Integer programming (part 2)

Lecturer: Javier Peña

Convex Optimization 10-725/36-725

Last time: integer programming

Consider the problem

$$\begin{aligned} \min_{x} \quad & f(x) \\ \text{subject to} \quad & x \in C \\ & x_j \in \mathbb{Z},\ j \in J \end{aligned}$$

where $f : \mathbb{R}^n \to \mathbb{R},\ C \subseteq \mathbb{R}^n$ are convex, and $J \subseteq \{1, \ldots, n\}$.

Branch and bound

Algorithm to solve integer programs.
Divide-and-conquer scheme combined with upper and lower bounds for efficiency.

Branch and bound algorithm

Consider the problem

$$\begin{array}{ll} \min\limits_{x} & f(x) \\ \text{subject to} & x \in C \\ & x_j \in \mathbb{Z},\ j \in J \end{array} \qquad \text{(IP)}$$

where $f : \mathbb{R}^n \to \mathbb{R},\ C \subseteq \mathbb{R}^n$ are convex and $J \subseteq \{1, \dots, n\}$.

1. Solve the convex relaxation

$$\begin{array}{ll} \min\limits_{x} & f(x) \\ \text{subject to} & x \in C \end{array} \qquad \text{(CR)}$$

2. (CR) infeasible $\Rightarrow$ (IP) is infeasible. Stop
3. Solution $x^\star$ to (CR) is (IP) feasible $\Rightarrow$ $x^\star$ solves (IP). Stop
4. Solution $x^\star$ to (CR) not (IP) feasible $\Rightarrow$ lower bound for (IP). Branch and recursively solve subproblems.

After branching

Key component of B&B

- After branching compute a lower bound for each subproblem.
- If a lower bound for a subproblem is larger than the current upper bound, no need to consider the subproblem.
- Most straightforward way to compute lower bounds is via a convex relaxation but other methods (e.g., Lagrangian relaxations) can also be used.

Tightness of relaxations

Suppose we have two equivalent formulations
(e.g., facility location)

$$\begin{array}{rl} \min_x & f(x) \\ \text{subject to} & x \in C \\ & x_j \in \mathbb{Z},\ j \in J \end{array} \qquad \begin{array}{rl} \min_x & f(x) \\ \text{subject to} & x \in C' \\ & x_j \in \mathbb{Z},\ j \in J \end{array}$$

with $C \subseteq C'$.

Which one should we prefer?

Outline

Today:

- More about solution techniques
  - Cutting planes
  - Branch and cut
- Two extended examples
  - Best subset selection
  - Least mean squares

Convexification

Consider the special case of an integer program with linear objective

$$\begin{aligned} \min_x \quad & c^\mathsf{T} x \\ \text{subject to} \quad & x \in C \\ & x_j \in \mathbb{Z},\ j \in J \end{aligned}$$

This problem is equivalent to

$$\begin{aligned} \min_x \quad & c^\mathsf{T} x \\ \text{subject to} \quad & x \in S \end{aligned}$$

where $S := \mathrm{conv}\{x \in C : x_j \in \mathbb{Z},\ j \in J\}$.

Special case: integer linear programs

Consider the problem

$$\begin{array}{rl} \min\limits_{x} & c^{\mathsf{T}} x \\ \text{subject to} & Ax \leq b \\ & x_j \in \mathbb{Z},\ j \in J \end{array}$$

Theorem

If A, b are rational, then the set

$$S := \textit{conv}\{x : Ax \leq b,\ x_j \in \mathbb{Z},\ j \in J\}$$

is a polyhedron.

Thus the above integer linear program is equivalent to a linear program.

How hard could that be?

Cutting plane algorithm

We say that the inequality $\pi^\mathsf{T} x \le \pi_0$ is valid for a set S if

$$\pi^\mathsf{T} x \le \pi_0 \ \text{ for all } \ x \in S.$$

Consider the problem

$$\begin{array}{rl} \min\limits_{x} & c^\mathsf{T} x \\ \text{subject to} & x \in C \\ & x_j \in \mathbb{Z},\ j \in J \end{array}$$

and let $S := \text{conv}\{x \in C : x_j \in \mathbb{Z},\ j \in J\}$.

Cutting plane algorithm

Recall: $S = \text{conv}\{x \in C : x_j \in \mathbb{Z},\ j \in J\}$ and want to solve

$$\begin{array}{ll} \min\limits_{x} & c^{\mathsf{T}} x \\ \text{subject to} & x \in C \\ & x_j \in \mathbb{Z},\ j \in J \end{array} \qquad \text{(IP)}$$

Cutting plane algorithm

1. let $C_0 := C$ and compute $x^{(0)} := \underset{x}{\operatorname{argmin}}\{c^{\mathsf{T}} x : x \in C_0\}$
2. for $k = 0, 1, \ldots$

   if $x^{(k)}$ is (IP) feasible then $x^{(k)}$ is an optimal solution. Stop

   else

   find a valid inequality (π, π_0) for S that cuts off $x^{(k)}$

   let $C_{k+1} := C_k \cap \{x : \pi^{\mathsf{T}} x \leq \pi_0\}$

   compute $x^{(k+1)} := \operatorname{argmin}_x\{c^{\mathsf{T}} x : x \in C_{k+1}\}$

   end if

   end for

A valid inequality is also called a cutting plane or a cut

A bit of history on cutting planes

In 1954 Dantzig, Fulkerson, and Johnson pioneered the cutting plane approach for the traveling salesman problem. In 1958 Gomory proposed a general-purpose cutting plane method to solve any integer linear program.

For more than three decades Gomory cuts were deemed impractical for solving actual problems. In the early 1990s Sebastian Ceria (at CMU) successfully incorporated Gomory cuts in a branch and cut framework. By the late 1990s cutting planes had become a key component of commercial optimization solvers.

This subject has a long tradition at Carnegie Mellon and is associated to some of our biggest names: Balas, Cornuejols, Jeroslow, Kilinc-Karzan and many of their collaborators.

Gomory cuts (1958)

This class of cuts is based on the following observation:

if $a \leq b$ and a is an integer then $a \leq \lfloor b \rfloor$.

Suppose

$$S \subseteq \left\{ x \in \mathbb{Z}_+^n : \sum_{j=1}^{n} a_j x_j = a_0 \right\}$$

where $a_0 \notin \mathbb{Z}$. The Gomory fractional cut is

$$\sum_{j=1}^{n} (a_j - \lfloor a_j \rfloor) x_j \geq a_0 - \lfloor a_0 \rfloor.$$

There is a rich variety of extensions of the above idea: Chvatal cuts, split cuts, lift-and-project cuts, etc.

Lift-and-project cuts

Theorem (Balas 1974)

Assume $C = \{x : Ax \leq b\} \subseteq \{x : 0 \leq x_j \leq 1\}$. *Then* $C_j :=$*conv*$\{x \in C : x_j \in \{0, 1\}\}$ *is the projection of the polyhedron* $\mathcal{L}_j(C)$ *defined by the following inequalities*

$$\begin{array}{rcl} Ay & \leq & \lambda b \\ Az & \leq & (1-\lambda)b \\ y+z & = & x \\ \lambda & \leq & 1 \\ \lambda & \geq & 0. \end{array}$$

Suppose $\tilde{x} \in C$ but $\tilde{x} \notin C_j$. To find a cut separating $\tilde{x}$ from C_j solve

$$\begin{array}{rl} \min\limits_{x,y,z,\lambda} & \|x - \tilde{x}\| \\ \text{subject to} & (x, y, z, \lambda) \in \mathcal{L}_j(C) \end{array}$$

Branch and cut algorithm

Combine strengths of both branch and bound and cutting planes.

Consider the problem

$$\begin{array}{rl} \min\limits_{x} & f(x) \\ \text{subject to} & x \in C \\ & x_j \in \mathbb{Z},\ j \in J \end{array} \tag{IP}$$

where $f : \mathbb{R}^n \to \mathbb{R},\ C \subseteq \mathbb{R}^n$ are convex and $J \subseteq \{1, \ldots, n\}$.

Branch and cut algorithm

Branch and cut algorithm

1. Solve the convex relaxation

$$\begin{array}{rl} \min\limits_{x} & f(x) \\ \text{subject to} & x \in C \end{array} \tag{CR}$$

2. (CR) infeasible $\Rightarrow$ (IP) is infeasible.
3. Solution $x^\star$ to (CR) is (IP) feasible $\Rightarrow$ $x^\star$ solution to (IP).
4. Solution $x^\star$ to (CR) is not (IP) feasible.
   Choose between two alternatives
   4.1 Add cuts and go back to step 1
   4.2 Branch and recursively solve subproblems

Integer programming technology

- State-of-the-art solvers (Gurobi, CPLEX, FICO) rely on extremely efficient implementations of numerical linear algebra, simplex, interior-point, and other algorithms for convex optimization.
- For mixed integer optimization, most solvers use a clever type of branch and cut algorithm. They rely extensively on convex relaxations. They also take advantage of warm starts as much as possible.
- Some interesting numbers
  Speedup in algorithms 1990–2016: over 500,000
  Speedup in hardware 1990–2016: over 500,000
  Total speedup over 250 billion $= 2.5 \cdot 10^{11}$

Best subset selection

Assume $X = \begin{bmatrix} x^1 & \cdots & x^p \end{bmatrix} \in \mathbb{R}^{n \times p}$ and $y \in \mathbb{R}^n$.

Best subset selection problem:

$$\begin{array}{rl} \min\limits_{\beta} & \frac{1}{2}\|y - X\beta\|_2^2 \\ \text{subject to} & \|\beta\|_0 \le k \end{array}$$

Here $\|\beta\|_0 :=$ number of nonzero entries of β.

Best subset selection

Integer programming formulation

$$\begin{array}{ll} \min_{\beta,z} & \frac{1}{2}\|y - X\beta\|_2^2 \\ \text{subject to} & |\beta_i| \le M_i \cdot z_i,\ i = 1,\ldots,p \\ & \sum_{i=1}^p z_i \le k \\ & z_i \in \{0,1\},\ i = 1,\ldots,p. \end{array}$$

Here M_i is some a priori known bound on $|\beta_i|$ for $i = 1, \ldots, p$.
They can be computed via suitable preprocessing of X, y.

A clever way to get good feasible solutions

Bertsimas, King, and Mazumder

Consider the problem

$$\min_{\beta} g(\beta) \text{ subject to } \|\beta\|_0 \leq k$$

where $g : \mathbb{R}^p \to \mathbb{R}$ is smooth convex and ∇g is L-Lipschitz.

Best subset selection corresponds to $g(\beta) = \frac{1}{2}\|X\beta - y\|_2^2$.

Observation

For $u \in \mathbb{R}^p$ the vector

$$H_k(u) = \underset{\beta:\|\beta\|_0 \leq k}{\operatorname{argmin}} \|\beta - u\|_2^2$$

is obtained by retaining the k largest entries of u.

Discrete first-order algorithm

Bertsimas et al.

Consider the problem

$$\min_{\beta} g(\beta) \text{ subject to } \|\beta\|_0 \leq k$$

where $g : \mathbb{R}^p \to \mathbb{R}$ is smooth convex and ∇g is L-Lipschitz.

1. start with some $\beta^{(0)}$
2. for $i = 0, 1, \ldots,$
   $\beta^{(i+1)} = H_k\left(\beta^{(i)} - \frac{1}{L}\nabla g(\beta^{(i)})\right)$
   end for

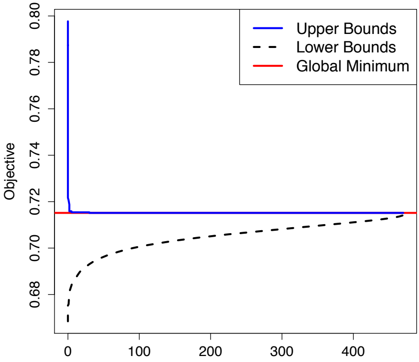
The above iterates satisfy $\beta^{(i)} \to \bar{\beta}$ where

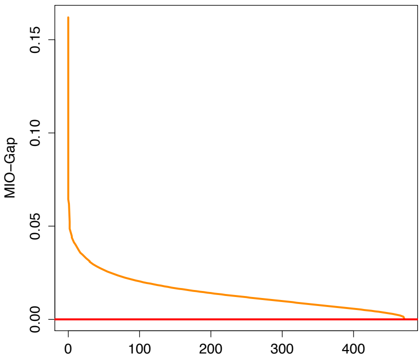
$$\bar{\beta} = H_k\left(\bar{\beta} - \frac{1}{L}\nabla g(\bar{\beta})\right).$$

This is a kind of "local" solution to the above minimization problem.

Computational results (Bertsimas et al.)

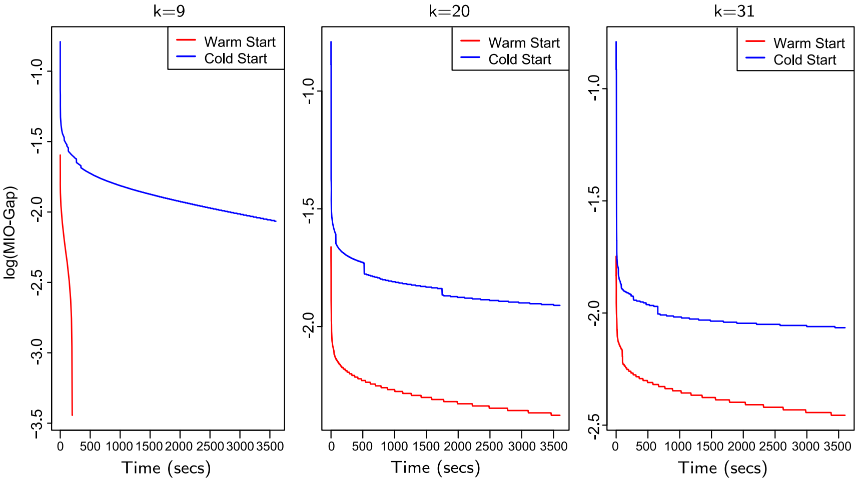
Diabetes dataset, $n = 350, p = 64, k = 6$





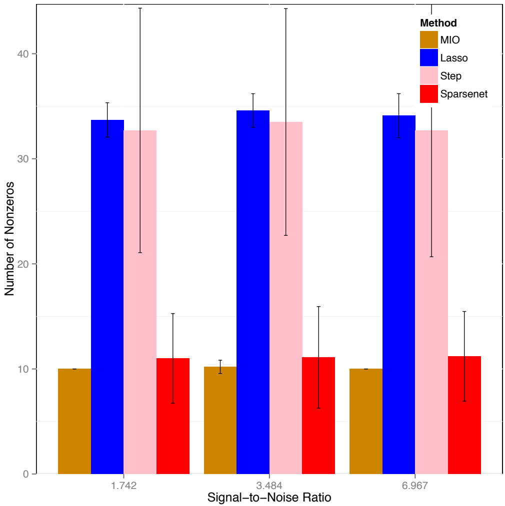
Computational results (Bertsimas et al.)

Cold vs Warm Starts



Computational results (Bertsimas et al.)

Sparsity detection (synthetic datasets)



Least median of squares regression

Assume $X = \begin{bmatrix} x^1 & \cdots & x^p \end{bmatrix} \in \mathbb{R}^{n \times p}$ and $y \in \mathbb{R}^n$.

Given $\beta \in \mathbb{R}^p$ let $r := y - X\beta$

Observe

- Least squares (LS): $\beta_{LS} := \underset{\beta}{\operatorname{argmin}} \sum_i r_i^2$
- Least absolute deviation (LAD): $\beta_{LAD} = \underset{\beta}{\operatorname{argmin}} \sum_i |r_i|$

Least Median of Squares (LMS)

$$\beta_{LMS} := \underset{\beta}{\operatorname{argmin}} (\text{median}|r_i|).$$

Least quantile regression

Least Quantile of Squares (LQS)

$$\beta_{LQS} := \underset{\beta}{\operatorname{argmin}} |r_{(q)}|$$

where $r_{(q)}$ is the qth ordered absolute residual:

$$|r_{(1)} \leq |r_{(2)}| \leq \cdots |r_{(n)}|.$$

Key step in the formulation

Use binary and auxiliary variables to encode the relevant condition

$$|r_i| \leq |r_{(q)}| \text{ or } |r_i| \geq |r_{(q)}|$$

for each entry i of r.

Least quantile regression

Integer programming formulation

$$
\begin{aligned}
\min_{\beta,\mu,\bar{\mu},z,\gamma} \quad & \gamma \\
\text{subject to} \quad & \gamma \leq |r_i| + \bar{\mu}_i \; i = 1, \ldots, n \\
& \gamma \geq |r_i| - \mu_i, \; i = 1, \ldots, n \\
& \bar{\mu}_i \leq M z_i, \; i = 1, \ldots, n \\
& \mu_i \leq M(1 - z_i), \; i = 1, \ldots, n \\
& \sum_{i=1}^{p} z_i = q \\
& \mu_i, \bar{\mu}_i \geq 0, \; i = 1, \ldots, n \\
& z_i \in \{0, 1\}, \; i = 1, \ldots, p.
\end{aligned}
$$

First-order algorithm

Bertsimas & Mazumder

Observe

$$|r_{(q)}| = |y_{(q)} - x_{(q)}^{\mathsf{T}}\beta| = H_q(\beta) - H_{q+1}(\beta)$$

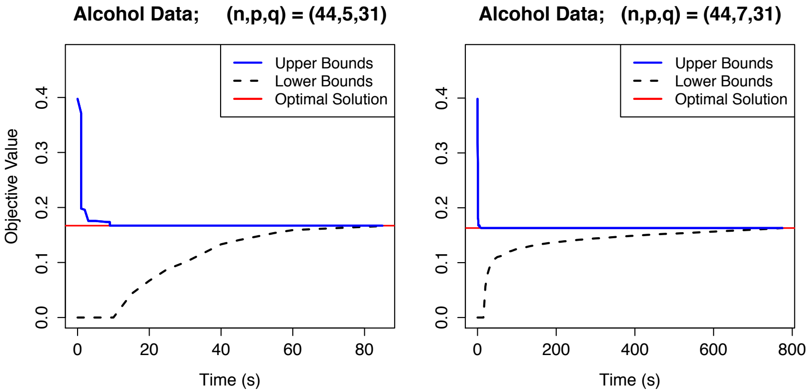
where

$$H_q(\beta) = \sum_{i=q}^{n} |y_{(i)} - x_{(i)}^{\mathsf{T}}\beta| = \begin{array}{rl} \max\limits_{w} & \sum\limits_{i=1}^{n} w_i |y_i - x_i^{\mathsf{T}}\beta| \\ \text{subject to} & \sum\limits_{i=1}^{n} w_i = q \\ & 0 \leq w_i \leq 1,\ i = 1, \ldots, n. \end{array}$$

The function $H_q(\beta)$ is convex.

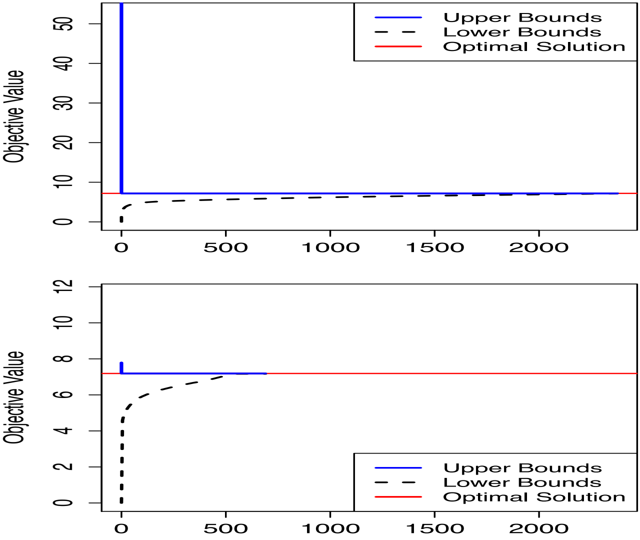
Subgradient algorithm yields a local min to $H_q(\beta) - H_{q+1}(\beta)$.

Computational results (Bertsimas & Mazumder)



Computational results (Bertsimas & Mazumder)

Cold vs Warm Starts



References and further reading

- Belotti, Kirches, Leyffer, Linderoth, Luedke, and Mahajan (2012), "Mixed-integer nonlinear optimization"
- Bertsimas and Mazumder (2016), "Best subset selection via a modern optimization lens"
- Bertsimas, King, and Mazumder (2014), "Least quantile regression via modern optimization"
- Conforti, Cornuejols, and Zambelli (2014), "Integer programming"
- Wolsey (1998), "Integer programming"