Lecture 9: More on Simplex Method

Zizhuo Wang

Institute of Data and Decision Analytics (iDDA)
Chinese University of Hong Kong, Shenzhen

Sep 30, 2018

Announcements

- Homework 3 due on Oct 10th

Recap: Simplex Tableau

Simplex tableau is a practical and efficient way to implement the simplex method.

- It is a table of numbers
- Each step in the simplex method procedure can be carried out as some manipulations on the numbers in the simplex tableau

$\mathbf{c}^T - \mathbf{c}_B^T A_B^{-1} A$	$-\mathbf{c}_B^T A_B^{-1} \mathbf{b}$
$A_B^{-1} A$	$A_B^{-1} \mathbf{b}$

- The upper left part is the reduced costs, and the upper right corner is the negative of the objective value
- The lower part is simply a transformation of the constraints. In particular, the right hand side column is the current basic variables

Recap: Simplex Tableau

From a canonical form, each iteration in the simple tableau operation consists of

1. Determine optimality: look at the top row (the reduced costs)
   - If all nonnegative, then the solution is already optimal
   - Otherwise, choose the smallest index j such that $\bar{c}_j < 0$ to be the incoming basis (pivot column)
2. Perform the minimal ratio test (between the current basic variables and the incoming column — positive entries only).
3. Choose the row that attains the smallest ratio to be the pivot row, and determine the outgoing basis (smallest index rule)
4. Update the tableau
   - Divide each entry in the pivot row by the pivot element
   - Add proper multiples of the new pivot row to each other rows, such that all other elements in the pivot column become zeros (including the top row)

Recap: Simplex Tableau

- If there is no positive entry in the pivot column, then the problem is unbounded
- The update of the tableau should always include the top row and also the right hand side $\mathbf{b}$
- One can still proceed normally even if there is a degeneracy solution in the process. As long as the smallest index rule (Bland's rule) is used, it is guaranteed that we will stop at an optimal solution in a finite number of steps

Example

Consider the following simplex tableau:

B	β	0	0	0	δ	2	0
2	-1	1	0	0	η	0	1
4	-1	0	0	1	-1	1	2
3	α	0	1	0	1	1	γ

The entries of α, β, γ, δ and η are unknown parameters. Find sufficient conditions on the ranges of them such that the following statements are true.

B	β	0	0	0	δ	2	0
2	-1	1	0	0	η	0	1
4	-1	0	0	1	-1	1	2
3	α	0	1	0	1	1	γ

1. The current basic solution is optimal for the problem
   - $\beta \geq 0$, $\delta \geq 0$, $\gamma \geq 0$
2. The current basic solution is feasible. At the next iteration, x_5 is the only candidate to enter the basis. And if x_5 enters the basis, x_2 leaves the basis according to Bland's rule.
   - $\delta < 0$, $\beta \geq 0$, $1/\eta \leq \gamma$, $\eta > 0$
3. The current basic solution is feasible, but the problem is unbounded.
   - $\beta < 0$, $\alpha \leq 0$, $\gamma \geq 0$
4. The last row indicates that the problem is infeasible.
   - $\alpha \geq 0$, $\gamma < 0$

Two-Phase Method in Simplex Tableau

For the simplex tableau, when there is no obvious initial basic feasible solution, we still need to use the two-phase method.

To carry out the two-phase methods in the simplex tableau, we need to solve some additional issues.

Phase I:

- The auxiliary problem is not in a canonical form: Although the constraint contains an identity matrix (which is the initial basis), the corresponding objective coefficients are not 0's
- We need to compute the top row (reduced costs)
  - For basic part, the reduced costs are 0
  - For nonbasic part, $\bar{c}_j = c_j - \mathbf{c}_B^T A_B^{-1} A_j = -\mathbf{e}^T A_j$, so the jth reduced cost is the negative of the sum of that column
- This also applies to the initial objective value, which equals the negative of the sum of the right hand side vector.

Two-Phase Method in Simplex Tableau

Suppose we have finished the Phase I Simplex tableau operations, resulting in an optimal solution $(\mathbf{x}^*, 0)$ with optimal value 0.

Then to carry out Phase II in the Simplex tableau:

1. Drop all the columns associated with the auxiliary variables
2. Change the top row to the reduced costs corresponding to the original problem by using the reduced cost formula:

$$\bar{c}_j = c_j - \mathbf{c}_B^T A_B^{-1} A_j$$

   Also compute the current objective value
3. Continue the simplex tableau to solve the original problem

If the optimal solution to the auxiliary problem is degenerate, then replace the basic indices for the auxiliary variables with (any) indices of original variables and recalculate the table.

Example

$$
\begin{array}{lrrrrcr}
\text{minimize} & x_1 & +x_2 & +x_3 & & & \\
\text{subject to} & x_1 & +2x_2 & +3x_3 & & = & 3 \\
 & & -4x_2 & -9x_3 & & = & -5 \\
 & & & +3x_3 & +x_4 & = & 1 \\
 & x_1 & , x_2 & , x_3 & , x_4 & \geq & 0
\end{array}
$$

First, make $\mathbf{b}$ positive and construct the auxiliary problem:

$$
\begin{array}{lrrrrrrrcr}
\text{minimize} & & & & & x_5 & +x_6 & +x_7 & & \\
\text{subject to} & x_1 & +2x_2 & +3x_3 & & +x_5 & & & = & 3 \\
 & & 4x_2 & +9x_3 & & & +x_6 & & = & 5 \\
 & & & +3x_3 & +x_4 & & & +x_7 & = & 1 \\
 & x_1, & x_2, & x_3, & x_4, & x_5, & x_6, & x_7 & \geq & 0
\end{array}
$$

Example Continued

Construct the initial tableau for the auxiliary problem

B	-1	-6	-15	-1	0	0	0	-9
5	1	2	3	0	1	0	0	3
6	0	4	9	0	0	1	0	5
7	0	0	3	1	0	0	1	1

Carry out the simplex method (Step 1):

B	0	-4	-12	-1	1	0	0	-6
1	1	2	3	0	1	0	0	3
6	0	4	9	0	0	1	0	5
7	0	0	3	1	0	0	1	1

Example Continued

Step 2:

B	0	0	-3	-1	1	1	0	-1
1	1	0	-3/2	0	1	-1/2	0	1/2
2	0	1	9/4	0	0	1/4	0	5/4
7	0	0	3	1	0	0	1	1

Step 3:

B	0	0	0	0	1	1	1	0
1	1	0	0	1/2	1	-1/2	1/2	1
2	0	1	0	-3/4	0	1/4	-3/4	1/2
3	0	0	1	1/3	0	0	1/3	1/3

This is optimal for the auxiliary problem. $\mathbf{x} = (1, 1/2, 1/3, 0)$ is a BFS for the original problem ($B = \{1, 2, 3\}$).

Example Continued

B	0	0	0	0	1	1	1	0
1	1	0	0	1/2	1	-1/2	1/2	1
2	0	1	0	-3/4	0	1/4	-3/4	1/2
3	0	0	1	1/3	0	0	1/3	1/3

We drop all the columns for auxiliary variables. Then we recompute the reduced cost for the original problem for $B = \{1, 2, 3\}$:

$$\bar{\mathbf{c}} = \mathbf{c}^T - \mathbf{c}_B^T A_B^{-1} A = (0, 0, 0, -1/12)$$

We also need to compute the current objective value: $11/6$

Now the Simplex tableau becomes:

B	0	0	0	-1/12	-11/6
1	1	0	0	1/2	1
2	0	1	0	-3/4	1/2
3	0	0	1	1/3	1/3

Example Continued

Then we continue from the new simplex tableau:

B	0	0	0	-1/12	-11/6
1	1	0	0	1/2	1
2	0	1	0	-3/4	1/2
3	0	0	1	1/3	1/3

The next pivot:

B	0	0	1/4	0	-7/4
1	1	0	-3/2	0	1/2
2	0	1	9/4	0	5/4
4	0	0	3	1	1

This is optimal. The optimal solution is $\mathbf{x} = (1/2, 5/4, 0, 1)$. The optimal value is $7/4$.

Simplex Method Summary

We have completed our discussions on the simplex method.

- The idea of simplex method (search among the BFS and search among the neighbors) and its justifications
- The algebraic procedures
- The simplex tableau
- Other issues about simplex method (initialization, degeneracy, etc.)

How Good is the Simplex Method

We have learned that simplex method must finish in a finite number of iterations. But *finite* could still be *large*...

- In particular, we know the number of iterations should not exceed the number of BFS, which is bounded by $C(n, m)$. But this is a large number

To answer this question, we introduce some basics about *complexity theory*

Complexity Theory

We count the number of arithmetic operations (add, subtract, multiplication, division, comparison, etc.) that an algorithm needs to solve a problem and call it the *complexity of the algorithm*

When we consider the complexity of an algorithm, we consider the worst-case instance for that algorithm

- For a specific instance, an algorithm may finish much faster than its complexity suggests.

Examples:

- Find the largest element among n numbers has complexity n
- Add two m-by-n matrices has complexity mn
- Multiply two n-by-n matrices in a naive way has complexity n^3

Polynomial-Time Algorithms

The main watershed in complexity is whether an algorithm has a polynomial complexity or not.

- We call an algorithm a *polynomial-time algorithm* if the number of arithmetic operations it takes to solve *any* instance of the problem is bounded by a polynomial of the input size.
- A polynomial-time algorithm is usually deemed to be a practical algorithm
- Otherwise, it is usually deemed impractical

Polynomial vs Non-Polynomial

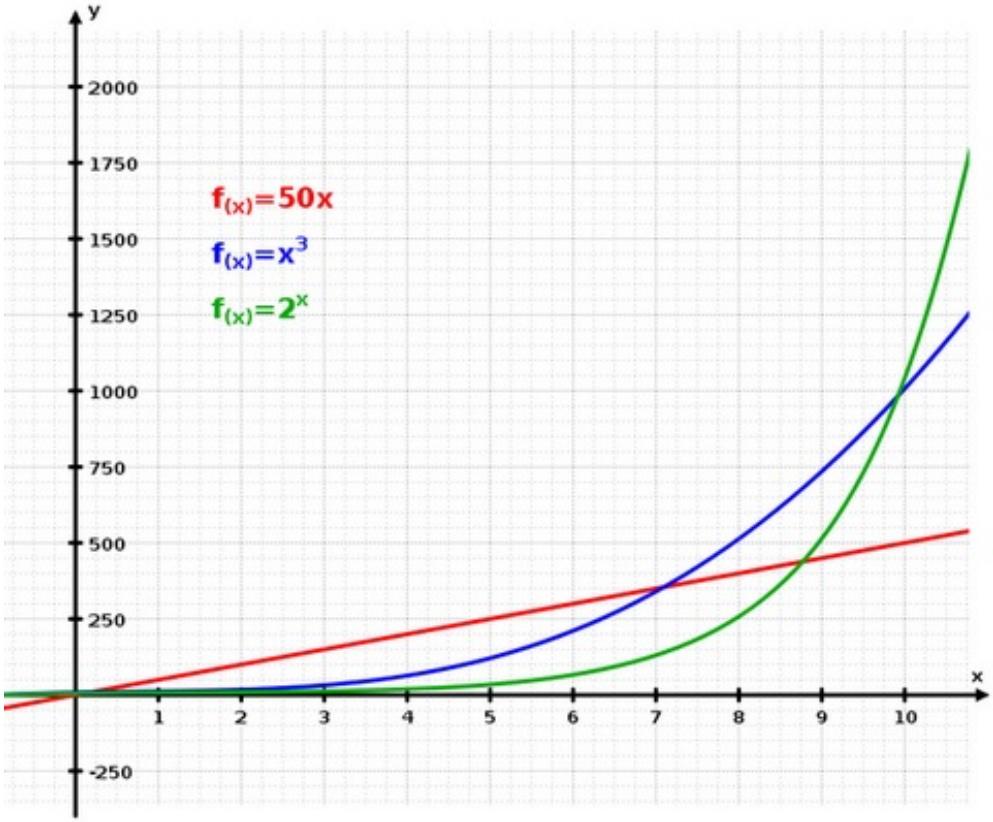

Figure: Polynomial vs Exponential Function

Polynomial-Time Algorithms

Here are some known polynomial algorithms:

- Gaussian elimination for matrix inversion (n^3)
- Fast method for matrix inversion ($\sim n^{2.373}$)
- Naive method for sorting n numbers (n^2)
- Merge sort for sorting n numbers ($n \log n$)

Here are some non-polynomial algorithms:

- Enumeration method for traveling salesman problem ($n!$)
- Dynamic programming for traveling salesman problem ($2^n n^2$)

Complexity of a Problem

Definition

For a problem, if there exists a polynomial-time algorithm, then we say it is a polynomial-time solvable problem, or in P.

There exists a class of problems that researchers haven't found any polynomial-time algorithm (e.g., the traveling salesman problem).

- This class is called NP-Hard problems
- In fact, if one can show that one problem in NP-Hard has a polynomial-time algorithm, then a large subclass of them (NP-Complete) will have polynomial-time algorithms

Million dollar problem:

- Whether there exists polynomial-time algorithm for NP-Hard problem (NP=P?)

Complexity of Simplex Method

Now we consider a standard LP with n variables and m constraints.

What is the complexity of the simplex method?

- Remember that one can configure simplex method by selecting different rules for choosing incoming/outgoing basis. Each will result in a different algorithm and may have different complexity.

Current Result: For all the rules that people have tried, none of them will result in a polynomial-time algorithm.

- In other words, for each configuration that people have tried, there is an instance such that the simplex method with the specified configuration requires exponentially many iterations to find the optimal solution
- It remains an open question whether there is a configuration such that simplex method is a polynomial-time algorithm

Complexity of Simplex Method

For all the known versions of simplex methods, people have found problem instances such that the algorithm will run exponential iterations in terms of n and m to solve that problem (Klee and Minty Example)

- This is very bad news from theoretical point of view

However...

- This is only the worst-case performance
- The practical performance of simplex method is quite good. Typically it needs a small multiple of m iterations to stop
- One can prove that *on average*, simplex method stops in a polynomial number of iterations
- It is still one of the most widely-used algorithms

Complexity of LP

Now we know that simplex method may not be a polynomial-time algorithm, but this doesn't mean that LP is not polynomial-time solvable. After all, simplex method is just one possible algorithm for solving LP.

Whether LP is in P was a major maths problem in the 20th century. It was finally solved by Soviet Union mathematician Khachiyan in 1979, who showed a first polynomial-time algorithm for LP. His method is called the *ellipsoid method*.

- However, although ellipsoid method is a polynomial-time algorithm, it is extremely slow in practice. Therefore, it is more of a theoretical contribution than a practical one.

Can we do better both theoretically and practically?

The Answer is Yes

Later in the course, we will introduce another algorithm for LP called the *interior point method*.

- Before we get to that, we need to first study some important theories for LP: Duality theory.
- We will discuss the duality theory in the next two weeks.