10-725/36-725: Convex Optimization **Fall 2016**

Lecture 24: November 28

Lecturer: Lecturer: Javier Peña *Scribes: Hiroaki Hayashi, Jayanth Koushik*

24.1 Review of coordinate descent

Given $f = g + \sum h_i(x_i)$ where g is convex, differentiable and each h_i is convex, minimizing f in the following way is called coordinate descent.

$$\begin{aligned}
x_1^{(k)} &\in \arg\min_{x_1} f(x_1, x_2^{(k-1)}, x_3^{(k-1)}, ..., x_n^{(k-1)}) \\
x_2^{(k)} &\in \arg\min_{x_2} f(x_1^{(k)}, x_2, x_3^{(k-1)}, ..., x_n^{(k-1)}) \\
x_3^{(k)} &\in \arg\min_{x_3} f(x_1^{(k)}, x_2^{(k)}, x_3, ..., x_n^{(k-1)}) \\
&\dots \\
x_n^{(k)} &\in \arg\min_{x_n} f(x_1^{(k)}, x_2^{(k)}, x_3^{(k)}, ..., x_n)
\end{aligned} \tag{24.1}$$

The main idea is to use the previously updated parameter $x_i^{(k)}$ for the minimization of the following problems within the kth update.

24.2 Mixed integer programs

24.2.1 Problem definition

When an optimization problem contains variables that are restricted to be integers, it is called an integer program. When all the variables are restricted to be integers, it is called a pure integer program. In general, the following optimization formulates integer programs.

$$\begin{aligned}
\min_x \quad & f(x) \\
\text{subject to} \quad & x \in C \\
& x_j \in \mathbb{Z}, \quad j \in J
\end{aligned} \tag{24.2}$$

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

24.2.2 Binary variables

Some variables might take only two values, which can correspond to yes / no options. In these cases, such variables are called binary variables.

Combinatorial optimizations have straight-forward relations to integer programs, in that the former problems can be reformulated into the latter by introducing binary variables. See some examples in the next section.

24.3 Examples of integer programs

24.3.1 Knapsack problem

Knapsack problem is a classic optimization problem that aims to maximize the total value of items under the constraint that the total volume is limited. The problem is written using a binary variable x. The value of x_j corresponds to whether you pick the jth item or not.

$$\begin{aligned} \max_x \quad & c^T x \\ \text{subject to} \quad & a^T x \leq b \\ & x_j \in \{0,1\} \quad j = 1 \dots n \end{aligned} \tag{24.3}$$

where c_j, a_j are the value and volume for the jth item.

24.3.2 Assignment problem

Another example is the job assignment problem. The goal is to assign n people n jobs, where you have c_{ij} indicating the cost for the person i to perform job j. This results into the following optimization:

$$\begin{aligned} \min_x \quad & \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij} \\ \text{subject to} \quad & \sum_{i=1}^{n} x_{ij} = 1, \quad j = 1 \dots n \\ & \sum_{j=1}^{n} x_{ij} = 1, \quad i = 1 \dots n \\ & x_{ij} \in \{0,1\} \quad i = 1 \dots n, \quad j = 1 \dots n \end{aligned} \tag{24.4}$$

First and second constraints enforce that each person is assigned exactly one job, and vice versa.

24.3.3 Facility location problem

A more complex example is the facility location problem, where the objective is to minimize the operation cost of facilities and transportation costs for clients from certain facilities. Two binary variables x, y are defined for this problem. x serves as the indicator whether client i is served from depot j, and y denotes whether the depot j is open or not. With these variables, the corresponding optimization formulation is obtained.

$$\begin{aligned} \min_{x,y} \quad & \sum_{j=1}^{n} f_j y_j + \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij} x_{ij} \\ \text{subject to} \quad & \sum_{j=1}^{n} x_{ij} = 1, \quad i = 1 \dots n \\ & x_{ij} \le y_j, \quad i = 1 \dots m, \quad j = 1 \dots n \\ & x_{ij} \in \{0, 1\}, \quad i = 1 \dots m, \quad j = 1 \dots n \\ & y_j \in 0, 1, \quad j = 1 \dots n \end{aligned} \tag{24.5}$$

The first constraint ascertains that each client has one depot as the server. The second constraint implies that the facility j has to be open in order to serve for the customer i. This condition yields mn constraints.

Alternatively, one can think of a different constraint which is a "marginalized" form:

$$\sum_{i=1}^{n} x_{ij} \le m y_j, j = 1 \dots n \tag{24.6}$$

This reduces the number of constraints significantly, however, the former constraints are preferred. See the later sections for detail.

24.3.4 K-means and K-medoids clustering

K-means is an algorithm to find partitions $S_i, i = 1 \dots n$ in the given data by minimizing following objective:

$$\begin{aligned} & \sum_{i=1}^{K} \sum_{j \in S_i} \|x^{(j)} - \mu^{(j)}\|^2 \\ \text{where} \quad & \mu^{(i)} = \frac{1}{|S_i|} \sum_{j \in S_i} x^{(i)} \end{aligned} \tag{24.7}$$

μ is the centroid for each partition. The problem is easier to solve when one chooses centroids from datapoints rather than by computing in above way, which yields the following K-medoids clustering:

$$\sum_{i=1}^{K} \sum_{j \in S_i} \|x^{(j)} - y^{(i)}\|^2 \tag{24.8}$$

by selecting $y^{(i)} \in \{x^{(j)} : j \in S_i\}$. This problem can be formulated into an integer program.

First, define $d_{ij} = \|x^{(i)} - x^{(j)}\|^2$, and introduce two variables as follows:

$$\begin{aligned} w_i &= \begin{cases} 1 & \text{if choose } x^{(i)} \text{ as a centroid.} \\ 0 & \text{otherwise.} \end{cases} \\ z_{ji} &= \begin{cases} 1 & \text{if } \quad x^{(j)} \quad \text{in the cluster with centroid } x^{(i)} \\ 0 & \text{otherwise.} \end{cases} \end{aligned} \tag{24.9}$$

Then the K-medoids problem is expressed as below optimization problem:

$$\begin{aligned} \min_{w,z} \quad & \sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij} z_{ji} \\ \text{subject to} \quad & z_{ji} \leq w_i \\ & \sum_{i=1}^{n} w_i = k \\ & w_i \in 0, 1, \quad i = 1 \dots n \\ & z_{ji} \in 0, 1, \quad j, i = 1 \dots n \end{aligned} \tag{24.10}$$

Similar to the facility location problem, the first constraint enforces that $x^{(i)}$ has to be chosen as the centroid before achieving $z_{ji} = 1$.

24.3.5 Best subset selection

Consider a regression setting with design matrix $X \in \mathbb{R}^{n \times p}$ and $y \in \mathbb{R}^n$. The best subset selection problem is

$$\begin{aligned} \min_{\beta} \quad & \frac{1}{2}\|y - X\beta\|_2^2 \\ \text{subject to} \quad & \|\beta\|_0 \leq k \end{aligned} \tag{24.11}$$

Here $\|\beta\|_0$ is the number of non-zero elements of β and k is some non-negative integer. This problem can be formulated as an integer program by introducing additional binary variables z, and a scalar M which represents an a-priori bound on the elements of β. Specifically, we have

$$\begin{aligned} \min_{\beta,z} \quad & \frac{1}{2}\|y - X\beta\|_2^2 \\ \text{subject to} \quad & |\beta_i| \leq M z_i \quad i = 1 \dots p \\ & z_i \in \{0, 1\} \quad i = 1 \dots p \\ & \sum_{i=1}^{p} z_i \leq k \end{aligned} \tag{24.12}$$

24.3.6 Least median of squares regression

Consider the same regression setting from the previous section, and denote the residuals by $r = y - X\beta$. Then the least median of squares (LMS) regression problem is given by $\min_\beta \text{median}(|r|)$. Here the absolute value is element-wise, and the median function returns the median of the elements in the vector.

Exercise: Formulate the LMS regression problem as an integer program.

24.4 Solving integer programs

24.4.1 Hardness of integer programs

Integer programs are much harder to solve than convex programs. General mixed integer programming is NP-hard implying that there are no known (and possibly no possible) polynomial time algorithms for solving integer programs. They are among the hardest (computationally) class of problems studied. Further, even verifying possible solutions to integer programs is a challenging task i.e. given a candidate solution to an integer program, it is not trivial to check whether or not the solution is optimal.

By ignoring the integer constraints, we get the convex relaxation of the integer program, and solving this relaxation gives a lower bound on its optimal value. This is somewhat limited in utility however, since 1) rounding the solution to the convex relaxation may not be easy, and may result in a solution far from the optimal, and 2) the optimal solution to the convex relaxation may be far from the optimal solution to the original problem.

24.4.2 Algorithmic template for solving integer programs

A naive template for solving integer programs is based on bounding the optimal solution. Let z be the optimal value of an integer program, and let $\bar{z}$, $\underline{z}$ be upper and lower bounds respectively. If $\bar{z}$ and $\underline{z}$ are close to each other, then they represent good approximations to z. More generally, we can find a sequence of upper bounds $\bar{z}_1 \geq \bar{z}_2, \ldots, \bar{z}_s \geq z$; and a sequence of lower bounds $\underline{z}_1 \leq \underline{z}_2, \ldots, \underline{z}_t \leq z$. To get an ϵ optimal solution, we can stop when $\bar{z}_s - \underline{z}_t \leq \epsilon$.

Upper bounds, also called primal bounds can be found by picking any feasible point for the program. In some cases, this is easy to do but this is not always the case.

Lower bounds, also called dual bounds are commonly found via relaxations which are discussed next.

24.5 Relaxations

Consider a general optimization problem

$$\min_{x \in X} f(x) \tag{24.13}$$

A relaxation of this problem is any optimization problem

$$\min_{x \in Y} g(x) \tag{24.14}$$

such that

- $X \subseteq Y$
- $g(x) \leq f(x)$ for all $x \in X$

Note that we do not require the relaxation to optimize the same criterion as the original problem. However, because of the two conditions, the optimal value of the relaxation is a lower bound on the optimal value of the original problem. We consider two kinds of relaxations.

24.5.1 Convex relaxations

Consider again the general mixed integer program

$$\begin{aligned} \min_x \quad & f(x) \\ \text{subject to} \quad & x \in C \\ & x_j \in \mathbb{Z} \quad j \in J \end{aligned} \tag{24.15}$$

Here $f : \mathbb{R}^n \to \mathbb{R}$ and $C \subseteq \mathbb{R}^n$ are both convex, and $J = \{1, \ldots, n\}$. The convex relaxation of this program is obtained by simply dropping the integer constraints to get

$$\begin{aligned} \min_x \quad & f(x) \\ \text{subject to} \quad & x \in C \end{aligned} \tag{24.16}$$

24.5.2 Lagrangian relaxations

Now consider a problem of the form

$$\begin{aligned} \min_x \quad & f(x) \\ \text{subject to} \quad & Ax \leq b \\ & x_j \in \mathbb{Z} \quad x \in X \end{aligned} \tag{24.17}$$

where X includes both convex and integer constraints. The Lagrangian relaxation is obtained by shifting some of the constraints to the objective. Specifically we take some $u \geq 0$ and define

$$\begin{aligned} L(u) = \quad \min_x \quad & f(x) + u^T(Ax - b) \\ \text{subject to} \quad & x \in X \end{aligned} \tag{24.18}$$

This is indeed a relaxation. The constraint set is becoming larger. And for any x feasible for the original problem, we have $Ax \leq b$, so $f(x) + u^T(Ax - b) \leq f(x)$ (since $u \geq 0$). So $L(u)$ is a lower bound for any $u \geq 0$, and the best lower bound can be obtained by solving the dual problem $\max_{u \geq 0} L(u)$. Note that this is a concave optimization problem since $L(u)$ is the point-wise minimization of convex functions.

As an exercise, let us consider the Lagrangian relaxation of the facility location problem. Here it turns out to be beneficial to consider the first formulation of the problem, and move the summation constraints to the objective. So, we have for an unconstrained v

$$\begin{aligned} L(v) = \quad \min_{x,y} \quad & \sum_{j=1}^{n} f_j y_j + \sum_{i=1}^{m} \sum_{j=1}^{n} (c_{ij} - v_i) x_{ij} + \sum_{i=1}^{m} v_i \\ \text{subject to} \quad & x_{ij} \leq y_j \quad i = 1 \ldots m, j = 1 \ldots n \\ & x_{ij}, y_j \in \{0, 1\} \quad i = 1 \ldots m, j = 1 \ldots n \end{aligned} \tag{24.19}$$

It turns out that the Lagrangian relaxation is easy to solve, and the solution is given by

$$x_{ij}(v) = \begin{cases} 1 & \text{if } c_{ij} - v_i < 0 \text{ and } \sum_l (c_{lj} - v_l)^- + f_j < 0 \\ 0 & \text{otherwise} \end{cases} \tag{24.20}$$

$$y_j(v) = \begin{cases} 1 & \text{if } \sum_l (c_{lj} - v_l)^- + f_j < 0 \\ 0 & \text{otherwise} \end{cases} \tag{24.21}$$

This gives a lower bound to the original solution. Furthermore, it turns out that it is easy to compute the subdifferential of $-L(v)$. So the best lower bound can be found by using the subgradient method to solve $\max_v L(v) \equiv \min_v -L(v)$.

24.6 Branch and bound algorithm

The branch and bound method algorithm is the most common method for solving integer programs. It is a divide and conquer algorithm which divides the original problem into many sub-problems. Let X be the constraint set of an integer program, and let $X_1, \ldots, X_k$ be a partition of X. Then the original problem $\min_{x \in X} f(x)$ is equivalent to

$$\min_{i=1,\ldots,k} \min_{x \in X_i} f(x) \tag{24.22}$$

Note that a feasible solution to any of the sub-problems yields an upper bound $u(X)$ to the original problem. To obtain a lower bound, we can obtain a lower bound $l(X_i)$ to each of the sub-problems. And the key observation is that if $l(X_i) \geq u(X)$, then we can ignore the corresponding sub-problem $\min_{x \in X_i} f(x)$. So this leads to a recursive algorithm:

1. If the constraint set is trivial, solve the problem. If the solution is less than the current upper bound, update the upper bound. stop.
2. Find a lower bound to the problem.
3. If the lower bound is greater than the current upper bound, then stop.
4. Split the constraint set, and solve each sub-problem recursively.

The final upper bound is the optimal solution.

References and further reading

[CCZ14] M. Conforti and G. Cornuéjols and G. Zambelli, "Integer Programming" *Vol. 271. Berlin: Springer*, 2014

[L98] L. Wolsey, "Integer Programming. Series in Discrete Mathematics and Optimization" *Wiley Interscience*, 1998

[BKLLLM13] P. Belotti and C. Kirches and S. Leyffer and J. Linderoth and J. Luedtke and A. Mahajan, "Mixed-integer nonlinear optimization" *Acta Numerica, 22*, 2013, pp. 1–131