SOLVING DEGENERATE PROBLEM BY USING SIMPLEX METHOD

Saad Shakir Mahmood

Departemen Matematika Program Pasca Sarjana
Universitas Muhammadiyah Surakarta
Jl. Ahmad Yani, Tromol Pos, 1 Pabelan, Kartasura, Surakarta 67102

ABSTRACT

Suatu program linear dikatakaan dalam bentuk degenerasi jika pada penyelesaian fisibel terdapat satu atau lebih dari variabel basisnya berzeroai zero. Degenerasi merupakan masalah dalam aplikasi dimana metode simplex tidak dapat menyelesaikan program linear karena akan terdapat cycle dalam program. Tujuan dari penelitian ini adalah untuk memodifikasikan system kanonikal sehingga dapat menghapus masalah degenerasi dengan cara menambahkan variabel yang diberikan name variable anti cycle ke kendala yang memiliki ruas kanan sama dengan zero, dan juga menambahkan suatu kendala yang membataskan zeroai variabel anti cycle tersebut, dan kemudian diselesaikan dengan metode simplex.

Kata Kunci: *linear program, degeneracy, simplex method*

INTRODUCTION

The simplex method is the most common way to solve large LP problems. Simplex is a mathematical term. In one dimension, a simplex is a line segment connecting two points. In two dimen- sions, a simplex is a triangle formed by joining the points. A three- dimensional simplex is a four-sided pyramid having four corners. The underlying concepts are geometrical, but the solution algo- rithm, developed by George Dantzig in 1947, is an algebraic procedure. As with the graphical method, the simplex method finds the most attractive corner of the feasible region to solve the LP prob- lem. Remember, any LP problem having a solution must have an optimal solution that corresponds to a corner, although there may be multiple or alternative optimal solutions. Simplex usually starts at the corner that represents doing noth- ing. It moves to the neighboring corner that best improves the solution. It does this over and over again, making the greatest possible improvement each time. When no more improvements can be made, the most attractive corner corresponding to the optimal solution has been found.

It is will known that every linear programming problem can be perturbed into non-degenerate problems in J. V. Robert (2008). The original proof that the simplex algorithm would converge to an optimal solution based on the non-dege- neracy assumption. For such a problem, Dantzig's prescription for the simplex algorithm was guaranteed converges to an optimal solution or show that the optimum was unbounded. If the problem does not satisfy the non degenerate assumption, then there is a possibility that the simplex algorithm would not converge to the optimal solution, that is, it would be cycle. If the repeated basis happened to be the optimal one, the simplex method would not so indicate. Problems that did not satisfy the non degenerate assumption were easy to construct, but to find one that did not converge took some effort. The first instance of the linear programming problem that was shown to cycle is the one constructed by Hoffman in V. Chavatal. (1983).

All commercial LP software that we are a ware of apply rules for handling degeneracy, breaking ties, perturbation techniques, and composite primal and dual computations that enable the computer-based simplex algorithm to converge to an optimal solution even if the given problem exhibits classical cycling.

The general form of LP is as follows

Maximize $z = c^T x$

Subject to $Ax \leq b$

$x \geq 0$

Or

Minimize $z = c^T x$

Subject to $Ax \geq b$

$x \geq 0$

Were $c, x, b \in R^n$ and $A \in R^{nxn}$

A linear programming problem is a degenerate if there exists at least one of the basic variables is zero.

A degenerate system could cause difficulties during performing the simplex method. Degeneracy may become evident in the simplex method, when leaving variable is being selected in the iterative process, under the pivot column which determines the leaving variable if there is a tie arbitrary selection of one of these variables may result in one or more variables becoming zero in the next iteration and the problem become degenerate, and in this case, it is usual that one or more of the subsequent pivots will be degenerate, and return to a case that has appeared before, in which case the simplex enters an infinite loop and never attains to the optimal solution, and this behavior is called " Cycling". Therefore, if the simplex method cycles, then all the pivots within the cycle must be degenerate, since the objective function value never changes. Hence, it follows that all the pivots within the cycle must have the some objective function value, i.e. all of these pivots must be degenerate. In practice, degeneracy is very common, but cycling is rare. In fact, it is so rare that most efficient implementations do not take precautions against it.

To illustrate the cycling case, we consider the following problem;

Maximize $z = 10x_1 - 57x_2 - 9x_3 - 24x_4$

subject to

$$0.5x_1 - 5.5x_2 - 2.5x_3 + 9x_4 \leq 0$$
$$0.5x_1 - 1.5x_2 - 0.5x_3 + x_4 \leq 0$$
$$x_1 \leq 1$$

$$all\ variables \geq 0$$

The canonical system is as follows

Maximize

$$z = 10x_1 - 57x_2 - 9x_3 - 24x_4 + 0.slack1 + 0.slack2 + 0.slack3$$

subject to

$$0.5x_1 - 5.5x_2 - 2.5x_3 + 9x_4 + slack1 = 0$$
$$0.5x_1 - 1.5x_2 - 0.5x_3 + x_4 + slack2 = 0$$
$$x_1 + slack3 = 1$$

Iteration 1

Max.z		x_1	x_2	x_3	x_4	**Slack1**	**Slack2**	**Slack3**	**RHS**
c_j		10	-57	-9	-24	0	0	0	
	$c_j - z_j$	10	-57	-9	-24	0	0	0	z = 0
0	Slack1	0.5	-5.5	-2.5	9	1	0	0	0
0	Slack2	0.5	-1.5	-0.5	1	0	1	0	0
0	Slack3	1	0	0	0	0	0	1	1

Iteration 2

Max.z		x_1	x_2	x_3	x_4	**Slack1**	**Slack2**	**Slack3**	**RHS**
c_j		10	-57	-9	-24	0	0	0	
	$c_j - z_j$	0	53	41	-204	-20	0	0	z =
10	x_1	1	-11	-5	18	2	0	0	0
0	Slack2	0	4	2	-8	-1	1	0	0
0	Slack3	0	11	5	-18	-2	0	1	1

Iteration 3

Max.z		x_1	x_2	x_3	x_4	**Slack1**	**Slack2**	**Slack3**	**RHS**
c_j		10	-57	-9	-24	0	0	0	
	$c_j - z_j$	0	0	14.5	-98	-6.75	-13.25	0	z = 0
10	x_1	1	0	0.5	-4	-0.75	2.75	0	0
-57	x_2	0	1	0.5	-2	-0.25	0.25	0	0
0	Slack3	0	0	-0.5	4	0.75	-2.75	1	1

Iteration 4

Max.z		x_1	x_2	x_3	x_4	**Slack1**	**Slack2**	**Slack3**	**RHS**
c_j		10	-57	-9	-24	0	0	0	
	$c_j - z_j$	-29	0	0	18	15	-93	0	z = 0
-9	x_3	2	0	1	-8	-1.5	5.5	0	0
-57	x_2	-1	1	0	2	0.5	-2.5	0	0
0	Slack3	1	0	0	0	0	0	1	1

Iteration 5

Max.z		x_1	x_2	x_3	x_4	**Slack1**	**Slack2**	**Slack3**	**RHS**
c_j		10	-57	-9	-24	0	0	0	
	$c_j - z_j$	-20	-9	0	0	10.5	-70.5	0	z = 0
-9	x_3	-2	4	1	0	0.5	-4.5	0	0
-24	x_4	-0.5	0.5	0	1	0.25	-1.25	0	0
0	Slack3	1	0	0	0	0	0	1	1

Iteration 6

Max.z		x_1	x_2	x_3	x_4	**Slack1**	**Slack2**	**Slack3**	**RHS**
c_j		10	-57	-9	-24	0	0	0	
	$c_j - z_j$	22	-93	-21	0	0	24	0	z = 0
0	Slack1	-4	8	2	0	1	-9	0	0
-24	x_4	0.5	-1.5	-0.5	1	0	1	0	0
0	Slack3	1	0	0	0	0	0	1	1

Iteration 7

Max.z		x_1	x_2	x_3	x_4	**Slack1**	**Slack2**	**Slack3**	**RHS**
c_j		10	-57	-9	-24	0	0	0	
	$c_j - z_j$	10	-57	-9	-24	0	0	0	z = 0
0	Slack1	0.5	-5.5	-2.5	9	1	0	0	0
0	Slack2	0.5	-1.5	-0.5	1	0	1	0	0
0	Slack3	1	0	0	0	0	0	1	1

Which is the same table 1.
Solving the above program by using POM FOR WINDOWS given the following result:
Total iterations : 32767

Note

"This is a fail-safe error message, the error is not your fault. For now though we will resume the program at module level. If you are on the results screen then click on the 'Edit data' button.
If you can I would appreciate it if you would send me an e-mail message describing the problem and maybe attaching the data file to the e-mail message. I am very sorry for the inconvenience, but your feedback will help future users.
hweiss@sbm.temple.edu"

SOLVING THE DEGENERATE PROBLEM BY USING ANTY CYCLE CANONICAL SYSTEM

To solve the degenerate problem, we introduce the following algorithm

1. Get the canonical system to the given LP
2. If thers n constrained with RHS equal to nol then, add a new variabel named anty cycle variabel (equal 1, for example y=1) to all or n-1 constrained wich have RHS equal to nol
3. Add a new constrained to determine the value of the anty cycle variable (for example y=1)
4. Solve the LP by simplex method.

To apply the anty cycle canonical system, we solve the above LP by using POM FOR WINDOWS as follows:

Maximize $z = 10x_1 - 57x_2 - 9x_3 - 24x_4$
subject to

$0.5x_1 - 5.5x_2 - 2.5x_3 + 9x_4 \leq 0$
$0.5x_1 - 1.5x_2 - 0.5x_3 + x_4 \leq 0$
$x_1 \leq 1$
$all\ variables \geq 0$

The canonical system is as follows
Maximize
$z = 10x_1 - 57x_2 - 9x_3 - 24x_4 + 0.slack1 + 0.slack2 + 0.slack3$

subject to

$0.5x_1 - 5.5x_2 - 2.5x_3 + 9x_4 + slack1 = 0$
$0.5x_1 - 1.5x_2 - 0.5x_3 + x_4 + slack2 = 0$
$x_1 + slack3 = 1$
$all\ variables \geq 0$

The anty canonical system is as follows:
Max
$z = 10x_1 - 57x_2 - 9x_3 - 24x_4 + 0.slack1 + 0.slack2 + 0.slack3 + 0.x_5$

Subject to

$0.5x_1 - 5.5x_2 - 2.5x_3 + 9x_4 + slack1 + x_5 = 1$
$0.5x_1 - 1.5x_2 - 0.5x_3 + slack2 = 0$
$x_1 + slack3 = 1$
$x_5 + artfcl4 = 1$
$all\ variables \geq 0$

Total iterations : 7

Solution list

X1	Basic	1
X2	NONBasic	0
X3	Basic	1
X4	NONBasic	0
X5	Basic	1
slack 1	Basic	2
slack 2	NONBasic	0
slack 3	NONBasic	0
artfcl 4	NONBasic	0
Optimal Value (Z)		1

CONCLUTION

Simplex algorithm can'not solve the degenerate linear program problem because there is an infinite cycle in the solution.which make LP return from k-th iteration to the first iteration befor it get the optimal solution, so the LP in this case has no optimal solution. A linear programming problem is a degenerate if there exists at least one of the basic variables equal to zero. Anti cycle Algorithm can solve the infinity cycle problem with add a new variable named anty cycle variable to the constrained which has RHS equal to nol, and allso we add anew constrained to determine the valu of the anty cycle varible which has a value equal one.

REFERENCES

Grafes R. L. & Wolfe (1963);”Recent Advances in Mathematical Programming”. McGraw-Hill Book Company.

Gregoire A. 2007, ‘Numerical Analysis and Optimization’, Oxford University Prees

Hasle G. and Lie A., 2007, ‘Geometric Modelling Numerical Simulation and Optimization’ , Springer-Berlin

Igor G., Stephen G. and Sofer A., 2009, ‘Linear and Non-linear Optimization’ , SIAM-Philadelphia

Nocedal J. and Stephen J., 1999,‘Numerical Optimization’, Springer Series in Operations Research, Springer-verlag New York. Inc

Petrowski A. and Taillard E., 2006, ‘Metaheuristics For Hard Optimization’, Springer-Verlag, Berlin

Price C. J.and Coop I. D., (2003); “Frames and grids in unconstrained and linearly constrained optimization”. A nonsmooth approach. SIAM J. Optim., 14(2), 415-438.

Robert J. V. (2008); “Linear programming foundations and extensions”. 3-ed edition, Springer company.

V. Chavatal. (1983);” Linear programming”,W. H. Freeman and Co., New York