Optimization: a Ride on the Carousel (with an Eye to Energy)

Antonio Frangioni

Dipartimento di Informatica, Università di Pisa

3rd Winter School for PhD students on

FLUID MACHINES AND ENERGY SYSTEMS

School of Engineering

University of Pisa

Pisa — March 27th, 2019

Beware Mathematicians When They Seem to Speak Simple

Black-box Optimization

PDE-Constrained Optimization

NonLinear Nonconvex Problems

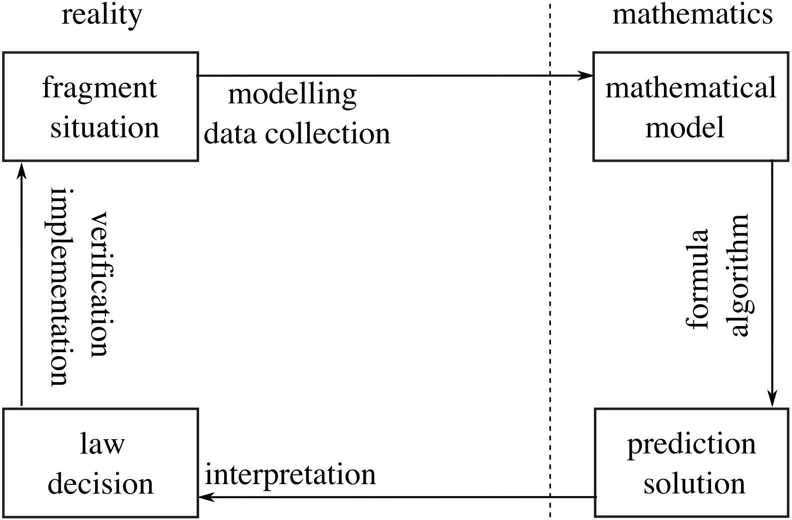
Mixed-Integer Convex (Linear) Problems

An Aside: Multiple Objectives

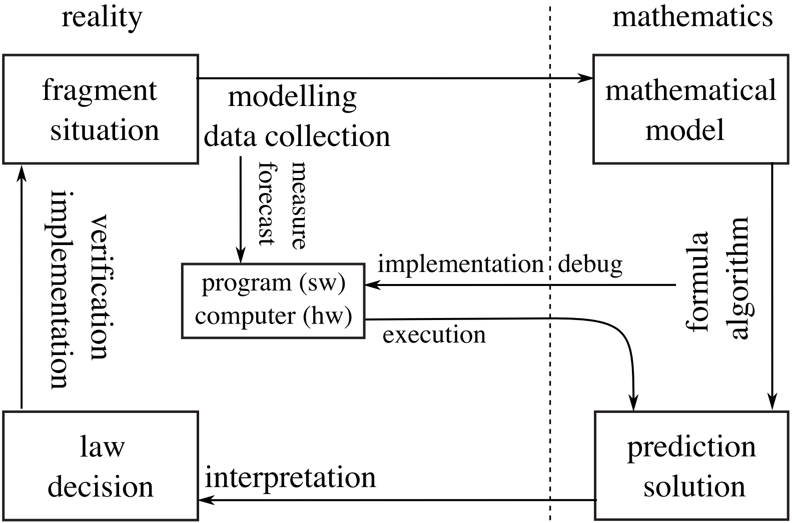
Do Not Underestimate Mixed-Integer Linear (Convex) Problems

Conclusions: Don't Be Afraid Of Optimization

- Everything you do, there always is at least one scarce resource (time, material, energy, money, manpower, data, knowledge, . . .)
- If you want to do more, there are basically two ways:
  - get more resources (slaughter some more of your neighbours, pollute some more the environment, . . .)
  - make better use of the resources you have
- Technology = learning to better exploit available resources
- This requires understanding of how the world works
- Understanding is almost invariably best expressed via mathematics (although not completely clear as to why this is the case)



- The fundamental cycle



- The fundamental cycle and its implementation

- Descriptive model: tells how the world (supposedly) is
- Prescriptive model: tells how the world (supposedly) should be
  a.k.a. optimization problem:

$$(P) \qquad f_* = \min \left\{ f(x) \; : \; x \in X \right\}$$

  - arbitrary set X = feasible region of possible choices x
  - typically X specified by $G \supset X$ (ground set) + constraints dictating required properties of feasible solutions $x \in X$
    [$\Longrightarrow x \in G \setminus X$ = unfeasible solution (??)]
  - $f : X \to Y$ objective function mapping preferences (cost)
  - optimal value $f_* \leq f(x) \forall x \in X$, $\forall v > f_* \, \exists x \in X$ s.t. $f(x) < v$
  - we want optimal solution: $x_* \in X$ s.t. $f(x_*) = f_*$
- Everything looks pretty straightforward

- Descriptive model: tells how the world (supposedly) is

- Prescriptive model: tells how the world (supposedly) should be
  a.k.a. optimization problem:

$$(P) \qquad f_* = \min \left\{ f(x) \ : \ x \in X \right\}$$

  - arbitrary set X = feasible region of possible choices x
  - typically X specified by $G \supset X$ (ground set) + constraints dictating required properties of feasible solutions $x \in X$
    [$\Longrightarrow x \in G \setminus X$ = unfeasible solution (??)]
  - $f : X \to Y$ objective function mapping preferences (cost)
  - optimal value $f_* \leq f(x) \forall x \in X$, $\forall v > f_* \, \exists x \in X$ s.t. $f(x) < v$
  - we want optimal solution: $x_* \in X$ s.t. $f(x_*) = f_*$

- Everything looks pretty straightforward ... or is it?

- “Bad case” I: $X = \emptyset$ (“empty”)

  $$\min\{\, x \;:\; x \in \mathbb{R} \wedge x \leq -1 \wedge x \geq 1 \,\}$$

  there just is no solution (which may be important to know)

- “Bad case” II: $\forall M \, \exists x_M \in X$ s.t. $f(x_M) \leq M$ (“unbounded [below]”)

  $$\min\{\, x \;:\; x \in \mathbb{R} \wedge x \leq 0 \,\}$$

  there are solutions as good as you like (which may be important to know)

- Not really bad cases, just things that can happen

- Solving an optimization problem actually three different things:
  - Finding x_* and proving it is optimal (how??)
  - Proving $X = \emptyset$ (how??)
  - Constructively prove $\forall M \, \exists x_M \in X$ s.t. $f(x_M) \leq M$ (how??)

- Often OK to find approximately optimal $\bar{x}$ and prove it (how??)

  $$f(\bar{x}) - f_* \leq \varepsilon \text{ (absolute)} \quad \text{or} \quad (\, f(\bar{x}) - f_* \,)/|\, f_* \,| \leq \varepsilon \text{ (relative)}$$

- Let's just stick to nonempty and bounded $X \implies \exists x_*$

- "Bad case" I: $X = \emptyset$ ("empty")

  $$\min\{\, x \;:\; x \in \mathbb{R} \wedge x \leq -1 \wedge x \geq 1 \,\}$$

  there just is no solution (which may be important to know)

- "Bad case" II: $\forall M \, \exists x_M \in X$ s.t. $f(x_M) \leq M$ ("unbounded [below]")

  $$\min\{\, x \;:\; x \in \mathbb{R} \wedge x \leq 0 \,\}$$

  there are solutions as good as you like (which may be important to know)

- Not really bad cases, just things that can happen

- Solving an optimization problem actually three different things:
  - Finding x_* and proving it is optimal (how??)
  - Proving $X = \emptyset$ (how??)
  - Constructively prove $\forall M \, \exists x_M \in X$ s.t. $f(x_M) \leq M$ (how??)

- Often OK to find approximately optimal $\bar{x}$ and prove it (how??)

  $$f(\bar{x}) - f_* \leq \varepsilon \text{ (absolute)} \quad \text{or} \quad (\, f(\bar{x}) - f_* \,)/|\, f_* \,| \leq \varepsilon \text{ (relative)}$$

- Let's just stick to nonempty and bounded $X \Longrightarrow \exists x_*$ ... or does it?

- Things can be worse: not empty, not unbounded, but no x_* either:
  - $\min\{\, x \;:\; x \in \mathbb{R} \wedge x > 0 \,\}$ (“bad” X)
  - $\min\{\, 1 / x \;:\; x \in \mathbb{R} \wedge x > 0 \,\}$ (“bad” f and X)
  - $\min \left\{ f(x) = \left\{ \begin{array}{ll} x & \text{if } x > 0 \\ 1 & \text{if } x = 0 \end{array} \right. \;:\; x \in [\, 0 \,,\, 1 \,] \right\}$ (“bad” f)
- Still $\exists$ approximately optimal solutions $\forall\, \varepsilon > 0$, good enough for us
- Has to ensure somehow things don’t go awry
  (continuity, closeness, boundedness, …)
- Then optimizaton problems are simple objects

- Things can be worse: not empty, not unbounded, but no x_* either:
  - $\min\{\, x \;:\; x \in \mathbb{R} \wedge x > 0 \,\}$ ("bad" X)
  - $\min\{\, 1 / x \;:\; x \in \mathbb{R} \wedge x > 0 \,\}$ ("bad" f and X)
  - $\min \left\{ f(x) = \begin{cases} x & \text{if } x > 0 \\ 1 & \text{if } x = 0 \end{cases} \;:\; x \in [\,0\,,\,1\,] \right\}$ ("bad" f)
- Still $\exists$ approximately optimal solutions $\forall\, \varepsilon > 0$, good enough for us
- Has to ensure somehow things don't go awry (continuity, closeness, boundedness, ...)
- Then optimizaton problems are simple objects ... or are they?

A Set Need Be Simple, Right?

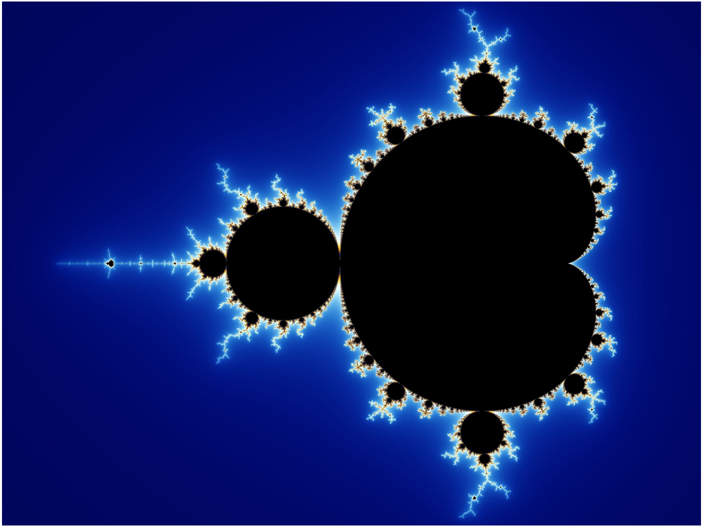
- . . . if it is defined by a simple mathematical relationship, right?

A Set Need Be Simple, Right?

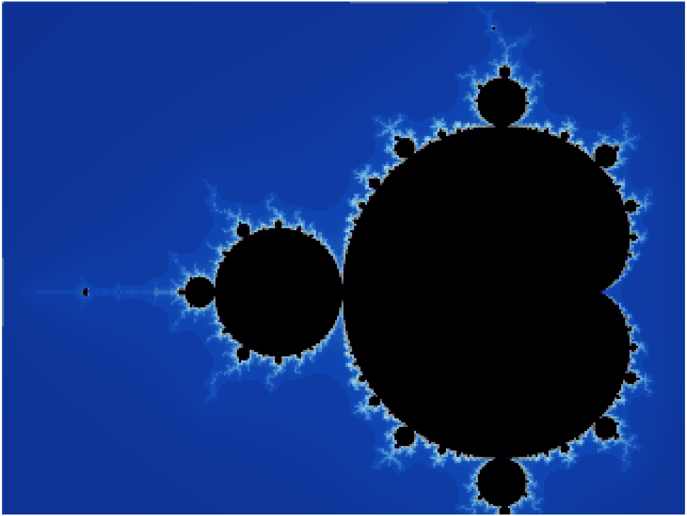
- ...if it is defined by a simple mathematical relationship, right?
- Sure! try $M = \{\, c \in \mathbb{C} : \sup_n \{\, | x_{n+1} \leftarrow x_n^2 + c \,| \,\} < \infty \,\}$ $\quad (x_0 = 0)$

A Set Need Be Simple, Right?

- …if it is defined by a simple mathematical relationship, right?
- Sure! try $M = \{\, c \in \mathbb{C} : \sup_n \{\, |\, x_{n+1} \leftarrow x_n^2 + c \,|\, \} < \infty \,\}$ $\quad (x_0 = 0)$



- ...if it is defined by a simple mathematical relationship, right?
- Sure! try $M = \{\, c \in \mathbb{C} : \sup_n \{\, |\, x_{n+1} \leftarrow x_n^2 + c \,|\, \} < \infty \,\} \quad (x_0 = 0)$



- $X \subset G \equiv$ (indicator) function $\imath_X : G \to \{ 0 , \infty \}$
- $x \in X \equiv \imath_X(x) \leq 0$ (constraint)
- All the difficulty lies in computing function values:

$$(P) \equiv \min \{ f(x) + \imath_X(x) \}$$

  the objective can take up all the complication of constraints

- Vice-versa also true: objective can always be linear

$$(P) \equiv \min \{ v \ : \ x \in X \ , \ v \geq f(x) \}$$

- And at least I can compute $f(x) / \imath_X(x)$, right? How hard can that be?
- Functions can be demonstrably impossible to compute $\Longrightarrow$ (P) demonstrably impossible to solve
- Even if not impossible, computing a function can be very hard

Beware Mathematicians When They Seem to Speak Simple

Black-box Optimization

PDE-Constrained Optimization

NonLinear Nonconvex Problems

Mixed-Integer Convex (Linear) Problems

An Aside: Multiple Objectives

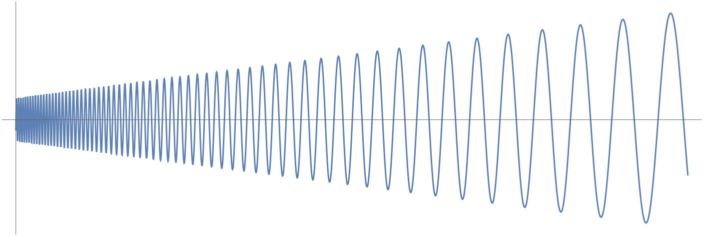
Do Not Underestimate Mixed-Integer Linear (Convex) Problems

Conclusions: Don't Be Afraid Of Optimization

- (P) where $f(\cdot)$ / $\imath_X(\cdot)$ are "just any function" $\equiv$ complex mathematical model with no closed formulæ (most of them):
  - numerical integration
  - systems of PDEs
  - electromagnetic propagation models (ray-tracing, ...)
  - heat propagation models (heating/cooling of buildings, ...)
  - systems with complex management procedures (storage/plant design with route/machine optimization ...)
  - systems with stochastic components (+ possibly complex management) (queues in ERs, users of cellular networks, ...)
- A.k.a. simulation-based optimization: the system can only be numerically simulated as opposed to algebraically described
- Computation of $f(x)$ / $\imath_X(x)$ costly (can do few 100s/1000s of them)
- No information about the behaviour of $f(\cdot)$ "close" to x

- Typically require bound constraints: w.l.o.g. $X \subseteq [0, 1]^n$ (hence $X = [0, 1]^n$, other constraints "hidden" in $f(\cdot)$)
- Basically only (clever) "shotgun approach": fire enough rounds and eventually a good solution happens
- Good playground for population-based approaches (genetic algorithms, particle swarm, ...)
- Any other standard search (simulated annealing, taboo search, GRASP, variable-neighbourhood search, ...)
- Better idea: construct a model of $f(\cdot)$ out of past iterates to drive the search (regression, kriging, radial-basis functions, SVR, ML, ...)
- Bad news: none of these can possibly work efficiently (in theory)

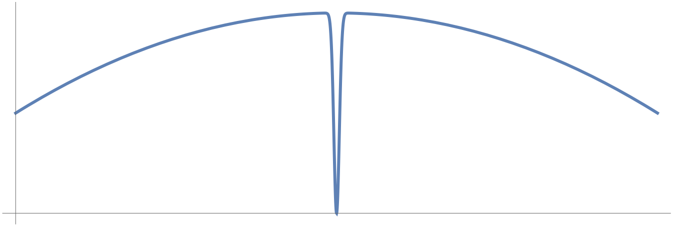
- If $f(\cdot)$ "swings wildly", things can be arbitrarily bad



- $f : \mathbb{R}^n \to \mathbb{R}$ (globally) Lipschitz continuous with constant L ($|f(x) - f(y)| \leq L||x - y|| \implies$ not too wild swings $\implies$ continuous)
- For each algorithm $\exists\, f(\cdot)$ s.t. finding ε-optimal solution requires $\geq O(L/\varepsilon)^n$ evaluations — that's very bad
- No free lunch theorem says "all algorithms equally bad"
- In practice is not as bad, but cost indeed grows very rapidly with n
- $n \approx 10 - 100$ if $f(\cdot)$ very costly, perhaps $n \approx 1000$ if not too costly

- $f(x)$ may be a random process:
  average performance computed via Montecarlo out of simulations
- Many examples:
  - behaviour of users
  - impact of weather on energy production/consumption
  - errors in measurement/impurity of materials ...
- Interesting tidbit: almost all approaches are inherently randomized
  (if you don't know anything, you may as well throw dices)
- Good part: can be trivially parallelized (as all Montecarlo do)
- Bad part: many runs = costly to compute average with high accuracy
- Intuitively, high accuracy only needed close to x_*
- But how do I tell if I'm close x_*? And which x_*?

- In a nutshell: if everything goes very, very well
  - you don't have many parameters (n in the few tens, ...)
  - you don't really need the best solution, a good one is OK
  - you have a lot of time and/or a supercomputer at hand
  - f is "nice enough": Lipschitz continuous, no isolated local minima, ...



- Good news: plenty of general-purpose black-box solvers, simple to use
- Bad news: difficult to choose/tune, none will ever scale to large-size
- In many cases, it is just what is needed
- Can we do better? Yes, we can if we have more structure

Beware Mathematicians When They Seem to Speak Simple

Black-box Optimization

PDE-Constrained Optimization

NonLinear Nonconvex Problems

Mixed-Integer Convex (Linear) Problems

An Aside: Multiple Objectives

Do Not Underestimate Mixed-Integer Linear (Convex) Problems

Conclusions: Don't Be Afraid Of Optimization

- Fundamental concept: if you know the structure of $f(\cdot)/X$, exploit it
- Very important structure: Partial Differential Equations
- Model disparate phenomena as such as:
  - sound, heat, diffusion
  - electromagnetism (Maxwell's equations)
  - fluid dynamics (Navier–Stokes equations)
  - elasticity, ...
- Countless many applications:
  - weather forecast, ocean currents, pollution diffusion, ...
  - flows in pipes (water, gas, blood, ...)
  - air flow (airplane wing, car, wind turbine, ...)
  - behaviour of complex materials/objects (buildings, seismic models, ...)
- Optimal design/operation of many systems has PDE-defined $f(\cdot)/X$: PDE-Constrained Optimization (PDE-CO) problem

- General form of the problem:

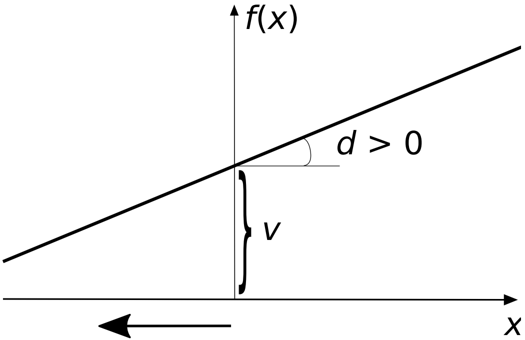
$$\text{(PDE-CO)} \qquad \min \left\{ \, \ell(c,s) \; : \; \mathcal{H}(c,s) = 0 \;\; , \;\; \mathcal{G}(c,s) \geq 0 \, \right\}$$

  $x = [\, c \, , \, s \,]$, explicit description of X:
  - $s =$ state (pressure/velocity of air, force in material, . . .)
  - $c =$ controls (shape of wing/blade, position of actuators, . . .)
  - $\ell(c,s) =$ measure of function $\Longrightarrow$ typically involves integrals
  - $\mathcal{H}(c,s) =$ PDE constraints (Navier–Stokes equations, . . .)
  - $\mathcal{G}(c,s) =$ "other" algebraic constraints (min/max size/position, . . .)
- each $s_i : \mathbb{R}^k \to \mathbb{R}$ a function: $X \subset \mathbb{F}^n$
- often k small-ish: 2D/3D coordinates, fields, time (optimal control)
- controls may be functions or "simple" reals ($\equiv$ linear functions)
- $\mathbb{F}^n$ is a whole lot bigger than even $\mathbb{R}^n$ (all functions vs. linear ones): Banach space, infinite-dimensional while $\mathbb{R}^n$ has finite dimension n
- What did I gain from knowing ℓ, $\mathcal{H}$, $\mathcal{G}$?

- (the) "Space ($\mathbb{F}^n$) is big. Really big. You just won't believe how vastly, hugely, mind-bogglingly big it is."

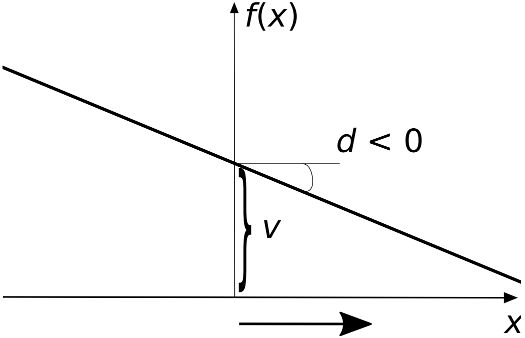
- (the) "Space ($\mathbb{F}^n$) is big. Really big. You just won't believe how vastly, hugely, mind-bogglingly big it is." Which way is x_*?

- (the) "Space ($\mathbb{F}^n$) is big. Really big. You just won't believe how vastly, hugely, mind-bogglingly big it is." Which way is x_*?



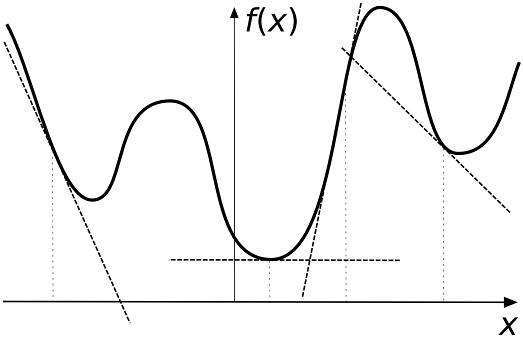
- $f(x) := dx + v : \mathbb{R} \to \mathbb{R}$ (linear) easy: always left if $d > 0$,

- (the) “Space ($\mathbb{F}^n$) is big. Really big. You just won’t believe how vastly, hugely, mind-bogglingly big it is.” Which way is x_*?



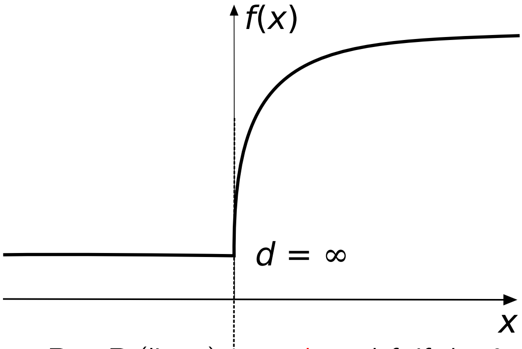
- $f(x) := dx + v : \mathbb{R} \to \mathbb{R}$ (linear) easy: always left if $d > 0$, right if $d < 0$

- (the) "Space ($\mathbb{F}^n$) is big. Really big. You just won't believe how vastly, hugely, mind-bogglingly big it is." Which way is x_*?



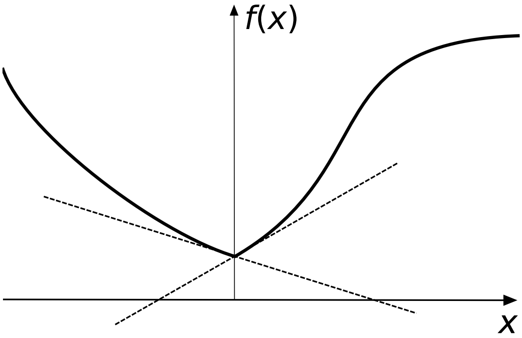
- $f(x) := dx + v : \mathbb{R} \to \mathbb{R}$ (linear) easy: always left if $d > 0$, right if $d < 0$
- Obvious idea: use the linear function that best locally approximates f
- Trusty old derivative $d = f'(x) = \lim_{t \to 0}[\, f(x+t) - f(x)\,]/t$ (putting a lot under the carpet even in $\mathbb{R}^n$, not to mention $\mathbb{F}^n$)

- (the) "Space ($\mathbb{F}^n$) is big. Really big. You just won't believe how vastly, hugely, mind-bogglingly big it is." Which way is x_*?

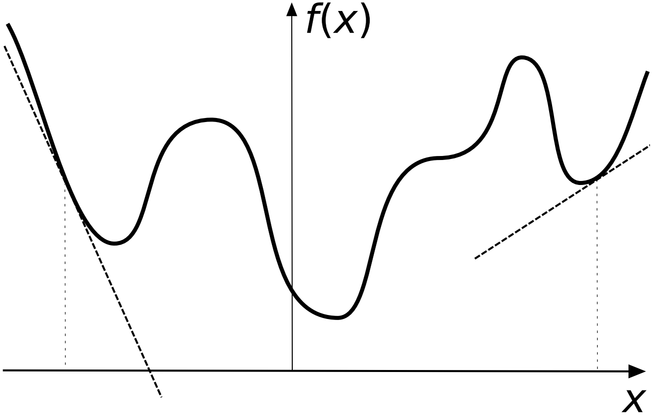


- $f(x) := dx + v : \mathbb{R} \to \mathbb{R}$ (linear) easy: always left if $d > 0$, right if $d < 0$
- Obvious idea: use the linear function that best locally approximates f
- Trusty old derivative $d = f'(x) = \lim_{t \to 0}[f(x+t) - f(x)]/t$ (putting a lot under the carpet even in $\mathbb{R}^n$, not to mention $\mathbb{F}^n$)
- Provided it exists

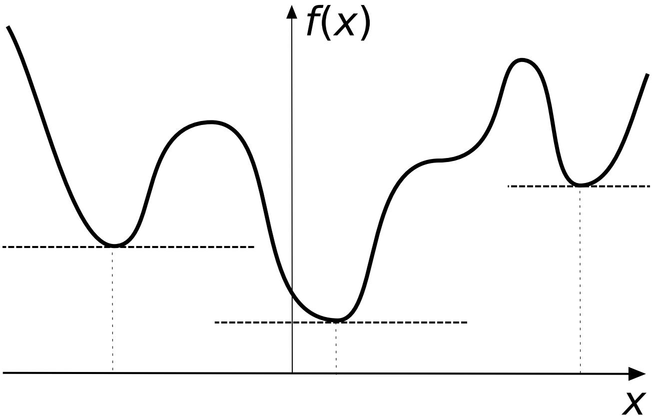
- (the) "Space ($\mathbb{F}^n$) is big. Really big. You just won't believe how vastly, hugely, mind-bogglingly big it is." Which way is x_*?



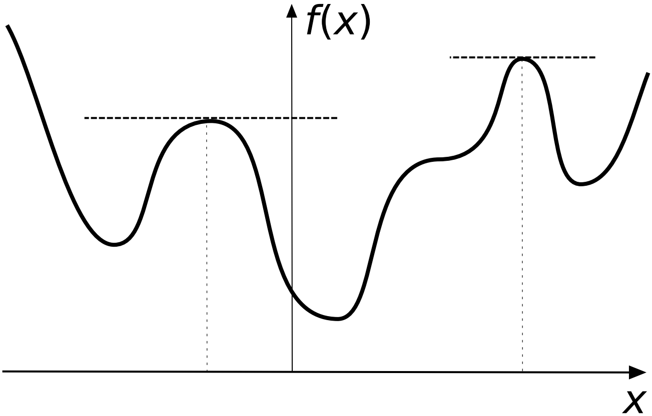
- $f(x) := dx + v : \mathbb{R} \to \mathbb{R}$ (linear) easy: always left if $d > 0$, right if $d < 0$
- Obvious idea: use the linear function that best locally approximates f
- Trusty old derivative $d = f'(x) = \lim_{t\to 0}[\,f(x+t) - f(x)\,]/t$ (putting a lot under the carpet even in $\mathbb{R}^n$, not to mention $\mathbb{F}^n$)
- Provided it exists ... and it is unique



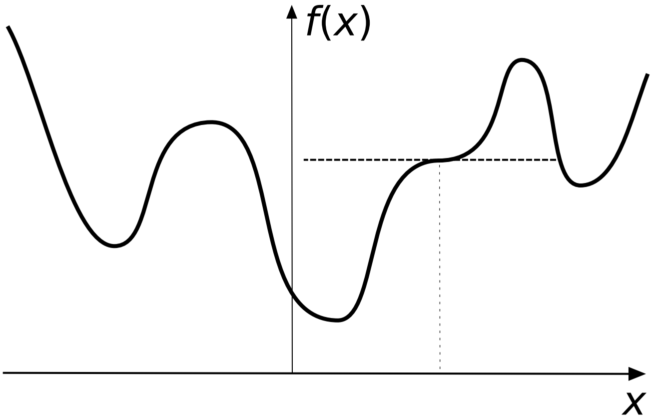
- $f'(x) < 0$ or $f'(x) > 0 \Longrightarrow x$ cannot be a local minimum



- $f'(x) < 0$ or $f'(x) > 0 \Longrightarrow x$ cannot be a local minimum
- $f'(x) = 0$ in all local minima $\Longrightarrow$ in the global one



- $f'(x) < 0$ or $f'(x) > 0 \Longrightarrow x$ cannot be a local minimum
- $f'(x) = 0$ in all local minima $\Longrightarrow$ in the global one
- However, $f'(x) = 0$ also in local (global) maxima



- $f'(x) < 0$ or $f'(x) > 0 \Longrightarrow x$ cannot be a local minimum
- $f'(x) = 0$ in all local minima $\Longrightarrow$ in the global one
- However, $f'(x) = 0$ also in local (global) maxima and in saddle points

- To find x_*, try finding stationary point x s.t. $f'(x) = 0$
- $f'(x) = 0$ (necessary, not sufficient) optimality condition: optimization is a system of (nonlinear) equations
- Still a lot hidden under the carpet:
  - what exactly is f' when $X \neq \mathbb{R}$?
  - $\imath_X$ has no derivative on "border" of $X \Longrightarrow$ has to "explicitly write" X as opposed to "hide" it in $\imath_X$ ($\mathcal{H}$, $\mathcal{G}$)
  - lots of "ifs" and "buts" (f' has to exist, X has to be "nice", ...)
- If all goes well, (local) optimality can be detected using derivatives (we'll see more details later in a simpler setting) $\Longrightarrow$ optimality conditions for PDE-CO is a PDE system. But:

- To find x_*, try finding stationary point x s.t. $f'(x) = 0$
- $f'(x) = 0$ (necessary, not sufficient) optimality condition: optimization is a system of (nonlinear) equations
- Still a lot hidden under the carpet:
  - what exactly is f' when $X \neq \mathbb{R}$?
  - $\imath_X$ has no derivative on "border" of $X \Longrightarrow$ has to "explicitly write" X as opposed to "hide" it in $\imath_X$ ($\mathcal{H}$, $\mathcal{G}$)
  - lots of "ifs" and "buts" (f' has to exist, X has to be "nice", ...)
- If all goes well, (local) optimality can be detected using derivatives (we'll see more details later in a simpler setting) $\Longrightarrow$ optimality conditions for PDE-CO is a PDE system. But:
  - significantly more complex than $\mathcal{H}$ itself

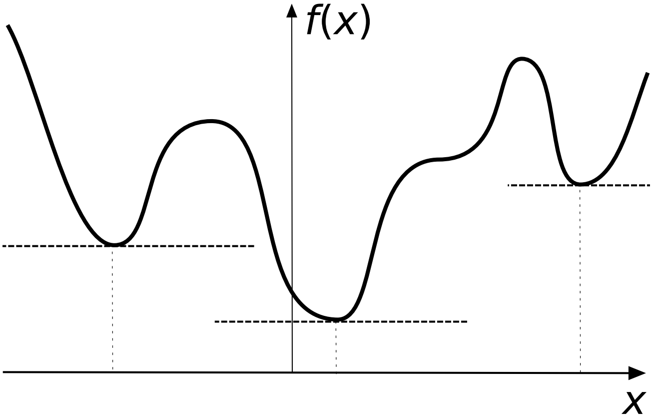
- To find x_*, try finding stationary point x s.t. $f'(x) = 0$
- $f'(x) = 0$ (necessary, not sufficient) optimality condition: optimization is a system of (nonlinear) equations
- Still a lot hidden under the carpet:
  - what exactly is f' when $X \neq \mathbb{R}$?
  - $\imath_X$ has no derivative on "border" of $X \Longrightarrow$ has to "explicitly write" X as opposed to "hide" it in $\imath_X$ ($\mathcal{H}$, $\mathcal{G}$)
  - lots of "ifs" and "buts" (f' has to exist, X has to be "nice", . . .)
- If all goes well, (local) optimality can be detected using derivatives (we'll see more details later in a simpler setting) $\Longrightarrow$ optimality conditions for PDE-CO is a PDE system. But:
  - significantly more complex than $\mathcal{H}$ itself
  - mathematical details far from easy

- To find x_*, try finding stationary point x s.t. $f'(x) = 0$
- $f'(x) = 0$ (necessary, not sufficient) optimality condition: optimization is a system of (nonlinear) equations
- Still a lot hidden under the carpet:
  - what exactly is f' when $X \neq \mathbb{R}$?
  - $\imath_X$ has no derivative on "border" of $X \Longrightarrow$ has to "explicitly write" X as opposed to "hide" it in $\imath_X$ ($\mathcal{H}$, $\mathcal{G}$)
  - lots of "ifs" and "buts" (f' has to exist, X has to be "nice", ...)
- If all goes well, (local) optimality can be detected using derivatives (we'll see more details later in a simpler setting) $\Longrightarrow$ optimality conditions for PDE-CO is a PDE system. But:
  - significantly more complex than $\mathcal{H}$ itself
  - mathematical details far from easy
  - PDE systems have no closed-form solution anyway

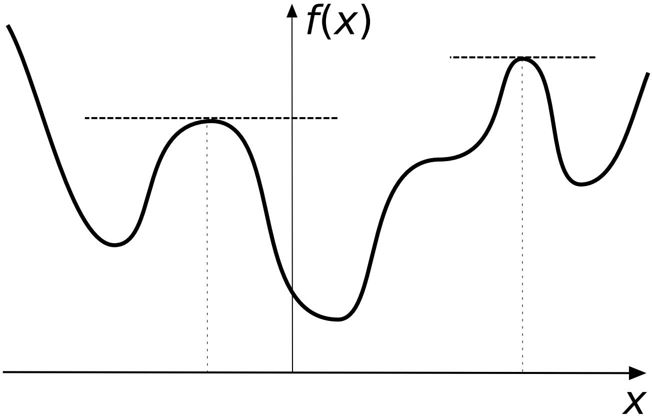
- To find x_*, try finding stationary point x s.t. $f'(x) = 0$
- $f'(x) = 0$ (necessary, not sufficient) optimality condition:
  optimization is a system of (nonlinear) equations
- Still a lot hidden under the carpet:
  - what exactly is f' when $X \neq \mathbb{R}$?
  - $\imath_X$ has no derivative on "border" of $X \Longrightarrow$
    has to "explicitly write" X as opposed to "hide" it in $\imath_X$ ($\mathcal{H}$, $\mathcal{G}$)
  - lots of "ifs" and "buts" (f' has to exist, X has to be "nice", ...)
- If all goes well, (local) optimality can be detected using derivatives
  (we'll see more details later in a simpler setting) $\Longrightarrow$
  optimality conditions for PDE-CO is a PDE system. But:
  - significantly more complex than $\mathcal{H}$ itself
  - mathematical details far from easy
  - PDE systems have no closed-form solution anyway
    $\Longrightarrow$ have to discretize the PDE and solve approximately

- Still back to "have to compute $f(\cdot)$ numerically"
- However, can now prove that x $(= [\, c\,,\, u\,])$ is a (local) minimum
- Also, algorithms that use derivatives are vastly more efficient
  (we'll see more details later in a simpler setting)
- Can quickly reach a (local) minimum and stop there:
  no more random moves for fear of having missed a better point nearby
- $|f'(x)| \approx$ "distance" from x_*, useful to choose accuracy of simulation
- Explicit optimality conditions leads to multiple strategies:
  - first discretize, then write optimality conditions
  - first write optimality conditions, then discretize

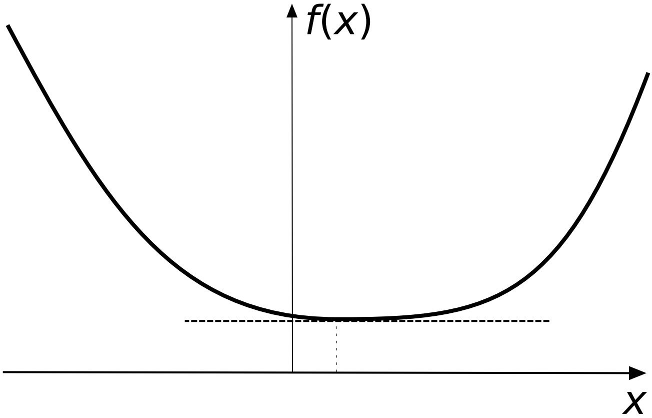
  (neither uniformly better, depends on the application)
- However, $f'(x) = 0$ only tells x may be a minimum,
  all is in vain if it rather is a maximum/saddle point
- There are ways to check it and ensure it'll never happen



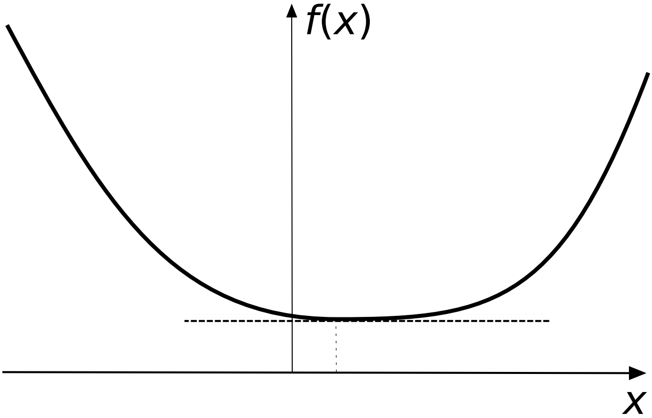
- If f has local minima



- If f has local minima and maxima, f' must first $\nearrow$ and then $\searrow$
- What if f' monotone?



- If f has local minima and maxima, f' must first $\nearrow$ and then $\searrow$
- What if f' monotone? f convex function $\Longrightarrow$



- If f has local minima and maxima, f' must first $\nearrow$ and then $\searrow$
- What if f' monotone? f convex function $\Longrightarrow$
  stationary point $\Longrightarrow$ local minimum $\Longrightarrow$ global minimum
- $f(\cdot)$ / X convex $\Longrightarrow$ x_* can be found "easily"
- (Restrictive) sufficient conditions to ensure $f(\cdot)$ / X convex

- In a nutshell: if everything goes very well
  - ℓ, $\mathcal{H}$ and $\mathcal{G}$ must have the right properties
  - details have to be worked out, options be wisely chosen
  - local optimality must be OK (or the problem convex to start with)
- There is no general-purpose PDE-OC solver,
  each case has to be dealt with individually
- However, tools are there, knowledge is there
- Problems of scale required by practical applications can be solved
  - with a little help from my (PDE-CO-savvy) friends
  - and possibly a supercomputer at hand
- As always, structure is your friend
  (e.g., optimal control has many specialized approaches exploiting time)
- Is it worth? In quite many cases, it is

Beware Mathematicians When They Seem to Speak Simple

Black-box Optimization

PDE-Constrained Optimization

NonLinear Nonconvex Problems

Mixed-Integer Convex (Linear) Problems

An Aside: Multiple Objectives

Do Not Underestimate Mixed-Integer Linear (Convex) Problems

Conclusions: Don't Be Afraid Of Optimization

- "Easier" problem: no PDE constraints, "only" algebraic ones

$$X = \{ x \in \mathbb{R}^n \,:\, g_i(x) \leq 0 \;\; i \in \mathcal{I} \,,\; h_j(x) = 0 \;\; j \in \mathcal{J} \}$$

  $\mathcal{I}$ = set of inequality constraints, $\mathcal{J}$ = set of equality constraints

- $G(x) = [\, g_i(x) \,]_{i \in \mathcal{I}} : \mathbb{R}^n \to \mathbb{R}^{|\mathcal{I}|}$, $H(x) = [\, h_i(x) \,]_{i \in \mathcal{J}} : \mathbb{R}^n \to \mathbb{R}^{|\mathcal{J}|}$

$$X = \{ x \in \mathbb{R}^n \,:\, G(x) \leq 0 \,,\; H(x) = 0 \}$$

  $G(\cdot)$, $H(\cdot)$ algebraic (vector-valued, multivariate) real functions

- Could always assume $|\mathcal{I}| = 1$ and $|\mathcal{J}| = 0$:
  - $h_j(x) = 0 \equiv h_j(x) \leq 0 \,\wedge\, -h_j(x) \leq 0$
  - $G(x) \leq 0 \equiv \max\{\, g_i(x) \,:\, i \in \mathcal{I} \,\} = g(x) \leq 0$

  but good reasons not to (exploit structure when is there)

- What does this gives to me? You know the drill: derivatives

- $f : \mathbb{R}^n \to \mathbb{R}$, partial derivative of f w.r.t. x_i at $x \in \mathbb{R}^n$:

  $$\frac{\partial f}{\partial x_i}(x) = \lim_{t \to 0} \frac{f(x_1, \ldots, x_{i-1}, x_i + t, x_{i+1}, \ldots, x_n) - f(x)}{t}$$

  just $f'(x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n)$ treating x_j for $j \neq i$ as constants

- Good news: computing derivatives mechanic, Automatic Differentiation software will do it for you, not only from formulæ but from code

- $f : \mathbb{R}^n \to \mathbb{R}$, partial derivative of f w.r.t. x_i at $x \in \mathbb{R}^n$:

$$\frac{\partial f}{\partial x_i}(x) = \lim_{t \to 0} \frac{f(\,x_1\,, \ldots\,, x_{i-1}\,, x_i + t\,, x_{i+1}\,, \ldots\,, x_n\,) - f(\,x\,)}{t}$$

  just $f'(x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n)$ treating x_j for $j \neq i$ as constants
- Good news: computing derivatives mechanic, Automatic Differentiation software will do it for you, not only from formulæ but from code
  … provided they exist ($f(x) = |x|$, $f'(x) = ???$)
- Gradient = vector of all partial derivatives all important in optimization

$$\nabla f(x) := \left[\ \frac{\partial f}{\partial x_1}(x)\,,\ \ldots\ ,\ \frac{\partial f}{\partial x_n}(x)\ \right]$$

- $\nabla f(x) \approx [\, f(x + e_i \varepsilon) - f(x)] \,/\, \varepsilon$, $i = 1, \ldots, n$ (which ε?)
- f differentiable at $x \approx \forall i \ \frac{\partial f}{\partial x_i}(\cdot)$ continuous ($\Longleftarrow \exists$)
- $f \in C^1$: $\nabla f(x)$ continuous $\equiv f$ differentiable ($\Longrightarrow f$ continuous) $\forall\, x$
- $f \in C^1 \Longrightarrow$ finding local minimum (actually, stationary point) "easy": just go in the other direction ($-\nabla f(x)$ = steepest descent direction)

- Vector-valued function $f : \mathbb{R}^n \to \mathbb{R}^m$, $f(x) = [\, f_1(x)\,,\, f_2(x)\,,\, \dots\,,\, f_m(x)\,]$
- Partial derivative: usual stuff, except with extra index

$$\frac{\partial f_j}{\partial x_i}(x) = \lim_{t\to 0} \frac{f_j(\, x_1\,, \dots, x_{i-1}\,, x_i + t\,, x_{i+1}\,, \dots, x_n\,) - f_j(\, x\,)}{t}$$

- $\nabla f(x) : \mathbb{R}^n \to \mathbb{R}^n$ itself has a gradient: Hessian of f

$$\nabla^2 f(x) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2}(x) & \frac{\partial^2 f}{\partial x_2 \partial x_1}(x) & \dots & \frac{\partial^2 f}{\partial x_n \partial x_1}(x) \\ \frac{\partial^2 f}{\partial x_1 \partial x_2}(x) & \frac{\partial^2 f}{\partial x_2^2}(x) & \dots & \frac{\partial^2 f}{\partial x_n \partial x_2}(x) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_1 \partial x_n}(x) & \frac{\partial^2 f}{\partial x_2 \partial x_n}(x) & \dots & \frac{\partial^2 f}{\partial x_n^2}(x) \end{bmatrix}$$

second-order partial derivative (just do it twice) $\quad \dfrac{\partial^2 f}{\partial x_j \partial x_i} \qquad \dfrac{\partial^2 f}{\partial x_i \partial x_i} = \dfrac{\partial^2 f}{\partial x_i^2}$

- $\nabla^2 f(x) \approx n$ times $\nabla f(x) \approx n^2$ times $f(x)$: packs a lot of information
- $f \in C^2$: $\nabla^2 f(x)$ continuous ($\Longrightarrow$ symmetric) $\equiv \nabla f$ differentiable $\forall\, x$
- $f \in C^2 \Longrightarrow$ finding local minimum (stationary point) “super-easy”

- $X = \mathbb{R}^n \Longrightarrow$ all depends on quality of derivatives of f:
  - $f \notin C^1 \Longrightarrow$ subgradient methods $\Longrightarrow$ sublinear convergence (error at step $k \approx 1/\sqrt{k}$, $O(1/\varepsilon^2)$ iterations)
  - $f \in C^1 \Longrightarrow$ gradient methods $\Longrightarrow$ linear convergence (error at step $k \approx \gamma^k$ with $\gamma < 1$, $O(1/\log(\varepsilon))$ iterations)
  - $f \in C^2 \Longrightarrow$ Newton-type methods $\Longrightarrow$ superlinear/quadratic convergence (error at step $k \approx \gamma^{k^2}/\gamma^{2^k}$ with $\gamma < 1$, $\approx O(1)$ iterations)
- All bounds $\approx$ independent from n (can be "hidden in the constants") $\Longrightarrow$ good for large-scale problems (n very large)
- Not all trivial, line search/trust region, globalization, ...
- Gradient methods can be rather slow in practice ($\gamma \approx 1$), need to cure zig-zagging (heavy ball, fast gradients, ...)
- Hessian a big guy, inverting it $O(n^3)$ a serious issue for large-scale: quasi-Newton/conjugate gradient only $O(n^2)$ / $O(kn)$ (but trade-offs)
- All in all, local (unconstrained) convergence very well dealt with

- Local optimality still "easy" to characterize via derivatives
- Karush-Kuhn-Tucker conditions: $\exists\, \lambda \in \mathbb{R}^{|\mathcal{I}|}_{+}$ and $\mu \in \mathbb{R}^{|\mathcal{J}|}$ s.t.

$$g_i(x) \leq 0 \;\; i \in \mathcal{I} \quad , \quad h_j(x) = 0 \;\; j \in \mathcal{J} \tag{KKT-F}$$

$$\nabla f(x) + \textstyle\sum_{i \in \mathcal{I}} \lambda_i \nabla g_i(x) + \sum_{j \in \mathcal{J}} \mu_j \nabla h_j(x) = 0 \tag{KKT-G}$$

$$\textstyle\sum_{i \in \mathcal{I}} \lambda_i g_i(x) = 0 \tag{KKT-CS}$$

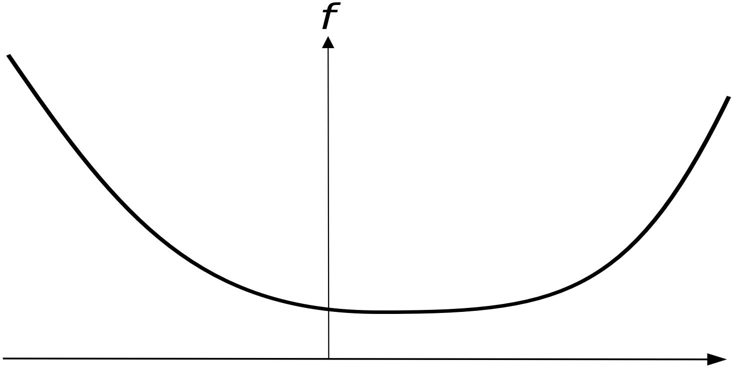
  $\equiv$ x stationary point of Lagrangian function (in x, λ / μ parameters)

  $L(\, x \,;\, \lambda \,,\, \mu \,) = f(x) + \sum_{i \in \mathcal{I}} \lambda_i g_i(x) + \sum_{j \in \mathcal{I}} \mu_j h_j(x)$ ($\rightsquigarrow$ duality ...)

- KKT Theorem: x local optimum + constraint qualifications $\Longrightarrow$ (KKT)
  a) Affine constraints: g_i and h_j affine $\forall\, i \in \mathcal{I}$ and $j \in \mathcal{J}$
  b) Slater's condition: g_i convex, h_j affine, $\exists\, \bar{x} \in X$ s.t. $g_i(\bar{x}) < 0 \; \forall\, i$
  c) Linear independence of $\nabla g_i(x)$ (i active) and $\nabla h_j(x)$ (all)
- (P) convex problem: (KKT) $\Longrightarrow$ x global optimum
- Otherwise, quite involved second-order optimality conditions ...

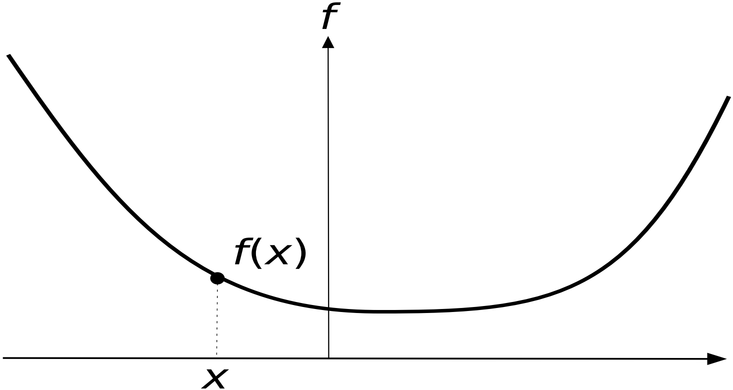
- In a nutshell, that $\exists$ efficient local algorithms
- At the very least, (KKT) $\approx$ tell x local minimum, stop the search
- Checking if (KKT) holds "easy" (Farkas' Lemma ...)
- Optimization $\equiv$ solving systems of nonlinear equations and inequalities
- Does not mean that algorithms are obvious:
  - several different forms (primal, dual, ...)
  - several different ideas (active set, projection, barrier, penalty, ...)
  - combinatorial aspects (active set choice) may make them inefficient
- Yet, provably and practically efficient algorithms are there if data of the problem nice (f, $G \in C^1/C^2$, H affine ...)
- Particularly relevant/elegant class: primal-dual interior point methods
- (Reasonably) robust and efficient implementations available, although numerical issues (linear algebra accuracy/cost) still nontrivial

- Unfortunately an entirely different game: sifting through all X required
- Derivatives a local object, can't give global information
  except in the convex case, where they actually do



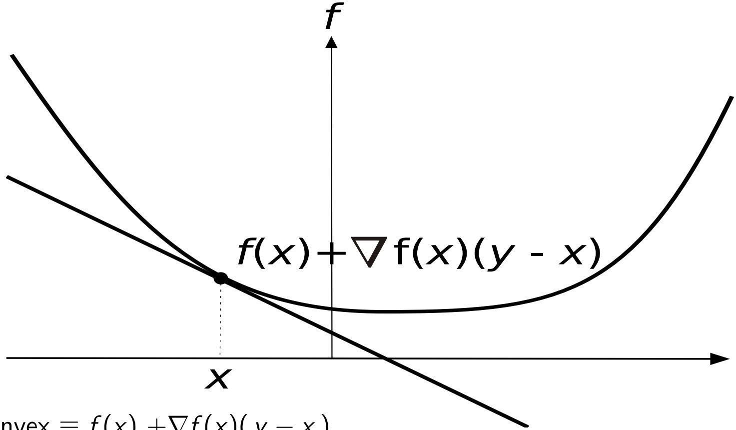
- f convex $\equiv$

- Unfortunately an entirely different game: sifting through all X required
- Derivatives a local object, can't give global information
  except in the convex case, where they actually do



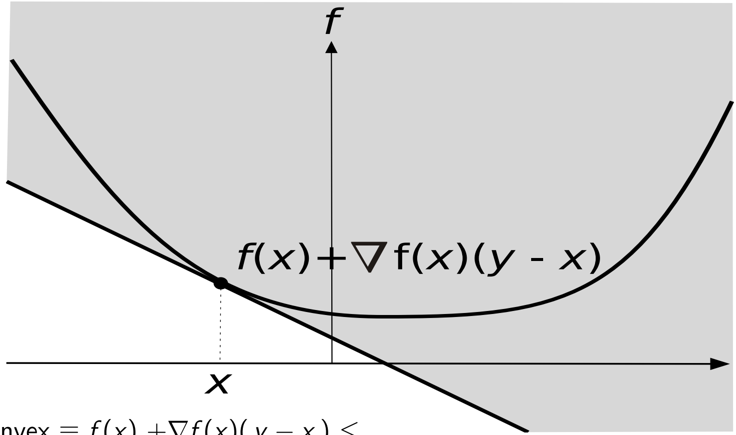
- f convex $\equiv f(x)$

- Unfortunately an entirely different game: sifting through all X required
- Derivatives a local object, can't give global information
  except in the convex case, where they actually do



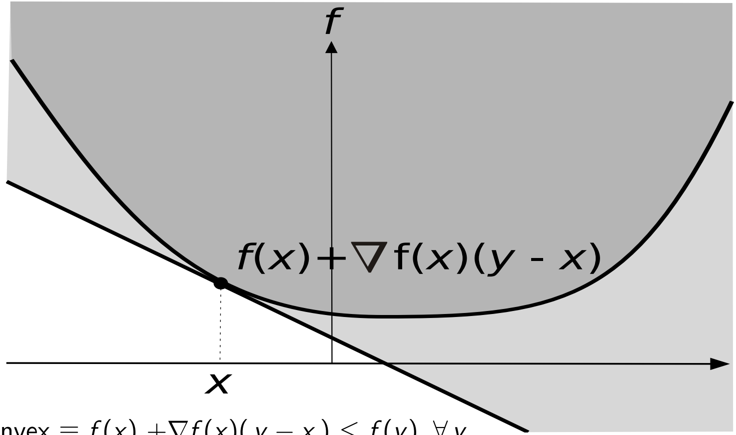
- f convex $\equiv f(x) + \nabla f(x)(y - x)$

- Unfortunately an entirely different game: sifting through all X required
- Derivatives a local object, can't give global information
  except in the convex case, where they actually do



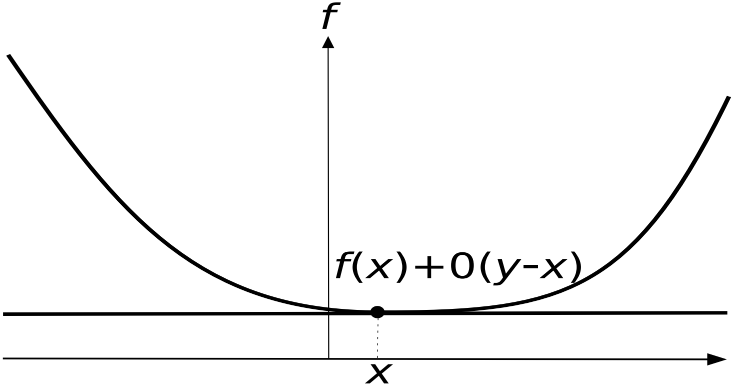
- f convex $\equiv f(x) + \nabla f(x)(y - x) \leq$

- Unfortunately an entirely different game: sifting through all X required
- Derivatives a local object, can't give global information
  except in the convex case, where they actually do



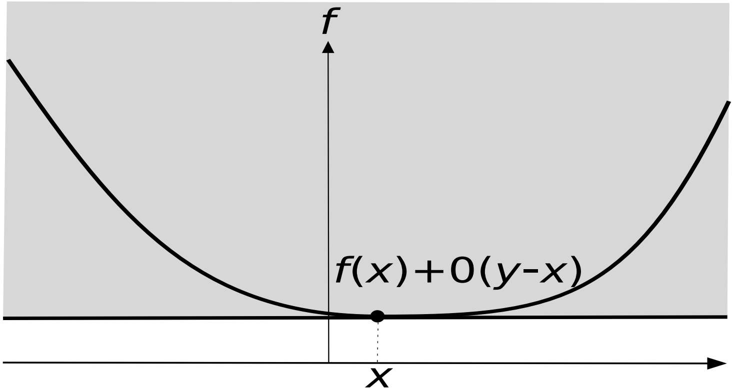
- f convex $\equiv f(x) + \nabla f(x)(y - x) \leq f(y) \;\; \forall\, y$

- Unfortunately an entirely different game: sifting through all X required
- Derivatives a local object, can't give global information
  except in the convex case, where they actually do



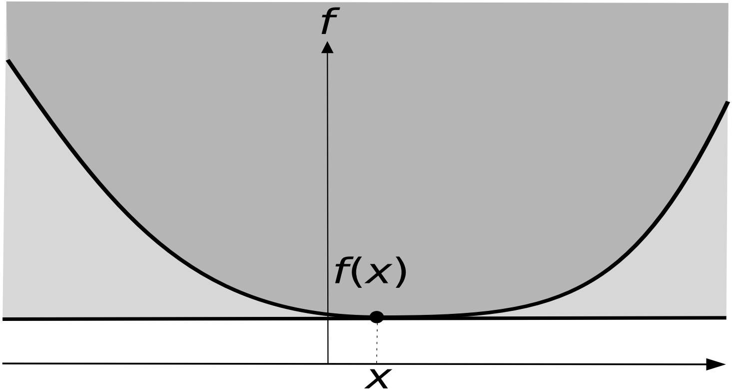
- f convex $\equiv f(x) + \nabla f(x)(y - x) \leq f(y) \;\; \forall\, y$
- $\nabla f(x) = 0$

- Unfortunately an entirely different game: sifting through all X required
- Derivatives a local object, can't give global information
  except in the convex case, where they actually do



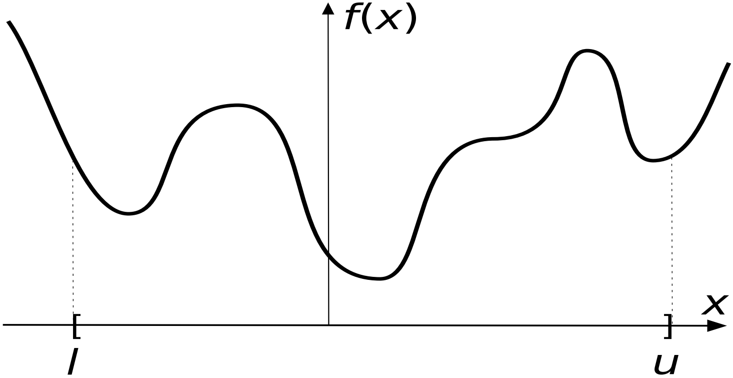
- f convex $\equiv f(x) + \nabla f(x)(\, y - x\,) \leq f(y) \;\; \forall\, y$
- $\nabla f(x) = 0 \Longrightarrow f(x)$

- Unfortunately an entirely different game: sifting through all X required
- Derivatives a local object, can't give global information
  except in the convex case, where they actually do

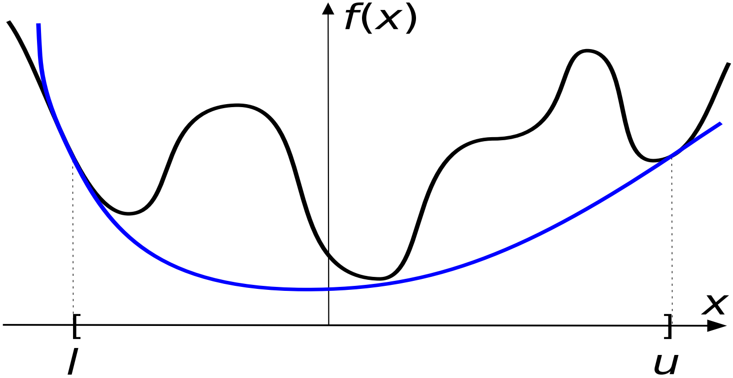


- f convex $\equiv f(x) + \nabla f(x)(y - x) \leq f(y) \;\; \forall\, y$
- $\nabla f(x) = 0 \Longrightarrow f(x) \;\; [+0(y - x)] \leq f(y) \;\; \forall\, y$

- Sift through all X $(= [\, l\, ,\ u\,])$ but using a clever guide

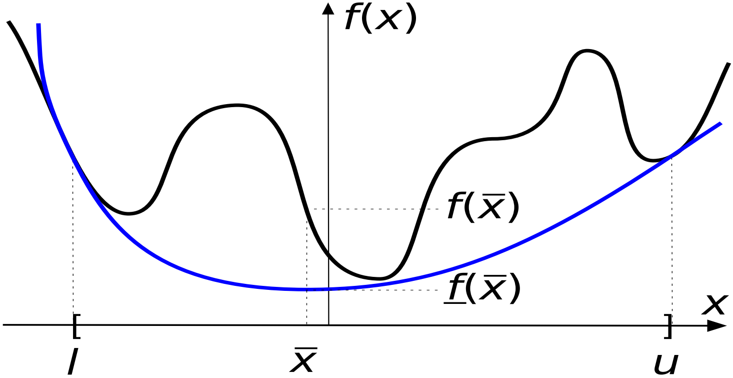


- Sift through all X ($= [\,l\,,\,u\,]$) but using a clever guide



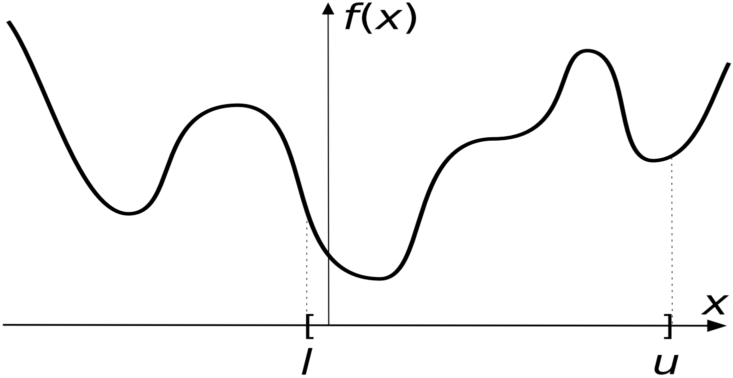
- Main idea: convex lower approximation $\underline{f}$ of nonconvex f on X

- Sift through all X $(= [\,l\,,\,u\,])$ but using a clever guide



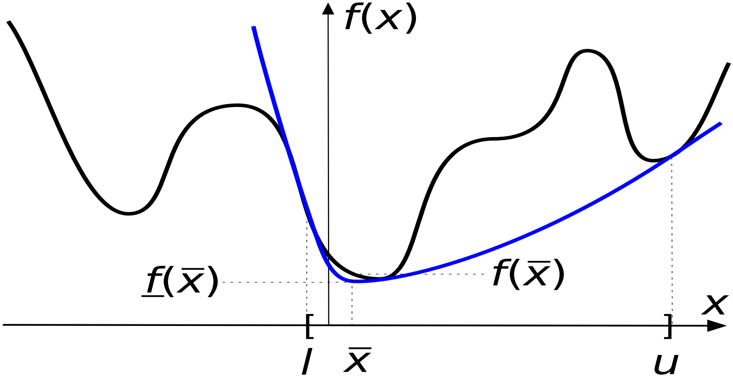
- Main idea: convex lower approximation $\underline{f}$ of nonconvex f on X
- Can easily find minimum $\bar{x}$, giving $\underline{f}(\bar{x}) \leq f_* \leq f(\bar{x})$

- Sift through all X $(= [\, l\,,\, u\,])$ but using a clever guide



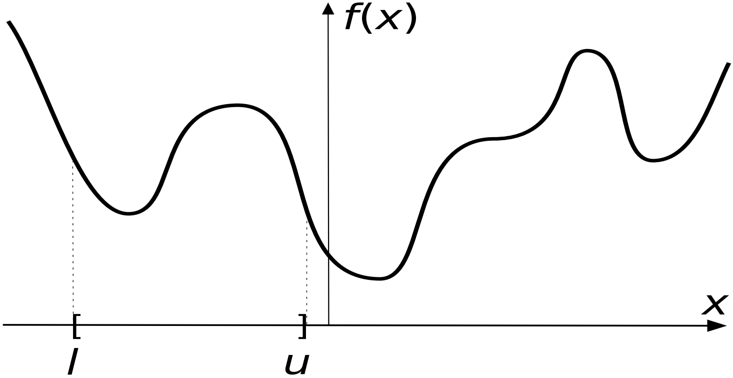
- Main idea: convex lower approximation $\underline{f}$ of nonconvex f on X
- Can easily find minimum $\bar{x}$, giving $\underline{f}(\bar{x}) \leq f_* \leq f(\bar{x})$
- If gap $f(\bar{x}) - \underline{f}(\bar{x})$ too large, partition X and iterate

- Sift through all X $(= [\,l\,,\ u\,])$ but using a clever guide



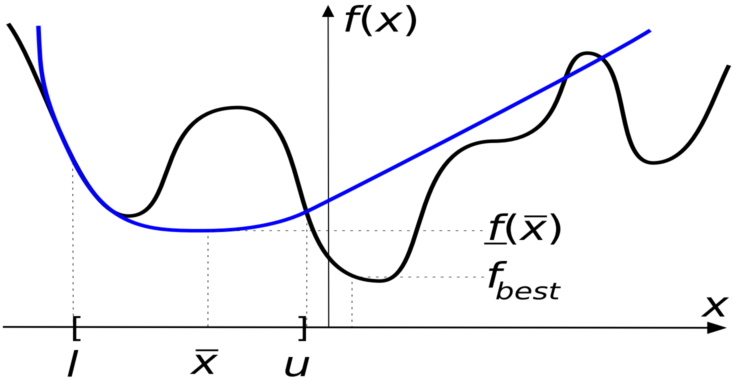
- Main idea: convex lower approximation $\underline{f}$ of nonconvex f on X
- Can easily find minimum $\bar{x}$, giving $\underline{f}(\bar{x}) \leq f_* \leq f(\bar{x})$
- If gap $f(\bar{x}) - \underline{f}(\bar{x})$ too large, partition X and iterate
- $\underline{f}$ depends on partition, smaller partition $\Longrightarrow$ better gap (hopefully)

- Sift through all X ($= [\, l\,,\, u\,]$) but using a clever guide



- Main idea: convex lower approximation $\underline{f}$ of nonconvex f on X
- Can easily find minimum $\bar{x}$, giving $\underline{f}(\bar{x}) \leq f_* \leq f(\bar{x})$
- If gap $f(\bar{x}) - \underline{f}(\bar{x})$ too large, partition X and iterate
- $\underline{f}$ depends on partition, smaller partition $\implies$ better gap (hopefully)
- If on some partition

- Sift through all X $(= [\,l\,,\ u\,])$ but using a clever guide



- Main idea: convex lower approximation $\underline{f}$ of nonconvex f on X
- Can easily find minimum $\bar{x}$, giving $\underline{f}(\bar{x}) \leq f_* \leq f(\bar{x})$
- If gap $f(\bar{x}) - \underline{f}(\bar{x})$ too large, partition X and iterate
- $\underline{f}$ depends on partition, smaller partition $\Longrightarrow$ better gap (hopefully)
- If on some partition $\underline{f}(\bar{x}) \geq$ best f-value so far, partition killed for good

- In a word? No
- Worst-case: keep dicing and slicing X until pieces "very small"
- However, in practice it depends on:
  - "how much nonconvex" f really is
  - how good $\underline{f}$ is as a lower approximation of f
- Best possible lower approximation: convex envelope
- Bad news: computing convex envelopes is hard
- Typical approach:
  - rewrite the expression of f in terms of unary/binary functions
  - apply specific convexification formulæ for each function
- Tedious job, bounds often rather weak
- Good news: implemented in available, well-engineered solvers
- Good news: immensely less inefficient in practice than blind search
  (at least, bounds allow to cut away whole regions for good)

- In a nutshell: if everything goes well
  - f, G and H must have the right properties
  - the less nonconvex they are, the better
  - the less "complicated" they are, the better
- Yet, there are general-purpose nonconvex MINLP solvers which can solve the problem to proven optimality
- Using them nontrivial, formulating the problem well crucial ($\equiv$ so that a "good $\underline{f}$ is available")
- Not really "large-scale", but 100s/1000s of variables often doable quickly enough with off-the-shelf tools
- Much larger problems also possible with special tools/effort
- As always, structure is your friend
- Is it worth? In quite many cases, it is (ask chemical Engineers)

Beware Mathematicians When They Seem to Speak Simple

Black-box Optimization

PDE-Constrained Optimization

NonLinear Nonconvex Problems

Mixed-Integer Convex (Linear) Problems

An Aside: Multiple Objectives

Do Not Underestimate Mixed-Integer Linear (Convex) Problems

Conclusions: Don't Be Afraid Of Optimization

- Computing "good" convex approximation both complex and difficult
- One very relevant case in which at least "easy": integrality constraints
- $x_i \in \mathbb{Z}$, most often $x_i \in \{\, 0 \,,\, 1 \,\}$ very convenient for discrete choices (start machine/don't, make trip/don't, ...)
- Clearly nonconvex, $\exists$ nonlinear versions ($x_i(1 - x_i) \leq 0, \ldots$)
- Actually quite powerful: many different nonlinear nonconvex structures can be expressed via that + "simple" (linear) constraints
- Yet, this requires some rather weird formulation tricks
  $z = xy \equiv [\, (z \leq x) \wedge (z \leq y) \wedge (z \geq x + y - 1) \,]$ if all $x, y, z \in \{\, 0 \,,\, 1 \,\}$
- If all the rest in the problem convex, then a convex relaxation very easy: continuous relaxation ($x_i \in \mathbb{Z} \longrightarrow x_i \in \mathbb{R}$)
- This does not mean convex relaxation is good, but it may be
- At least makes life a lot easier to solution algorithms

- Finding good relaxations crucial for practical efficiency
- Solvers helped a lot by having few, well-controlled nonconvexities
- Mixed-Integer Convex Problems the easiest class of hard problems
- Especially famous special case: Mixed-Integer Linear Program

  $$(MILP) \quad \min \{\, cx \,:\, Ax \geq b \;,\; x_i \in \mathbb{Z} \;\; i \in I \,\}$$

  $\Longrightarrow$ continuous relaxation $\equiv$ Linear Program
- Very stable and efficient algorithms, some $\approx$ unique (simplex methods)
- Very powerful methods to improve relaxation quality (valid inequalities)
- Countless many results about special combinatorial structures (paths, trees, cuts, matchings, cliques, covers, knapsacks, ...)
- Clever approaches to exploit structure, though some work for MINLP too (Column/Row Generation, Dantzig-Wolfe/Benders' Decomposition, ...)

- Fundamental point: formulating the problem well is crucial
- Almost anything can be written as a MILP, albeit to some $\approx$ (not always a good idea: some nonlinearities "nice")
- Many different ways to write the same problem: apparently minor changes can make orders-of-magnitude difference
- Several of the best formulation weird and/or very large (appropriate tricks to only generate the strictly required part)
- Doing it "by hand" should not be required: solvers should be able to automatically find the best formulation (reformulate)
- Good news: the "perfect" formulation provably exists
- Bad news: it is provably ($\mathcal{NP}$-)hard to construct
- Doing it automatically is clearly difficult (but we should try harder)
- Meanwhile, a well-trained eye can make a lot of difference

- Good news: can "hide" many nonlinearities in a Linear Program
- Conic Program: $(P) \;\; \min\{\, cx \,:\, Ax \geq_K b \,\}$
  where $x \geq_K y \equiv x - y \in K$, K pointed convex cone, e.g.
  - $K = \mathbb{R}^n_+ \equiv$ sign constraints $\equiv$ Linear Program
  - $K = \mathbb{L} = \{\, x \in \mathbb{R}^n \,:\, x_n \geq \sqrt{\sum_{i=1}^{n-1} x_i^2} \,\} \equiv$ Second-Order Cone Program
  - $K = \mathbb{S}_+ = \{\, A \succeq 0 \,\} \equiv$ "$\succeq$" constraints $\equiv$ SemiDefinite Program
- Exceedingly smart idea: everything is linear, but the cone is not
  $\equiv$ a nonlinear program disguised as a linear one
- Contains as special case convex quadratic functions
- Many interesting (convex) nonlinear functions have a conic representation (but have to learn some even weirder formulation trick)
- Continuous relaxation almost as efficient as Linear Program
- Many combinatorial MILP tricks extend di MI-SOCP (valid surfaces, ...)
- Support in general-purpose software growing, already quite advanced

So, Can I Solve Mixed-Integer Linear (Convex) Problems?

- In a nutshell: unless something goes very bad
  - data of the problem by definition is nice
  - a feasible relaxation always there, bounds can be quite good
  - lots of good ideas (cutting planes, general-purpose heuristics, . . .)
- Plenty of general-purpose, well-engineered MILP/MI-SOCP solvers which can solve the problem to proven optimality
- Lots of useful supporting software: algebraic modelling languages, (there for MINLP too), IDEs, interfaces with database/spreadsheet, . . .
- 10000/100000 variables often doable in minutes/hours on stock hw/sw if you write the right model
- Much larger problems (10^6 / 10^9) also possible with special tools/effort
- As always, structure is your friend, and many known forms of structures
- Is it worth? In very many cases, it is

Beware Mathematicians When They Seem to Speak Simple

Black-box Optimization

PDE-Constrained Optimization

NonLinear Nonconvex Problems

Mixed-Integer Convex (Linear) Problems

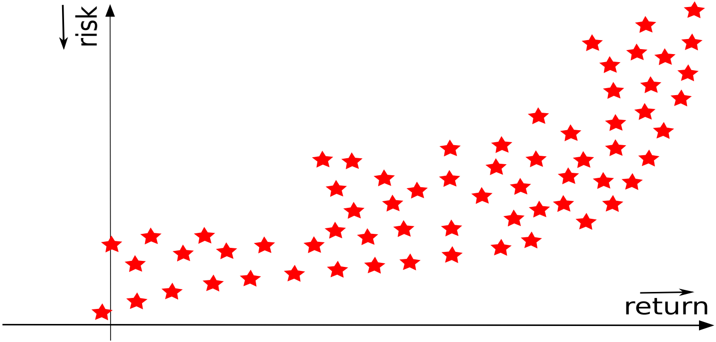
An Aside: Multiple Objectives

Do Not Underestimate Mixed-Integer Linear (Convex) Problems

Conclusions: Don't Be Afraid Of Optimization

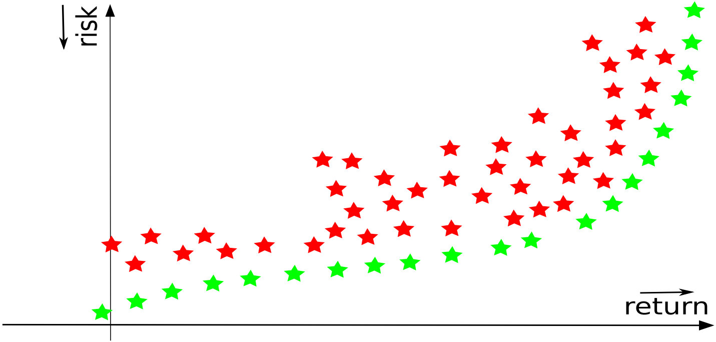
- Often, the actual problem is $\min \{ [f_1(x) , f_2(x)] \; : \; x \in X \}$
  more than one objective, with incomparable units (apples & oranges)

- Often, the actual problem is $\min \{ [f_1(x) , f_2(x)] : x \in X \}$
  more than one objective, with incomparable units (apples & oranges)
- Textbook example: portfolio selection problem



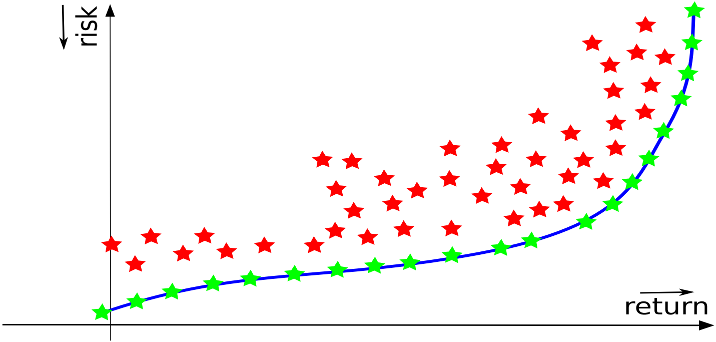
- No "best" solution, only

- Often, the actual problem is $\min \{ [f_1(x), f_2(x)] : x \in X \}$
  more than one objective, with incomparable units (apples & oranges)
- Textbook example: portfolio selection problem



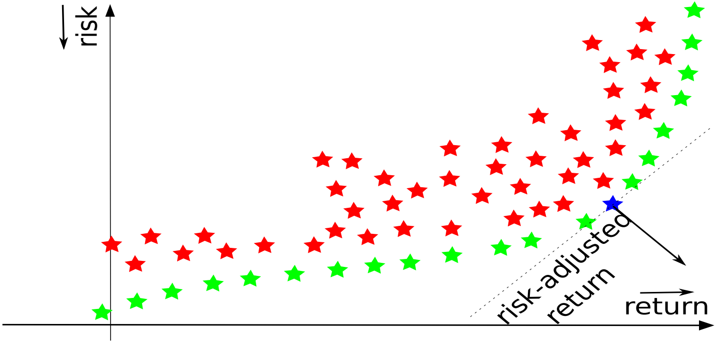
- No "best" solution, only non-dominated ones on the

- Often, the actual problem is $\min\{\, [\, f_1(x) \,,\, f_2(x) \,] \;:\; x \in X \,\}$
  more than one objective, with incomparable units (apples & oranges)
- Textbook example: portfolio selection problem



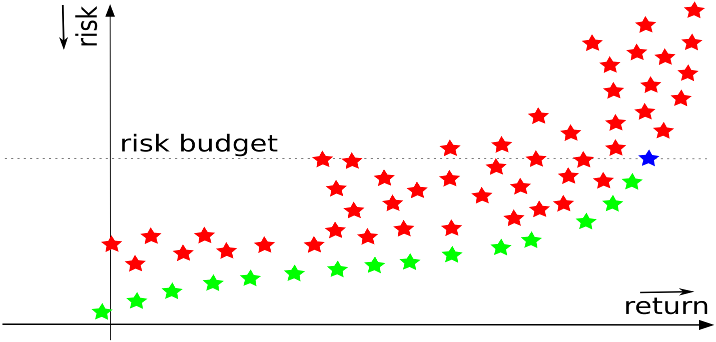
- No "best" solution, only non-dominated ones on the Pareto frontier
- Two practical solutions:

- Often, the actual problem is $\min \{ [f_1(x) , f_2(x)] \ : \ x \in X \}$
  more than one objective, with incomparable units (apples & oranges)
- Textbook example: portfolio selection problem



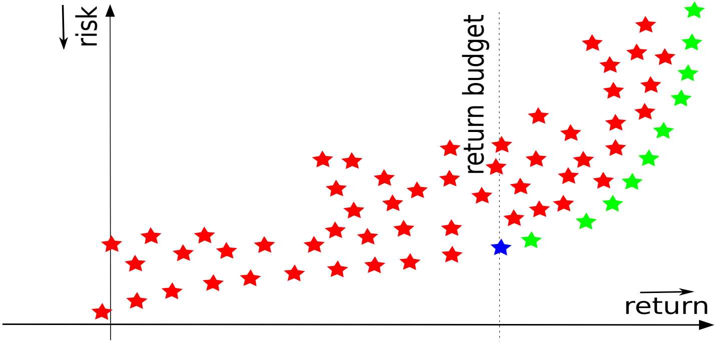
- No "best" solution, only non-dominated ones on the Pareto frontier
- Two practical solutions: maximize risk-adjusted return,
  a.k.a. scalarization $\min \{ f_1(x) + \alpha f_2(x) \ : \ x \in X \}$ (which α??)

- Often, the actual problem is $\min \{ [f_1(x) , f_2(x)] : x \in X \}$
  more than one objective, with incomparable units (apples & oranges)
- Textbook example: portfolio selection problem



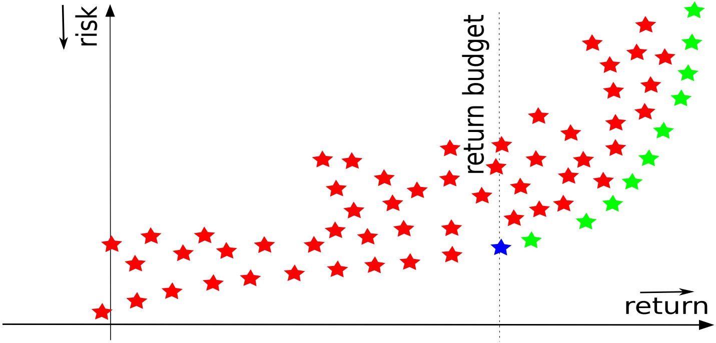
- No "best" solution, only non-dominated ones on the Pareto frontier
- Two practical solutions: maximize return with budget on maximum risk,
  a.k.a. budgeting $\min \{ f_1(x) : f_2(x) \leq \beta_2 , x \in X \}$ (which β_2??)

- Often, the actual problem is $\min \{ \, [\, f_1(x) \, , \, f_2(x) \,] \; : \; x \in X \, \}$
  more than one objective, with incomparable units (apples & oranges)
- Textbook example: portfolio selection problem



- No "best" solution, only non-dominated ones on the Pareto frontier
- Two practical solutions: minimize risk with budget on minimum return,
  a.k.a. budgeting $\min \{ \, f_2(x) \; : \; f_1(x) \leq \beta_1 \, , \, x \in X \, \}$ (which β_1??)

- Often, the actual problem is $\min \{ [f_1(x) , f_2(x)] \; : \; x \in X \}$
  more than one objective, with incomparable units (apples & oranges)
- Textbook example: portfolio selection problem



- No "best" solution, only non-dominated ones on the Pareto frontier
- Two practical solutions: minimize risk with budget on minimum return,
  a.k.a. budgeting $\min \{ f_2(x) \; : \; f_1(x) \leq \beta_1 \, , \; x \in X \}$ (which β_1??)
- All a bit fuzzy, but it's the nature of the beast

Beware Mathematicians When They Seem to Speak Simple

Black-box Optimization

PDE-Constrained Optimization

NonLinear Nonconvex Problems

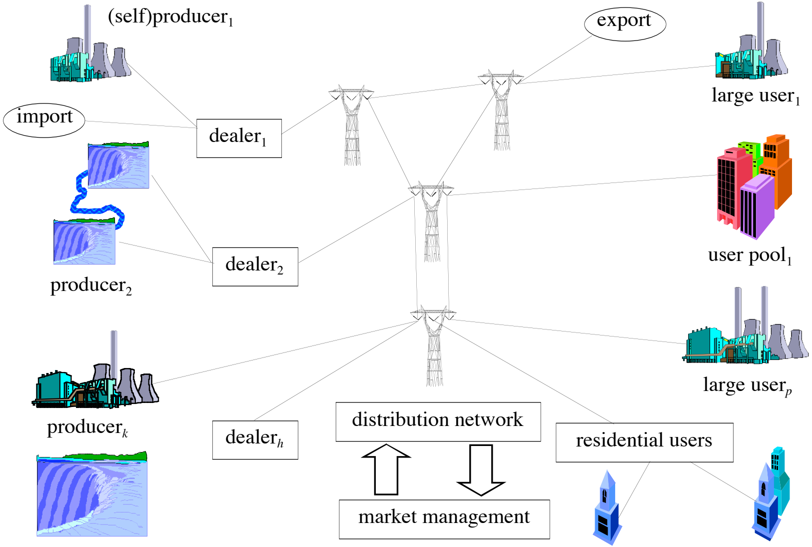
Mixed-Integer Convex (Linear) Problems

An Aside: Multiple Objectives

Do Not Underestimate Mixed-Integer Linear (Convex) Problems

Conclusions: Don't Be Afraid Of Optimization

- Schedule a set of generating units over a set of time instants to satisfy the (forecasted) energy demand at each



- Gazzillions €€€ / $$$, enormous amount of research[1]
- Different types of production units, different constraints:
  - Thermal (comprised nuclear): min/max production, min up/down time, ramp rates on production increase/decrease, start-up cost depending on previous downtime, others (modulation, ...)
  - Hydro (valleys): min/max production, min/max reservoir volume, time delay to get to the downstream reservoir, others (pumping, ...)
  - Non programmable (ROR hydro) intermittent units (solar/wind, ...)
  - Fancy things (small-scale storage, demand response, smart grids, ...)
- The electrical network (AC/DC, transmission/distribution, OTS, ...)
- The right formulation of the individual units may make a lot of difference for thermals[2,3], hydros[4], ...

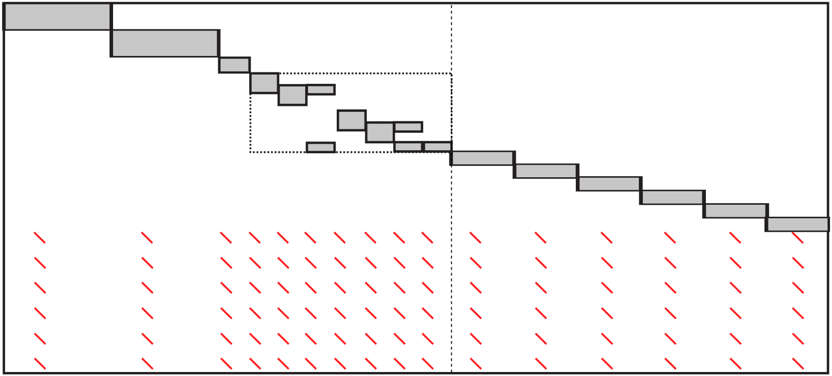
---

[1] van Ackooij, Danti Lopez, F., Lacalandra, Tahanan "Large-scale Unit Commitment Under Uncertainty [...]" *AOR*, 2018

[2] F., Gentile, Lacalandra "Tighter Approximated MILP Formulations for Unit Commitment Problems" IEEE TPWRS, 2009

[3] F., Gentile "New MIP Formulations for the Single-Unit Commitment Problems with Ramping Constraints" *TR*, 2015

[4] d'Ambrosio, Lodi, S. Martello. "Piecewise linear approxmation of functions of two variables in MILP models" *ORL*, 2010



- Many blocks of rather different shape and size + linking constraints
- A very-large-scale MIQP to be solved in <15' operationally
- Can be done via Lagrangian decomposition[5] and specialized solution methods for thermal[6]/hydro[7] units

---

[5] F., Gentile, Lacalandra "Solving Unit Commitment Problems with General Ramp Contraints" *IJEPES*, 2008

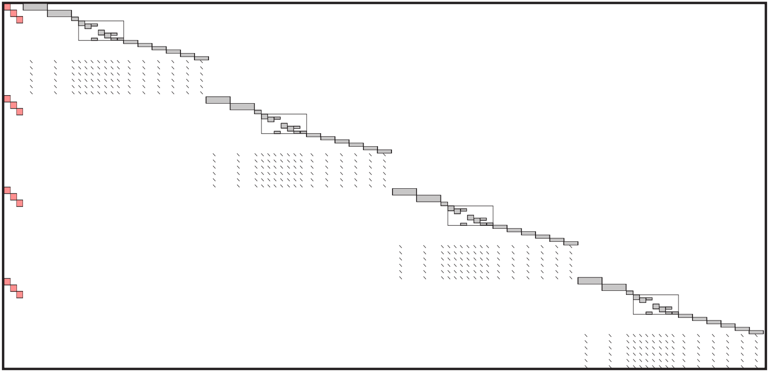
[6] F., Gentile "Solving non-linear single-unit commitment problems with ramping constraints"' *OR*, 2006

[7] Taktak, D'Ambrosio "An overview on [...] unit commitment problem in hydro valleys" *Energy Systems*, 2017

- Maybe if this were the end, but it is just the beginning
- Data is uncertain: demand, wind/solar production, units/network state …which cannot be ignored (increased RES penetration …)
- Need methods to represent uncertainty[1]:
  - Robust Optimization: uncertain data lives anywhere in a (convex) uncertainty set (how chosen? too conservative, price-of-robustness, …)
  - Chance-Constrained Optimization: constraints hold with high probability (estimate distributions, nasty numerical integrals galore, …)
  - Stochastic Optimization[8]: here-and-now decisions in advance, recourse decisions when the uncertainty reveals itself
  - Hybrids[9], …
- UC typically stochastic: day-ahead commitment, to-the-minute dispatch
- Simplest approach scenario-based: each scenario ≈ a full UC $\Longrightarrow$ size increases by a factor # scenarios, which should be large

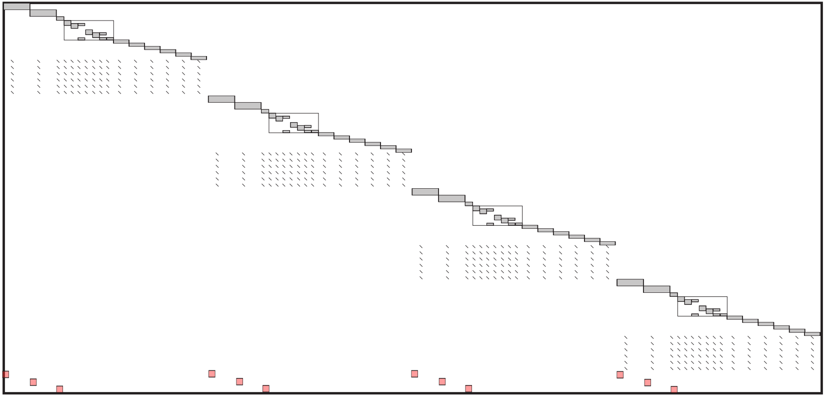
---

[8] Scuzziato, Finardi, F. "Comparing Spatial and Scenario Decomposition for Stochastic […]" *IEEE Trans. Sust. En.*, 2018
[9] van Ackooij, F., de Oliveira "Inexact Stabilized Benders' Decomposition Approaches, with Application […]" *CO&A*, 2016



- Perfect structure for Benders' decomposition: iteratively compute value function depending on linking variables
- Benders' decomposition with Lagrangian decomposition inside[10] (curiously, both happen to be the cutting-plane algorithm)
- . . . with (different) other subproblems inside

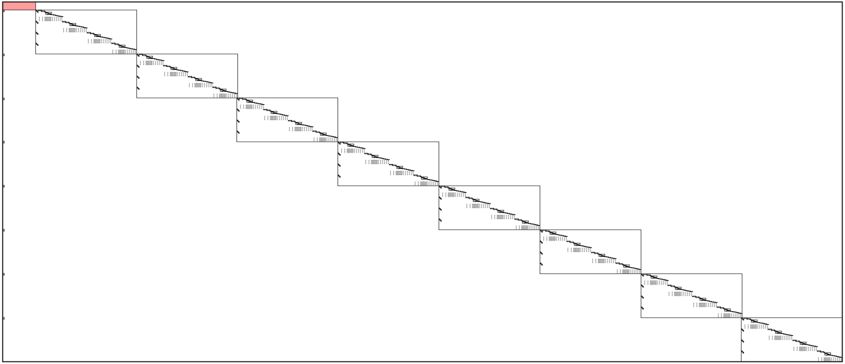
[10] van Ackooij, Malick "Decomposition algorithm for large-scale two-stage unit-commitment" *Ann. OR*, 2016



- Reformulation trick: split here-and-now variables, add copy constraints
  $\Longrightarrow$ perfect structure for Lagrangian decomposition again
- Lagrangian decomposition with Lagrangian decomposition inside ...
- Which is better? Very hard to say beforehand[8]

- Unit-Commitment is a short-term problem, lacks long-term strategies
- Issue: cost of water (none) / minimum reservoir volume (very low) $\Longrightarrow$ lot of water used $\Longrightarrow$ no water most of the year
- Hydro production most useful for peak shaving every day
- Computing value of water left in the reservoirs at T for each (significant) day of the year
- stochastic dual dynamic programming[11]:
  - proceed forward to construct solution
  - proceed backward to compute/refine lower bounds
  - sample the uncertain parameters, solve a parametric (uncertain) UC problem for each sample
  - construct stochastic cuts out of sampled dual information

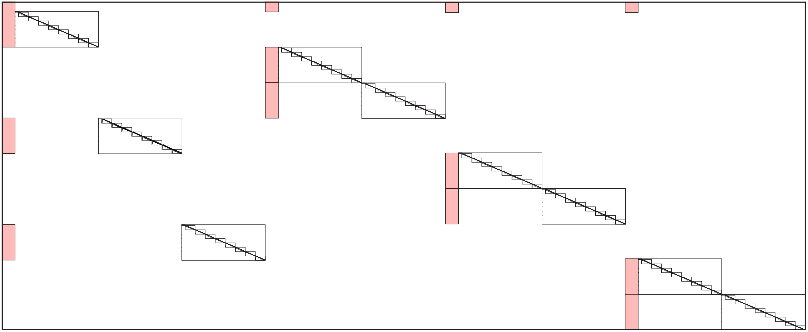
[11] Pereira, Pinto "Multi-stage stochastic optimization applied to energy planning" *Math. Prog.*, 1991



- Each block is scenario-uncertain on some data (RES production) and continuous-uncertain on other data (water inflows/outflows)

- SDDP + Benders + Lagrange + subproblems or
SDDP + Lagrange + Lagrange + subproblems or ...

- The energy system changes all the time, but modifications slow, extremely costly, with huge inertia
- Demand and production subject to very significant uncertainties: climate = RES production + demand, shifts in consumption patterns (EV, cryptocurrencies, ...), new technologies (shale, LED, ...), geo-political factors (energy security), economical factors (boost or boom), regulatory factors (EU energy market, ...), political factors (CO_2 emission treaties, nuclear power, ...), ...
- Planning long-term evolution very hard, yet necessary
- 20/30 years, 2/5 years steps (multi-level recourse), many scenarios



- Huge size, multiple nested structure
- Still OK for either Benders or Lagrange
- Benders + SDDP + Benders + Lagrange + subproblems or
  Lagrange + SDDP + Benders + Lagrange + subproblems ...

- Conceptually, yes. In practice

- Conceptually, yes. In practice I most definitely hope so

  `www.plan4res.eu`

- Is it "easy"? By no means
- Tools for doing this actually don't exist, writing them as I speak
- Large project, 4M€ (so not that large w.r.t. many others), but if we can shave 0.01% of total EU energy cost we re-pay them 1000-fold
- Is it the "typical" use of optimization? By no means, but:
  - hints at some useful techniques, in particular for uncertainty
  - nice idea: it helps if you know how to solve a subproblem
  - two inner levels actually useful in operations
  - if we can solve this, we can almost surely solve yours
- Usual project much more limited $\Longrightarrow$ doable with standard tools

- Conceptually, yes. In practice I most definitely hope so

  `www.plan4res.eu`

- Is it "easy"? By no means
- Tools for doing this actually don't exist, writing them as I speak
- Large project, 4M€ (so not that large w.r.t. many others), but if we can shave 0.01% of total EU energy cost we re-pay them 1000-fold
- Is it the "typical" use of optimization? By no means, but:
  - hints at some useful techniques, in particular for uncertainty
  - nice idea: it helps if you know how to solve a subproblem
  - two inner levels actually useful in operations
  - if we can solve this, we can almost surely solve yours
- Usual project much more limited $\Longrightarrow$ doable with standard tools and maybe a little help from a friend

Beware Mathematicians When They Seem to Speak Simple

Black-box Optimization

PDE-Constrained Optimization

NonLinear Nonconvex Problems

Mixed-Integer Convex (Linear) Problems

An Aside: Multiple Objectives

Do Not Underestimate Mixed-Integer Linear (Convex) Problems

Conclusions: Don't Be Afraid Of Optimization

- Optimization problems are difficult

- Optimization problems are difficult . . . but there are $\neq$ kinds of "difficult"
- Many problems have structure that can be exploited
- First crucial choice: which class of optimization problems
- Trade-off model accuracy vs. model complexity not trivial
- However, apparently very complex problems may not be that difficult if one knows the right set of modelling tricks
- Lots of stable, well-developed software (even open-source), especially for the most "tractable" problems
- Optimal solutions not always needed, "just good" ones may be easy
- Some mad people having fun just with solving optimization problems
- Can be worth a lot of €€€/performance
- If it really helps, it probably can be done

- Learn a bit about it:
  - mathematics seems imposing but not really though (unless you want to)
  - lots of software whose inner workings you only need to partly understand (oftentimes we don't know all the details ourselves)
- Often off-the-shelf tools enough, but have to be used well
- Consider inviting an optimization-savvy friend:
  - (most of us) will typically bend backwards to make it work
  - seeing our toys actually useful to someone a great joy
  - though applications need good methodologies, but good methodologies learn a lot from though applications
  - we won't eat your lunch: too many things we will never know, usually more than happy with our minor slice of the glory cake
- Optimization a minority stake in any good application, but at the right moment it can save your day

Thank You

frangio@di.unipi.it

http://www.di.unipi.it/∼frangio