in Problem Solving*

By JAMES GLEICK

A 28-year-old mathematician at A.T.&T. Bell Laboratories has made a startling theoretical breakthrough in the solving of systems of equations that often grow too vast and complex for the most powerful computers.

The discovery, which is to be formally published next month, is already circulating rapidly through the mathematical world. It has also set off a deluge of inquiries from brokerage houses, oil companies and airlines, industries with millions of dollars at stake in problems known as linear programming.

**Faster Solutions Seen**

These problems are fiendishly complicated systems, often with thousands of variables. They arise in a variety of commercial and government applications, ranging from allocating time on a communications satellite to routing millions of telephone calls over long distances, or whenever a limited, expensive resource must be spread most efficiently among competing users. And investment companies use them in creating portfolios with the best mix of stocks and bonds.

The Bell Labs mathematician, Dr. Narendra Karmarkar, has devised a radically new procedure that may speed the routine handling of such problems by businesses and Government agencies and also make it possible to tackle problems that are now far out of reach.

"This is a path-breaking result," said Dr, Ronald L. Graham, director of mathematical sciences for Bell Labs in Murray Hill, N.J.

"Science has its moments of great progress, and this may well be one of them."

Because problems in linear programming can have billions or more possible answers, even high-speed computers cannot check every one. So computers must use a special procedure, an algorithm, to examine as few answers as possible before finding the best one — typically the one that minimizes cost or maximizes efficiency.

A procedure devised in 1947, the simplex method, is now used for such problems,

Continued on Page A19, Column 1

THE NEW YORK TIMES, November 19, 1984

Karmarkar at Bell Labs: an equation to find a new way through the maze

### Folding the Perfect Corner

*A young Bell scientist makes a major math breakthrough*

Every day 1,200 American Airlines jets crisscross the U.S., Mexico, Canada and the Caribbean, stopping in 110 cities and bearing over 80,000 passengers. More than 4,000 pilots, copilots, flight personnel, maintenance workers and baggage carriers are shuffled among the flights; a total of 3.6 million gal. of high-octane fuel is burned. Nuts, bolts, altimeters, landing gears and the like must be checked at each destination. And while performing these scheduling gymnastics, the company must keep a close eye on costs, projected revenue and profits.

Like American Airlines, thousands of companies must routinely untangle the myriad variables that complicate the efficient distribution of their resources. Solving such monstrous problems requires the use of an abstruse branch of mathematics known as linear programming. It is the kind of math that has frustrated theoreticians for years, and even the fastest and most powerful computers have had great difficulty juggling the bits and pieces of data. Now Narendra Karmarkar, a 28-year-old Indian-born mathematician at Bell Laboratories in Murray Hill, N.J., after only a years' work has cracked the puzzle of linear programming by devising a new algorithm, a step-by-step mathematical formula. He has translated the procedure into a program that should allow computers to track a greater combination of tasks than ever before and in a fraction of the time.

Unlike most advances in theoretical mathematics, Karmarkar's work will have an immediate and major impact on the real world. "Breakthrough is one of the most abused words in science," says Ronald Graham, director of mathematical sciences at Bell Labs. "But this is one situation where it is truly appropriate."

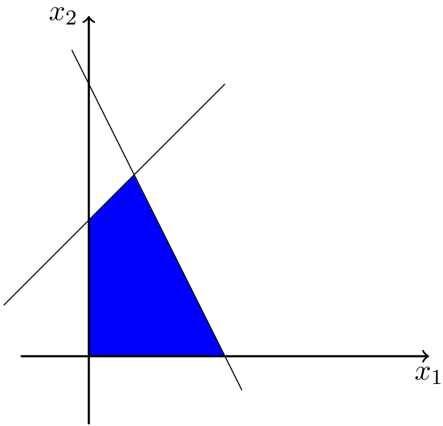
Before the Karmarkar method, linear equations could be solved only in a cumbersome fashion, ironically known as the simplex method, devised by Mathematician George Dantzig in 1947. Problems are conceived of as giant geodesic domes with thousands of sides. Each corner of a facet on the dome

TIME MAGAZINE, December 3, 1984

# Interior point method (central path)

The iterates, instead, follow a path inside the feasible region.

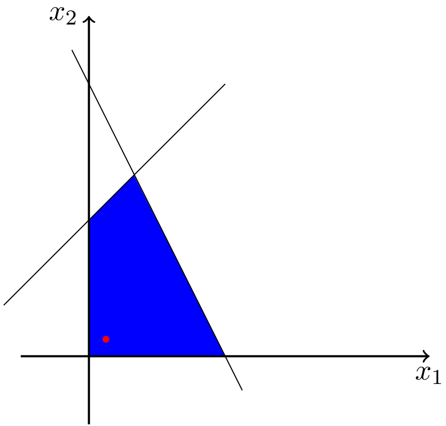
$$\begin{aligned} \text{maximize} \quad & {\color{red} x_1 + 2x_2} \\ \text{subject to} \quad & -x_1 + x_2 \le 2 \\ & 2x_1 + x_2 \le 4 \\ & x_1, x_2 \ge 0 \end{aligned}$$



# Interior point method (central path)

The iterates, instead, follow a path inside the feasible region.

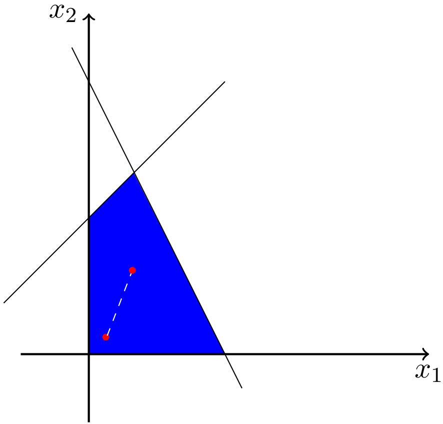
$$\begin{aligned} \text{maximize} \quad & {\color{red} x_1 + 2x_2} \\ \text{subject to} \quad & -x_1 + x_2 \le 2 \\ & 2x_1 + x_2 \le 4 \\ & x_1, x_2 \ge 0 \end{aligned}$$



# Interior point method (central path)

The iterates, instead, follow a path inside the feasible region.

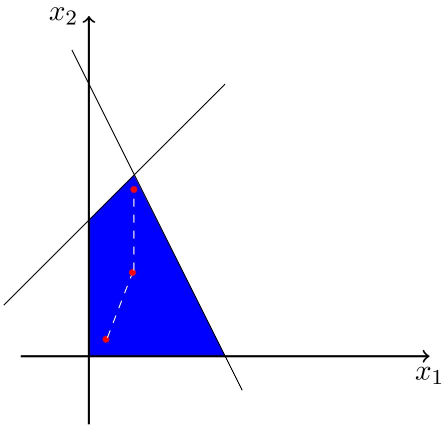
$$\begin{aligned} \text{maximize} \quad & x_1 + 2x_2 \\ \text{subject to} \quad & -x_1 + x_2 \le 2 \\ & 2x_1 + x_2 \le 4 \\ & x_1, x_2 \ge 0 \end{aligned}$$



# Interior point method (central path)

The iterates, instead, follow a path inside the feasible region.

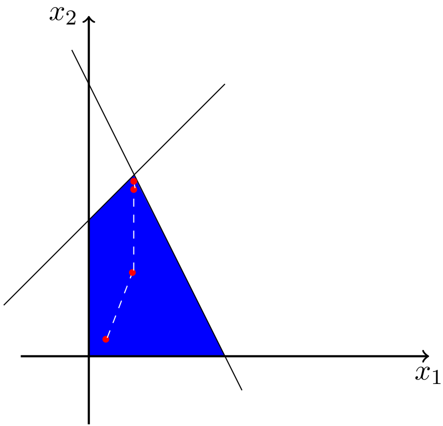
$$\begin{aligned} \text{maximize} \quad & x_1 + 2x_2 \\ \text{subject to} \quad & -x_1 + x_2 \leq 2 \\ & 2x_1 + x_2 \leq 4 \\ & x_1, x_2 \geq 0 \end{aligned}$$



# Interior point method (central path)

The iterates, instead, follow a path inside the feasible region.

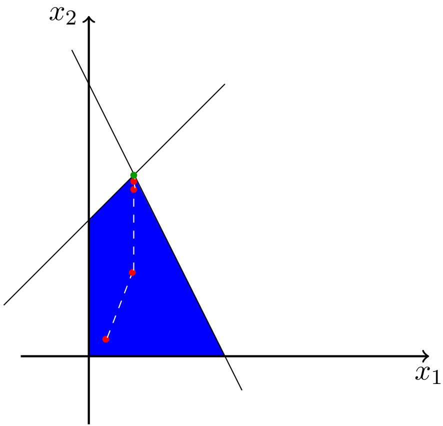
$$\begin{aligned} \text{maximize} \quad & x_1 + 2x_2 \\ \text{subject to} \quad & -x_1 + x_2 \le 2 \\ & 2x_1 + x_2 \le 4 \\ & x_1, x_2 \ge 0 \end{aligned}$$



# Interior point method (central path)

The iterates, instead, follow a path inside the feasible region.

$$\begin{aligned} \text{maximize} \quad & x_1 + 2x_2 \\ \text{subject to} \quad & -x_1 + x_2 \le 2 \\ & 2x_1 + x_2 \le 4 \\ & x_1, x_2 \ge 0 \end{aligned}$$



# LP polynomial-time complexity

Let $n$ be the number of variables.

# LP polynomial-time complexity

Let $n$ be the number of variables.

- The complexity bound for Karmarkar's algorithms is $\mathcal{O}(n)$ iterations and a total of $\mathcal{O}(n^{3.5})$ bit operations.

# LP polynomial-time complexity

Let $n$ be the number of variables.

- The complexity bound for Karmarkar's algorithms is $\mathcal{O}(n)$ iterations and a total of $\mathcal{O}(n^{3.5})$ bit operations.
- Later, the above bound on the # iterations was improved to $\mathcal{O}(n^{0.5})$.

# LP polynomial-time complexity

Let $n$ be the number of variables.

- The complexity bound for Karmarkar's algorithms is $\mathcal{O}(n)$ iterations and a total of $\mathcal{O}(n^{3.5})$ bit operations.
- Later, the above bound on the # iterations was improved to $\mathcal{O}(n^{0.5})$.
- The complexity of LP is thus $\mathcal{O}(n^3)$.

# The Kepler conjecture

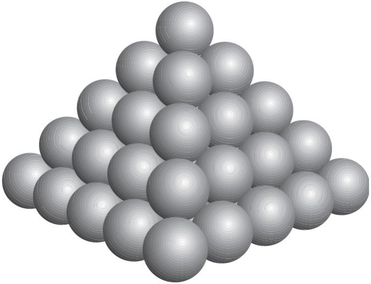
**Conjecture (Kepler)**

*No packing of congruent balls in $\mathbb{R}^3$ has density greater than face-centered cubic (FCC) or hexagonal-close packing (HCP).*

# The Kepler conjecture

**Conjecture (Kepler)**

*No packing of congruent balls in $\mathbb{R}^3$ has density greater than face-centered cubic (FCC) or hexagonal-close packing (HCP).*



FCC packing

The optimal density is $\frac{\pi}{\sqrt{18}} \approx 0.74$.

# A bit of history on Kepler's conjecture

- It was proposed by J. Kepler in 1611.

# A bit of history on Kepler's conjecture

- It was proposed by J. Kepler in 1611.
- In 1861, Gauss proved it for regular/periodic packing.

# A bit of history on Kepler's conjecture

- It was proposed by J. Kepler in 1611.
- In 1861, Gauss proved it for regular/periodic packing.
- In 1998, Thomas Hales, following L. F. Tóth in 1953, and assisted by his graduate student Samuel Ferguson, announced the proof!

# Outline of Hale's proof (1998)

- It assigns a graph to each possible packing.

# Outline of Hale's proof (1998)

- It assigns a graph to each possible packing.
- There are infinitely many such graphs, but up to isomorphism only a few thousands (L. F. Tóth).

# Outline of Hale's proof (1998)

- It assigns a graph to each possible packing.
- There are infinitely many such graphs, but up to isomorphism only a few thousands (L. F. Tóth).
- Then, using Linear Programming, none of the remaining possibilities has a packing denser than FCC/HPC.

# Outline of Hale's proof (1998)

- It assigns a graph to each possible packing.
- There are infinitely many such graphs, but up to isomorphism only a few thousands (L. F. Tóth).
- Then, using Linear Programming, none of the remaining possibilities has a packing denser than FCC/HPC.

In 2003, after a four-year review of twelve referees, Hale's proof was accepted for publication in Annals of Mathematics with $99\%$ certainty.

# Outline of Hale's proof (1998)

- It assigns a graph to each possible packing.
- There are infinitely many such graphs, but up to isomorphism only a few thousands (L. F. Tóth).
- Then, using Linear Programming, none of the remaining possibilities has a packing denser than FCC/HPC.

In 2003, after a four-year review of twelve referees, Hale's proof was accepted for publication in Annals of Mathematics with $99\%$ certainty.

The reviewers were not able to completely verify the computer programming part...

# Presentation outline

1. Importance of Optimization (illustration by Linear Programming)
2. Application of Optimization (illustration by Classification)
3. Classes of Optimization Problems
4. Type of Mathematics used in Optimization (illustration by Integrality, Convexity, Non-smooth Calculus)

# Support vector machine (SVM)

Suppose data points belonging to different classes are given. The goal is to determine to which class a new point belongs to.

# Support vector machine (SVM)

Suppose data points belonging to different classes are given. The goal is to determine to which class a new point belongs to.

SVM or Classification has many applications in engineering, computer science, bioinformatics, computational biology, etc..

# Support vector machine (SVM)

Suppose data points belonging to different classes are given. The goal is to determine to which class a new point belongs to.

SVM or Classification has many applications in engineering, computer science, bioinformatics, computational biology, etc..

## Example

- *(Medicine)* $n$ *patients with malignant tumors and* $m$ *patients with benign tumors. Based on the observed data, classify a new patient.*

# Support vector machine (SVM)

Suppose data points belonging to different classes are given. The goal is to determine to which class a new point belongs to.

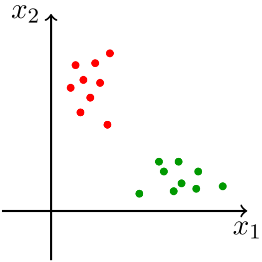
SVM or Classification has many applications in engineering, computer science, bioinformatics, computational biology, etc..

## Example

- *(Medicine) $n$ patients with malignant tumors and $m$ patients with benign tumors. Based on the observed data, classify a new patient.*
- *(Pattern Recognition) $n$ photos of cats and $m$ photos of dogs. Have a device telling us whether a new photo is a cat or a dog.*

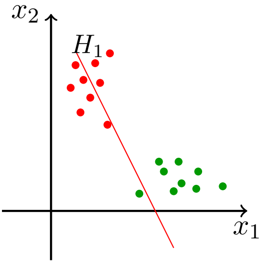
# SVM (linear classifier)

Consider the following two classes of data. If we can separate the data, then we can decide about a new case.



# SVM (linear classifier)

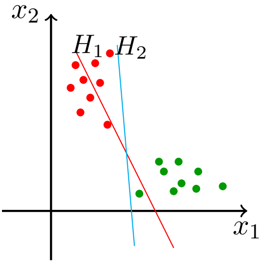
Consider the following two classes of data. If we can separate the data, then we can decide about a new case.




- $H_1$ does not separate the data.

# SVM (linear classifier)

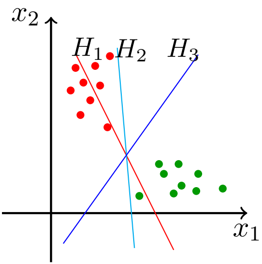
Consider the following two classes of data. If we can separate the data, then we can decide about a new case.




- $H_1$ does not separate the data.
- $H_2$ separates data with a small margin.

# SVM (linear classifier)

Consider the following two classes of data. If we can separate the data, then we can decide about a new case.




- $H_1$ does not separate the data.
- $H_2$ separates data with a small margin.
- $H_3$ separates data with a maximum margin.

# SVM (linear classifier)

Assume that the following linearly separable data points have been given:

$$\mathcal{D} = \{(x^i, y^i) \mid x^i \in \mathbb{R}^p, y^i \in \{-1, 1\}, i = 1, \cdots, n\},$$

where $y^i = 1$ and $y^i = -1$ represents the two classes that each $x^i$ belongs to.

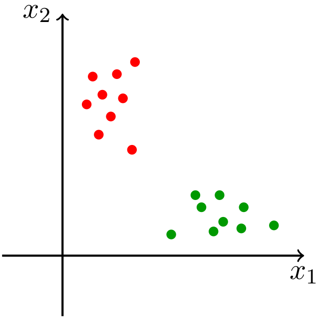
# SVM (linear classifier)

Assume that the following linearly separable data points have been given:

$$\mathcal{D} = \{(x^i, y^i) \mid x^i \in \mathbb{R}^p, y^i \in \{-1, 1\}, i = 1, \cdots, n\},$$

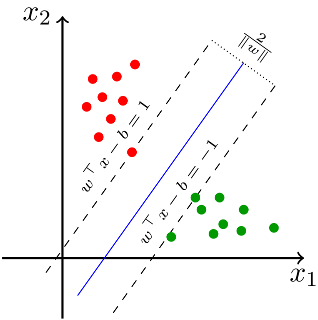
where $y^i = 1$ and $y^i = -1$ represents the two classes that each $x^i$ belongs to.

Below we have been given a set of $n = 18$ linearly separable data points in $\mathbb{R}^2$ for classification.



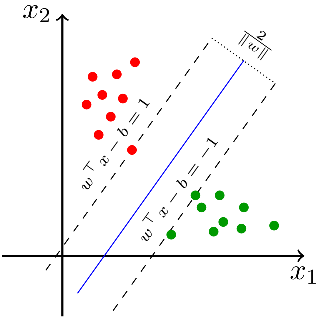
# SVM (linear classifier)

We need to maximize the distance between the hyperplanes $w^\top x - b = 1$ and $w^\top x - b = -1$ in a way that no data point falls between them:



# SVM (linear classifier)

We need to maximize the distance between the hyperplanes $w^\top x - b = 1$ and $w^\top x - b = -1$ in a way that no data point falls between them:




$$w^\top x - b \ \geq \ 1 \text{ when } x^i \text{ belongs to the class } y^i = 1$$

or

$$w^\top x' - b \ \leq \ -1 \text{ when } x^i \text{ belongs to the class } y^i = -1.$$

# SVM (linear classifier)

So, to maximize the distance we need to minimize $\|w\|$. Therefore, we have the following problem:

$$\begin{aligned} \underset{w,b}{\text{minimize}} \quad & \frac{1}{2}\|w\|^2 \\ \text{subject to} \quad & y^i(w^\top x - b) \geq 1, \text{ for any } i = 1, \ldots, n, \end{aligned}$$

which is a quadratic optimization/programming problem.

# Presentation outline

1. Importance of Optimization (illustration by Linear Programming)
2. Application of Optimization (illustration by Classification)
3. Classes of Optimization Problems
4. Type of Mathematics used in Optimization (illustration by Integrality, Convexity, Non-smooth Calculus)

# Classes of Optimization problems

$$\begin{array}{ll} \text{min} & f(x) \\ \text{subject to} & x \in \Omega \end{array}$$

# Classes of Optimization problems

$$\begin{array}{ll} \text{min} & f(x) \\ \text{subject to} & x \in \Omega \end{array}$$

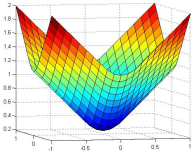
Linear Programming (LP)

Quadratic Programming (QP)

Conic Optimization ($f$ linear, $\Omega$ = Polyhedron LP $\cap$ Cone)

Convex Optimization ($f$ and $\Omega$ convex)

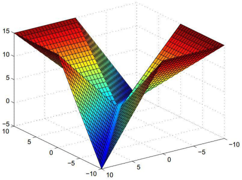
$$\begin{array}{ll} \text{min} & f(x) \\ \text{subject to} & x \in \Omega \end{array}$$




non-linear $f$

$\longrightarrow$ Nonlinear Programming.

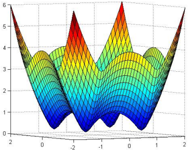
$$\begin{array}{ll} \text{min} & f(x) \\ \text{subject to} & x \in \Omega \end{array}$$




non-differentiable $f$

$\longrightarrow$ Non-differentiable Optimization.

$$\begin{array}{ll} \text{min} & f(x) \\ \text{subject to} & x \in \Omega \end{array}$$




non-convex $f$

$\longrightarrow$ Global Optimization.

$$\begin{array}{ll} \text{min} & f(x) \\ \text{subject to} & x \in \Omega \end{array}$$

$f$ can be a vectorial function

$$F(x) \ = \ (f_1(x), \ldots, f_m(x))$$

$\longrightarrow$ MultiObjective Optimization.

# Classes of Optimization problems

$$\begin{array}{ll} \text{min} & f(x) \\ \text{subject to} & x \in \Omega \end{array}$$

The variables can take only discrete values

$$x \in \mathbb{Z}^n$$

$\longrightarrow$ Combinatorial/Discrete Optimization.

## Algorithm

*A complicated formula to generate a sequence of points* $\{x_k\}$.

## Algorithm

*A complicated formula to generate a sequence of points* $\{x_k\}$.

When rigorous, one is able to prove convergence, e.g.,

$$x_k = x_* \quad \text{for some } k \qquad \text{or} \qquad \lim_{k\to+\infty} \nabla f(x_k) \;=\; 0.$$

## Algorithm

*A complicated formula to generate a sequence of points* $\{x_k\}$.

When rigorous, one is able to prove convergence, e.g.,

$$x_k = x_* \quad \text{for some } k \qquad \text{or} \qquad \lim_{k \to +\infty} \nabla f(x_k) \; = \; 0.$$

When not rigorous, it is a HEURISTIC.

# Presentation outline

1. Importance of Optimization (illustration by Linear Programming)

2. Application of Optimization (illustration by Classification)

3. Classes of Optimization Problems

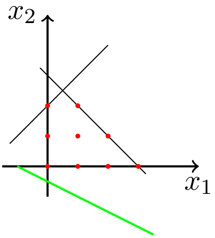
4. Type of Mathematics used in Optimization (illustration by Integrality, Convexity, Non-smooth Calculus)

maximize $x_1 + 2x_2$

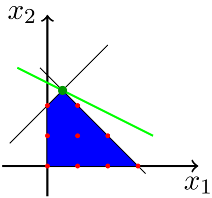
subject to $-x_1 + x_2 \leq 2$

$x_1 + x_2 \leq 3$

$x_1, x_2 \in \mathbb{Z}^+$



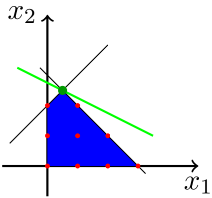
$$\begin{array}{ll} \text{maximize} & x_1 + 2x_2 \\ \text{subject to} & -x_1 + x_2 \le 2 \\ & x_1 + x_2 \le 3 \\ & x_1, x_2 \ge 0 \end{array}$$




The solution of the LP relaxation is not integer.

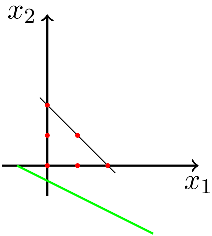
# Integer LP

$$\begin{aligned} \text{maximize} \quad & x_1 + 2x_2 \\ \text{subject to} \quad & -x_1 + x_2 \leq 2 \\ & x_1 + x_2 \leq 3 \\ & x_1, x_2 \geq 0 \end{aligned}$$

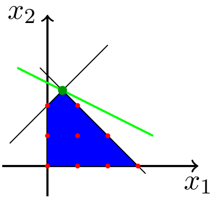



The solution of the LP relaxation is not integer.

$$\begin{aligned} \text{maximize} \quad & x_1 + 2x_2 \\ \text{subject to} \quad & x_1 + x_2 \leq 2 \\ & x_1, x_2 \in \mathbb{Z}^+ \end{aligned}$$

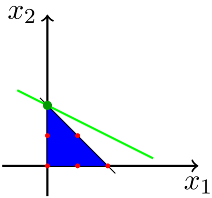


$$\begin{aligned} \text{maximize} \quad & x_1 + 2x_2 \\ \text{subject to} \quad & -x_1 + x_2 \leq 2 \\ & x_1 + x_2 \leq 3 \\ & x_1, x_2 \geq 0 \end{aligned}$$




The solution of the LP relaxation is not integer.

$$\begin{aligned} \text{maximize} \quad & x_1 + 2x_2 \\ \text{subject to} \quad & x_1 + x_2 \leq 2 \\ & x_1, x_2 \geq 0 \end{aligned}$$




The solution of the LP relaxation is integer.

# Total unimodularity

A matrix is totally unimodular (TU) if the determinant of every square submatrix has value -1, 0, or 1.

# Total unimodularity

A matrix is totally unimodular (TU) if the determinant of every square submatrix has value -1, 0, or 1.

- The matrix $\begin{bmatrix} 1 & 0 & 1 \\ 1 & -1 & 0 \end{bmatrix}$ is TU.

# Total unimodularity

A matrix is totally unimodular (TU) if the determinant of every square submatrix has value -1, 0, or 1.

- The matrix $\begin{bmatrix} 1 & 0 & 1 \\ 1 & -1 & 0 \end{bmatrix}$ is TU.
- The matrix $\begin{bmatrix} 1 & 0 & 1 \\ 1 & -1 & -1 \end{bmatrix}$ is not TU.

# Total unimodularity

A matrix is totally unimodular (TU) if the determinant of every square submatrix has value -1, 0, or 1.

- The matrix $\begin{bmatrix} 1 & 0 & 1 \\ 1 & -1 & 0 \end{bmatrix}$ is TU.
- The matrix $\begin{bmatrix} 1 & 0 & 1 \\ 1 & -1 & -1 \end{bmatrix}$ is not TU.

When $A$ is TU and $b$ is an integer vector, then every vertex of

$$\{x \in \mathbb{R}^n : \; Ax \leq b, \; x \geq 0\}$$

# Total unimodularity

A matrix is totally unimodular (TU) if the determinant of every square submatrix has value -1, 0, or 1.

- The matrix $\begin{bmatrix} 1 & 0 & 1 \\ 1 & -1 & 0 \end{bmatrix}$ is TU.
- The matrix $\begin{bmatrix} 1 & 0 & 1 \\ 1 & -1 & -1 \end{bmatrix}$ is not TU.

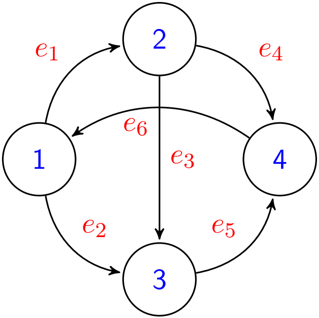
When $A$ is TU and $b$ is an integer vector, then every vertex of

$$\{x \in \mathbb{R}^n : \ Ax \leq b, \ x \geq 0\}$$

is integer (i.e., the polyhedron is integral).

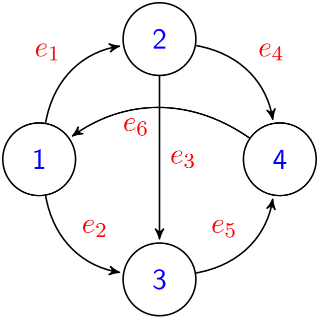
# A TU example

The incidence matrix of



# A TU example

The incidence matrix of




is

$$A = \begin{bmatrix} -1 & -1 & 0 & 0 & 0 & +1 \\ +1 & 0 & -1 & -1 & 0 & 0 \\ 0 & +1 & +1 & 0 & -1 & 0 \\ 0 & 0 & 0 & +1 & +1 & -1 \end{bmatrix}.$$

# A TU example

The incidence matrix of a (directed) graph is TU.

# A TU example

The incidence matrix of a (directed) graph is TU.

Thus, the shortest path problem between nodes $s$ and $t$

# A TU example

The incidence matrix of a (directed) graph is TU.

Thus, the shortest path problem between nodes $s$ and $t$

$$\text{minimize} \quad \sum_{u\to v} c_{u\to v} x_{u\to v}$$

$$\text{subject to} \quad \sum_{u} x_{u\to v} - \sum_{w} x_{v\to w} = \begin{cases} 0 & \text{if } v \neq s,t \\ -1 & \text{if } v = s \\ 1 & \text{if } v = t \end{cases}$$

$$x_{u\to v} \geq 0$$

$$x_{u\to v} \in \{0,1\}$$

# A TU example

The incidence matrix of a (directed) graph is TU.

Thus, the shortest path problem between nodes $s$ and $t$

$$\begin{aligned}
\text{minimize} \quad & \sum_{u\to v} c_{u\to v} x_{u\to v} \\
\text{subject to} \quad & \sum_{u} x_{u\to v} - \sum_{w} x_{v\to w} = \begin{cases} 0 & \text{if } v \neq s,t \\ -1 & \text{if } v = s \\ 1 & \text{if } v = t \end{cases} \\
& x_{u\to v} \geq 0 \\
& x_{u\to v} \in \{0,1\}
\end{aligned}$$

can be solved polynomially (since the solution of the LP relaxation is integer).

# Convexity (local/global and uniqueness)

Local/Global:

# Convexity (local/global and uniqueness)

Local/Global:

If $f$ is convex in $S$ convex, then every local minimizer of $f$ is global.

# Convexity (local/global and uniqueness)

Local/Global:

If $f$ is convex in $S$ convex, then every local minimizer of $f$ is global.

Uniqueness:

# Convexity (local/global and uniqueness)

Local/Global:

If $f$ is convex in $S$ convex, then every local minimizer of $f$ is global.

Uniqueness:

If $f$ is strictly convex in $S$ convex and $\exists$ a global minimizer, it is unique.

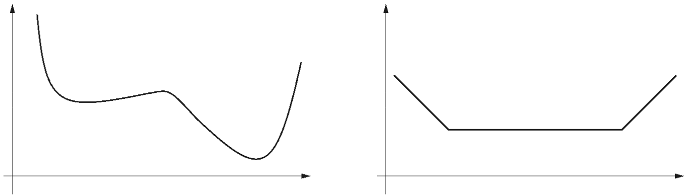
# Convexity (local/global and uniqueness)

Local/Global:

If $f$ is convex in $S$ convex, then every local minimizer of $f$ is global.

Uniqueness:

If $f$ is strictly convex in $S$ convex and $\exists$ a global minimizer, it is unique.



(a) $f$ not convex

(b) $f$ not strictly convex

# Convexity (existence)

Existence (general):

# Convexity (existence)

Existence (general):

If $f$ is continuous in $\Omega$ bounded and closed (i.e., compact in finite dimensions), then $\exists$ minimizer (and maximizer) — Weirstrass Theorem.

# Convexity (existence)

Existence (general):

If $f$ is continuous in $\Omega$ bounded and closed (i.e., compact in finite dimensions), then $\exists$ minimizer (and maximizer) — Weirstrass Theorem.

Existence (using convexity):

# Convexity (existence)

Existence (general):

If $f$ is continuous in $\Omega$ bounded and closed (i.e., compact in finite dimensions), then $\exists$ minimizer (and maximizer) — Weirstrass Theorem.

Existence (using convexity):

If $f$ is continuous and strongly/uniformly convex in $S$ convex and closed, then $\exists$ a unique minimizer.

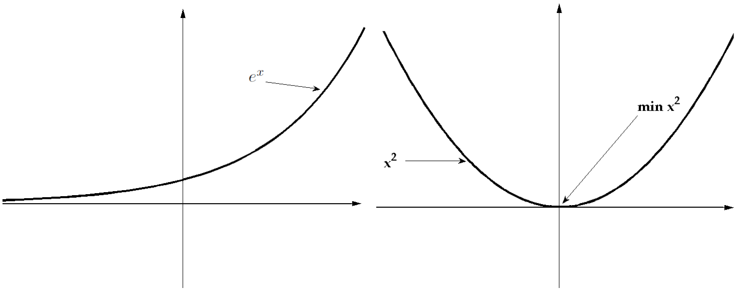
# Convexity (existence)

Existence (general):

If $f$ is continuous in $\Omega$ bounded and closed (i.e., compact in finite dimensions), then $\exists$ minimizer (and maximizer) — Weirstrass Theorem.

Existence (using convexity):

If $f$ is continuous and strongly/uniformly convex in $S$ convex and closed, then $\exists$ a unique minimizer.




(a) no minimizer (b) $\exists^1$ minimizer

# Non-smooth calculus (directional derivative)

## Definition

*For $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ Lipschitz continuous near $x$, the Clarke generalized directional derivative is:*

$$f^\circ(x; d) = \limsup_{y \to x \; t \downarrow 0} \frac{f(y + td) - f(y)}{t}.$$

# Non-smooth calculus (directional derivative)

**Definition**

*For $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ Lipschitz continuous near $x$, the Clarke generalized directional derivative is:*

$$f^\circ(x;d) = \limsup_{y \to x \; t \downarrow 0} \frac{f(y+td) - f(y)}{t}.$$

What does this $\limsup$ exactly mean?

$$\limsup_{y \to x \; t \downarrow 0} \frac{f(y+td) - f(y)}{t} \;=\; \lim_{\epsilon \downarrow 0} \; \sup_{\|y-x\| \le \epsilon, 0 < t \le \epsilon} \left\{ \frac{f(y+td) - f(y)}{t} \right\}.$$

# Non-smooth calculus (subdifferential)

## Definition

*Let $f$ be Lipschitz cont. near $x$. The Clarke subdifferential is given by:*

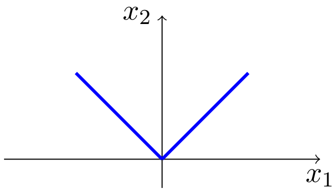
$$\partial f(x) \ = \ \{s \in \mathbb{R}^n : f^\circ(x; v) \geq \langle v, s \rangle, \ \forall v \in \mathbb{R}^n\}.$$

# Non-smooth calculus (subdifferential)

## Definition

*Let $f$ be Lipschitz cont. near $x$. The Clarke subdifferential is given by:*

$$\partial f(x) \; = \; \{s \in \mathbb{R}^n : f^\circ(x; v) \geq \langle v, s \rangle, \; \forall v \in \mathbb{R}^n\}.$$

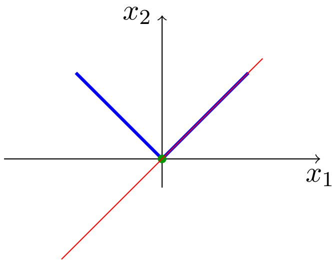


# Non-smooth calculus (subdifferential)

## Definition

*Let $f$ be Lipschitz cont. near $x$. The Clarke subdifferential is given by:*

$$\partial f(x) \ = \ \{s \in \mathbb{R}^n : f^\circ(x;v) \geq \langle v, s\rangle, \ \forall v \in \mathbb{R}^n\}.$$

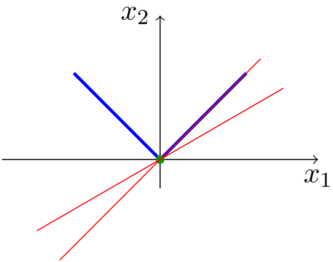


# Non-smooth calculus (subdifferential)

## Definition

*Let $f$ be Lipschitz cont. near $x$. The Clarke subdifferential is given by:*

$$\partial f(x) \; = \; \{s \in \mathbb{R}^n : f^\circ(x; v) \geq \langle v, s\rangle, \; \forall v \in \mathbb{R}^n\}.$$

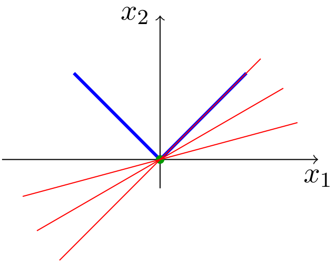


# Non-smooth calculus (subdifferential)

## Definition

*Let $f$ be Lipschitz cont. near $x$. The Clarke subdifferential is given by:*

$$\partial f(x) \;=\; \{s \in \mathbb{R}^n : f^\circ(x;v) \geq \langle v, s\rangle,\ \forall v \in \mathbb{R}^n\}.$$

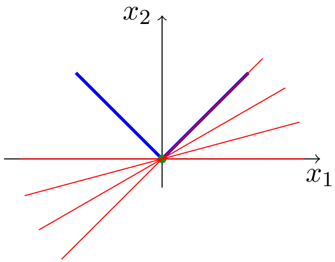


# Non-smooth calculus (subdifferential)

## Definition

*Let $f$ be Lipschitz cont. near $x$. The Clarke subdifferential is given by:*

$$\partial f(x) \;=\; \{s \in \mathbb{R}^n : f^\circ(x;v) \geq \langle v, s\rangle,\; \forall v \in \mathbb{R}^n\}.$$

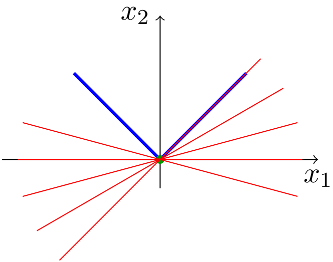


# Non-smooth calculus (subdifferential)

## Definition

*Let $f$ be Lipschitz cont. near $x$. The Clarke subdifferential is given by:*

$$\partial f(x) \;=\; \{s \in \mathbb{R}^n : f^\circ(x; v) \geq \langle v, s \rangle,\; \forall v \in \mathbb{R}^n\}.$$

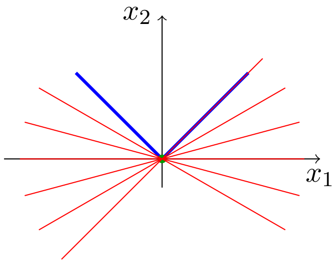


# Non-smooth calculus (subdifferential)

## Definition

*Let $f$ be Lipschitz cont. near $x$. The Clarke subdifferential is given by:*

$$\partial f(x) \;=\; \{s \in \mathbb{R}^n : f^{\circ}(x; v) \geq \langle v, s \rangle,\ \forall v \in \mathbb{R}^n\}.$$

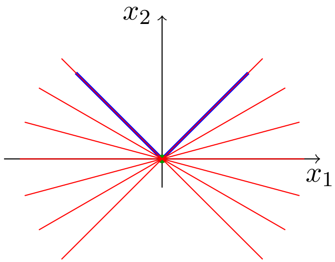


# Non-smooth calculus (subdifferential)

## Definition

*Let $f$ be Lipschitz cont. near $x$. The Clarke subdifferential is given by:*

$$\partial f(x) \;=\; \{s \in \mathbb{R}^n : f^{\circ}(x;v) \geq \langle v, s\rangle,\; \forall v \in \mathbb{R}^n\}.$$




At the origin, $\partial f(0) = [-1, 1]$.

# First order stationarity

If $f$ is increasing from $x$, along $d$, then

$$f^{\circ}(x;d) \;\geq\; 0.$$

# First order stationarity

If $f$ is increasing from $x$, along $d$, then

$$f^\circ(x; d) \;\geq\; 0.$$

**First order stationarity (Clarke)**

*If $x_*$ is a local minimizer, $f^\circ(x_*; v) \geq 0,\ \forall v \in \mathbb{R}^n$ or, equivalently, $0 \in \partial f(x_*)$.*

# Example of use of non-smooth calculus in Optimization

Suppose $x_k \to x_*$, $\alpha_k \to 0 \in \mathbb{R}$, and $d_k \to d$ for some infinite sequence $K$. Then:

# Example of use of non-smooth calculus in Optimization

Suppose $x_k \to x_*$, $\alpha_k \to 0 \in \mathbb{R}$, and $d_k \to d$ for some infinite sequence $K$. Then:

$$f^\circ(x_*; d) \quad = \quad \limsup_{y \to x_*, t \downarrow 0} \frac{f(y + td) - f(y)}{t}$$

# Example of use of non-smooth calculus in Optimization

Suppose $x_k \to x_*$, $\alpha_k \to 0 \in \mathbb{R}$, and $d_k \to d$ for some infinite sequence $K$. Then:

$$
\begin{aligned}
f^\circ(x_*; d) &= \limsup_{y \to x_*, t \downarrow 0} \frac{f(y + td) - f(y)}{t} \\
&= \limsup_{k \in K} \left\{ \frac{f(x_k + \alpha_k d_k) - f(x_k)}{\alpha_k} \right\}
\end{aligned}
$$

# Example of use of non-smooth calculus in Optimization

Suppose $x_k \to x_*$, $\alpha_k \to 0 \in \mathbb{R}$, and $d_k \to d$ for some infinite sequence $K$. Then:

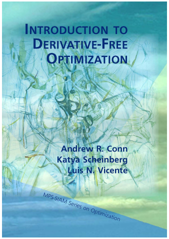
$$\begin{aligned} f^\circ(x_*; d) &= \limsup_{y \to x_*, t \downarrow 0} \frac{f(y+td) - f(y)}{t} \\ &= \limsup_{k \in K} \left\{ \frac{f(x_k + \alpha_k d_k) - f(x_k)}{\alpha_k} \right\} \\ &\geq 0 \end{aligned}$$

... if an Optimization algorithm can generate a subsequence of $K$ such that

$$f(x_k + \alpha_k d_k) \geq f(x_k).$$

# Luis Nunes Vicente

http://www.mat.uc.pt/~lnv




Introduction to Derivative-Free Optimization

My research interests include the development and analysis of numerical methods for large-scale nonlinear programming, sparse optimization, PDE constrained optimization problems, and derivative-free optimization problems, and applications in computational sciences, engineering, and finance.