Sparsity Prevention Pivoting Method for Linear Programming

Li, Peiqiang; Li, Qiyuan; Li, Canbing; Zhou, Bin; Cao, Yijia; Wu, Qiuwei; Fang, Baling

Published in:
IEEE Access

Link to article, DOI:
10.1109/ACCESS.2018.2817571

Publication date:
2018

Document Version
Publisher's PDF, also known as Version of record

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 6, 2018, accepted March 10, 2018, date of publication March 21, 2018, date of current version April 25, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2817571

Sparsity Prevention Pivoting Method for Linear Programming

PEIQIANG LI[1], QIYUAN LI[1,2], CANBING LI[1,2], (Senior Member, IEEE), BIN ZHOU[1,2], (Senior Member, IEEE), YIJIA CAO[1,2], (Senior Member, IEEE), QIUWEI WU[3], (Senior Member, IEEE), AND BALING FANG[2,4], (Member, IEEE)

[1]College of Electrical and Information Engineering, Hunan University, Changsha 410082, China
[2]Hunan Key Laboratory of Intelligent Information Analysis and Integrated Optimization for Energy Internet, Hunan University, Changsha 410082, China
[3]Center for Electric Power and Energy, Department of Electrical Engineering, Technical University of Denmark, 2800 Kgs. Lyngby, Denmark
[4]College of Electrical and Information Engineering, Hunan University of Technology, Zhuzhou 412007, China

Corresponding author: Canbing Li (lcb@hnu.edu.cn)

This work was supported by the National Natural Science Foundation of China under Grant 51722701.

ABSTRACT When the simplex algorithm is used to calculate a linear programming (LP) problem, if the matrix is a sparse matrix, it will be possible to lead to many zero-length calculation steps, and even iterative cycle will appear. To deal with the problem, a new pivoting method is proposed in this paper. The principle of this method is to avoid choosing the row which the value of the element in the right side of constraint expression for LP in this row is zero as the row of the pivot element to make the matrix in LP density and ensure that most subsequent steps will improve the value of the objective function. One step following this principle is inserted in the existing LP algorithm to reselect the pivot element. Both the conditions for inserting this step and the maximum number of allowed insertion steps are determined. In the case study, taking several numbers of LP problems as examples, the results indicate that this method can effectively improve the efficiency of LP for the sparse matrix.

INDEX TERMS Linear programming, pivoting rules, simplex algorithm, sparse matrix.

I. INTRODUCTION

Linear programming (LP) is one of the basic contents of mathematical planning. Many realistic problems in business planning, engineering design, industrial production and other fields, can be converted into linear programming problems [1]–[3]. In some case, such as mixed integer linear programming and nonlinear optimization problems, the linear programming need to be calculated several times [4], [5]. Therefore, it is of great significance to improve the computational efficiency of linear programming.

Linear programming can model large and complex problems and can solve these problems in a short period of time by using effective algorithms and modern computers [6]. In 1947, American scholar George B. Dantzig proposed a simplex algorithm for linear programming [7]. This algorithm selects the maximum change variable out of the base in each step, which makes the objective function decline fastest and can calculate the linear programming quickly [8]. This is the most frequently used method to solve linear programming problems, and on this basis, a variety of improvements are proposed, such as two-phase simplex algorithm, big-M simplex algorithm and dual simplex algorithm [9]–[11]. In 1979, the scholar of the Soviet Union Khachiyan L.G. proposed ellipsoid algorithm for linear programming [12]. By using the duality principle, the linear programming problem is transformed into a strict inequality group, and iteratively solves it by cutting the elliptical solution space. The algorithm is theoretically proved to be a polynomial time complexity algorithm, but it is a nonlinear class algorithm, so the amounts of calculation are large in each iteration. The actual calculation speed is not as good as the simplex algorithm. In 1984, American scholar N. Karmarkar proposed an interior-point algorithm for linear programming [13]. The interior-point algorithm seeks the best solution from the interior by constructing the potential function and uses projection transformation to search for the next solution. It has received extensive attention from the international academic circle. The simplex algorithm, ellipsoid algorithm and interior-point algorithm are the three most representative algorithms for solving linear programming problems at present.

The calculation efficiency of the simplex algorithm can be influenced by the degenerate situation. If all the basic

variables of the basic feasible solution x have positive values, it is called non-degenerate situation; conversely, when some basic variables are zero, it is called degenerate situation [14]. If x is a non-degenerate vector, a pole corresponds to a basic feasible solution. Otherwise, a pole may correspond to several basic feasible solutions. In other words, when the simplex algorithm is applied to a linear programming problem, the step length may be zero and the value of the objective function will not change after one iteration. This situation is called the degradation of the simplex algorithm, and the degradation problem will probably cause an iterative cycle [15].

The simplex algorithm starts from a basic feasible solution, each step selects a pivot element according to a method to get another basic feasible solution, and iteratively solve it until the optimal basic feasible solution of the problem is obtained. The choice of the pivot element has a decisive influence on the computational efficiency in the simplex algorithm. Bland's pivoting rule is improved on the basis of the simplex algorithm [16]. Each step chooses the first non-basic variable with the negative value in the reduced cost vector to become the basic variable. Although this method can avoid the iterative cycle due to the degradation, there are still many calculation steps with zero step length and the improvement of the value of the objective function is slow. Most-obtuse angle principle for the simplex algorithm introduces the identification of the angle between the target gradient and the constraint gradient as the basis for determining the priority of the variable to become basic variable [17], [18], but this method only applies to inequality constraints linear programming. Deficient-basis simplex algorithm introduces the concept of deficient basis, so as to form a one-to-one correspondence between different poles and the basic feasible solutions [19]. In this way, the iteration can be carried out between different poles, thus reducing the occurrence of the iterative cycle. These two algorithms can complement each other, therefore, it is very attractive to combine the most-obtuse angle principle and the basis-deficiency algorithms [20]–[22]. In addition, lexicographic order method and perturbation method are also the two kinds of pivoting methods that can avoid the occurrence of iterative cycle [23], [24], but the number of iterations needed in the calculation is more than that of the simplex algorithm.

In this paper, the sparsity prevention pivoting method for sparse matrix linear programming is proposed to reduce the zero-length calculation step caused by the degenerate situation in linear programming. Both the conditions for inserting this step and the maximum number of allowed insertion steps are also determined. In case study, 1000 linear programming problems with 3000 variables, 20 constraints, and 1000 linear programming problems with 2000 variables, 20 constraints are built to test this method. The results show that the method can effectively reduce the iterations of sparse matrix linear programming and improve the computational efficiency.

The rest of this paper is organized as follows. In Section II, the cause of the zero-length calculation step problem is analyzed. Then the sparsity prevention pivoting method is proposed. In Section III, a demonstration using the proposed method is shown. In Section IV, case study simulation is carried out. Finally, the main points are summarized in Section V.

II. THE SPARSITY PREVENTION PIVOTING METHOD

A. THE ZERO-LENGTH CALCULATION STEP PROBLEM IN SIMPLEX ALGORITHM AND ITS REASON

One of the problems in the simplex algorithm is that it is possible to lead to many zero-length calculation steps if the matrix is a sparse matrix. The standard form of the linear programming problem is as follows:

$$\begin{aligned} &\min f = cx \\ &s.t. Ax = b \\ &x \geq 0 \end{aligned} \tag{1}$$

where $A \in R^{m\times n}$, $b \in R^m$, $x, c \in R^n$.

A basis is a submatrix consisting of any linearly independent subset of columns of A. Components of x and c, and columns of A, corresponding to a basis and a non-basis are subscripted with B and N, respectively. The non-basic variable x_N with the maximum absolute value in the reduced cost vector is chosen to become the basic variable in the process of the simplex algorithm. It is considered that the subscript of this variable is k. The reduced cost vector is defined by

$$\sigma_{\mathrm{N}} = c_N^T - c_B^T A_N \tag{2}$$

where c_B is a part of the c vector which corresponds to the basic, c_N is a part of the c vector which corresponds to the non-basis, and A_N is a part of the A matrix which corresponds to the non-basis.

For the simplex algorithm, the a_{ik} is selected as the pivot element, which the subscript i satisfies the condition (3).

$$i = \arg\min \left\{ \frac{b_v}{a_{vk}} | a_{vk} > 0 \, , v = 1, 2, \ldots, m \right\} \tag{3}$$

If this algorithm is used to calculate a problem of sparse matrix in which there are a large number of 0 in the b vector, the value of b_i corresponding to the pivot element may be 0. In this case, the value of the objective function and the b vector will not change after one step using the simplex algorithm. This is a zero-length calculation step.

The reason is that in one step using the simplex algorithm, the column corresponding to the pivot element needs to be transformed into a vector which only the pivot element is 1 and the other elements are 0. This process needs to add the row corresponding to the pivot element to other rows. If the value of b_i in this row is 0, the b vector will not change after adding b_i element to other elements. In this case, value of b_i in the row corresponding to the new base is still 0 after one step using the simplex algorithm, and the value of the elements of other rows in the b vector and the corresponding base will also not change. The formula for the value of the

objective function after each step of the simplex algorithm is as follows:

$$z = c_B b \tag{4}$$

where c_B is a part of the c vector which corresponds to the basic

From this formula, it is seen that the value of the objective function will not change. This step is a zero-length calculation step. According to the pivoting rule in condition (3), it selects the pivot element which satisfies the condition that b_v/a_{vk} is minimum and a_{vk} is greater than 0. And in this case, there must be at least one element of 0 in the b vector. If there is one row that b_v is 0 and the corresponding a_{vk} is greater than 0, then the a_{vk} will be selected as the pivot element. In this way, the behavior that selects the row with the zero value of the element in b vector as the row of the pivot element will appear again, and the value of the objective function will not change until all a_{vk} which corresponding b_v equals 0 is less than or equal to 0.

For a sparse matrix, since there are many zeros in the b vector, the probability of satisfying this condition is low. At the same time, since the A matrix is also sparse matrix, the probability of appearing 0 in the column corresponding to the pivot element after choosing the pivot element is also higher. Then after one step using the simplex algorithm, the value of the element in the b vector of these 0 corresponding rows will not change, so the probability of decreasing the sparseness degree of the b vector will also be reduced. This situation will lead to many zero-length calculation steps in the whole algorithm.

B. BASIC IDEAS OF THE METHOD

To prevent the zero-length calculation step problem in linear programming for sparse matrix and improve computation efficiency, a new pivoting method which called sparsity prevention pivoting method is proposed in this section to modify the simplex algorithm.

For further discussion, sparseness degree of the matrix is defined as follows:

$$SD = \frac{ZN}{AN} \tag{5}$$

where SD is the sparseness degree of the matrix, ZN and AN are the number of 0 elements in the matrix and the number of all elements in the matrix, respectively.

From the analysis in subsection A, the reason for having a zero-length calculation step is that there are many zeros in the vector b. Therefore, the pivot element should be reselected in some of these steps to avoid choosing the row which the value of the element in b vector is zero as the row of the pivot element. The basic idea of this method is that if the b vector and the A matrix are sparse matrices with many zeros, then insert a certain number of steps using the sparsity prevention pivoting method in the calculation. In the step of using the sparsity prevention pivoting method, if the value of the element in the b vector corresponding to the pivot element is 0, which means that the choice of this pivot element will lead to the value of the objective function unchanged, it tries to select other elements which satisfy the conditions that the value of element in the reduced cost vector is negative and the value of element in the b vector corresponding to pivot element is not 0 as the pivot element. When adding or subtracting a non-zero element to other elements, the probability that other elements become non-zero is high. The choice of such elements will make the b vector density and improve the value of the objective function. From the analysis in subsection A, the reduction of the number of 0 in the b vector can help to reduce the appearance probability of the zero-length calculation step, which improves the speed of calculation and is beneficial to the subsequent calculation.

C. ALGORITHM FLOW

The steps of sparsity prevention pivoting method are as follows:

1) Listing the initial simplex tableau based on the normalized form of LP problem
2) Judging whether all numbers in the reduced cost vector are equal to or greater than 0, if yes, the optimal solution and the optimal value are reached. The algorithm is end; otherwise, turn 3).
3) Judging before each step of the algorithm, if the sparseness degree of b vector is greater than 0.4 in this step, and theb_i selected in condition (3) is zero at the same time, and the total number of insertion steps is less than one step, then one step using the sparsity prevention pivoting method will be inserted in this step to make the b vector density. Otherwise, the original selection pivot element method will continue to be used. The pivoting rule of the sparsity prevention pivoting method is as follows:
   (a) Giving priority to select the a_{ik} which satisfies the condition (6) and condition (7) to be the pivot element, x_k becomes the basic variable for calculation.

$$k = \arg\min \left\{ \sigma_{N_j} | \sigma_{N_j} \in \sigma_N \right\} \tag{6}$$

$$i = \arg\min \left\{ \frac{b_v}{a_{vk}} | a_{vk} > 0 \right\} \tag{7}$$

   (b) If the value of b_i which satisfies the condition (7) is zero, it is called that the pivot element a_{ik} is not selected. In this case, the a_{ik} which corresponds to the second smallest number variable in the reduced cost vector and satisfies the condition (7) is selected as the pivot element for the calculation.
   (c) If still cannot select the pivot element a_{ik}, choose a variable in the reduced cost vector which is less than 0 in order from small to large until found the pivot element a_{ik} which meets the condition (7). If indeed can't find the pivot element a_{ik} so that no variable becoming basic variable can improve the value of the objective function, randomly select a non-basic variable with the negative value in the reduced cost vector as the basic variable, and the

a_{ik} which satisfies the condition (3) is selected as the pivot element. The step is not included in the total number of insertion steps.

4) Still use the original selecting pivot element method to complete the calculation of linear programming problems in other steps.

D. DETERMINING THE CONDITION OF INSTERTION

When calculating the linear programming problem with sparse matrices, inserting a certain number of steps using the sparsity prevention pivoting method will help reduce the zero-length calculation step and to improve the computational efficiency. But the step of inserting is more complex than other steps, so it can simply find that this sparsity prevention pivoting method should not be implemented at every step. The condition for using the sparsity prevention pivoting method will be determined in this subsection.

Using mathematical tools, several numbers of linear programming problems with different sparseness degree are constructed and calculated. By calculation and analysis, the relationship between the calculation steps of the simplex algorithm and the sparseness degree of the matrix are shown in Figure 1.

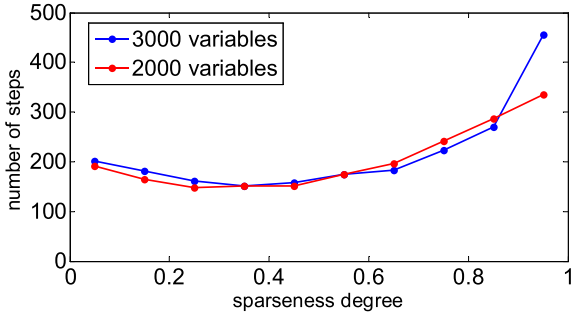


FIGURE 1. Relationship between the average number of calculation steps of the simplex algorithm and the sparseness degree of the matrix.

It is seen that the calculation steps required for the simplex algorithm in low sparseness degree are basically the same, and the calculation steps have a slight decrease when the sparseness degree increases. However, after the sparseness degree of b vector is greater than 0.4, the required calculation steps for computing are gradually increased. It implies that the zero-length calculation step begins to affect the linear programming calculation in this time.

Therefore, when solving a linear programming problem, the sparseness degree of b vector is judged at each step. If the sparseness degree of b vector is greater than 0.4 in this step, the sparsity prevention pivoting method proposed in this paper will be inserted one step to make the b vector density.

E. DETERMINING THE MAXIMUM NUMBER OF INSERTION STEPS

In this subsection, the maximum number of allowed insertion steps using the sparsity prevention pivoting method will be determined. Several linear programming problems with different sparseness degree are created as examples and the calculation efficiency after inserting different maximum number of step using the sparsity prevention pivoting method is analyzed. The result is shown in Figure 2.

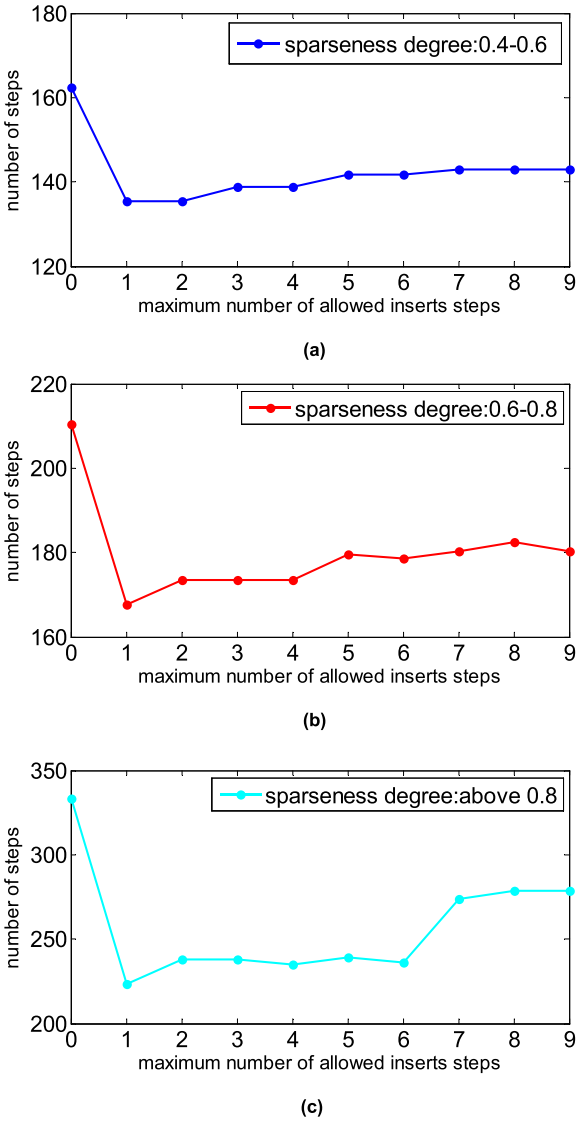


FIGURE 2. Relationship between the average number of calculation steps of the simplex algorithm and the maximum number of allowed inserts steps at different sparseness degree. (a) The case of sparseness degree in 0.4-0.6. (b) The case of sparseness degree in 0.6-0.8. (c) The case of sparseness degree above 0.8.

It is seen that in most cases, the insertion of one step using the sparsity prevention pivoting method is enough. The matrix can become dense by only one step of insertion, and the insertion of more steps will not increase the computational efficiency. At the same time, the step of inserting is more complex than other steps, so the time required for the insertion steps is larger than that for the other steps. Therefore, make a rule that the maximum number of permitted insertion steps using the sparsity prevention pivoting method is one step in each calculation.

III. DEMONSTRATION USING SPARSITY PREVENTION PIVOTING METHOD

Take the linear programming problem (8) as an example, a demonstration is shown in this section.

$$\begin{aligned} \min f &= -14x_1 + 25x_2 - \frac{7}{20}x_3 + 20x_4 \\ \text{subject to } & x_1 - 2x_2 - \frac{1}{10}x_3 + 5x_4 + x_5 = 0 \\ & \frac{7}{10}x_1 - \frac{3}{10}x_2 - \frac{1}{100}x_3 + \frac{19}{50}x_4 + x_6 = 0 \\ & x_1 + x_2 + x_3 + x_4 + x_7 = 5 \\ & x_1 + 2x_2 + 3x_3 + x_4 + x_8 = 10 \\ & x_1, x_2, \ldots, x_8 \geq 0 \end{aligned} \tag{8}$$

The two-phase simplex algorithm and the sparsity prevention pivoting method are used to calculate this problem respectively, the specific process is shown in Table 1 to Table 6.

TABLE 1. Initial simplex tableau.

	x_1	x_2	x_3	x_4	b
x_5	1	-2	-1/10	5	0
x_6	7/10	-3/10	-1/100	19/50	0
x_7	1	1	1	1	5
x_8	1	2	3	1	10
	-14	25	-0.35	20	0

TABLE 2. The simplex tableau after one step calculation using the simplex algorithm.

	x_2	x_3	x_4	x_5	b
x_1	-2	-1/10	5	1	0
x_6	11/10	3/50	-78/25	-7/10	0
x_7	3	11/10	-4	-1	5
x_8	4	31/10	-4	-1	10
	-3	-7/4	90	14	0

As shown in Table 1, a tableau can be used to represent a linear programming problem. In this tableau, the basic variables are x_5, x_6, x_7, x_8. After one step using the simplex algorithm, the $a_{51} = 1$ is selected as the pivot element, which means that x_5 becomes the non-basic variable and x_1 becomes the basic variable. The result is shown in Table 2.

In this case, the value of the objective function and the b vector do not change. It is a zero-length calculation step. By continuing to use the simplex algorithm to calculate, an iterative cycle will be appeared.

Reanalyze the situation of Table 1, since $\sigma_1 = -14$ is the smallest one in the reduced cost vector, it should select x_1 to become the basic variable and $a_{51} = 1$ to become the pivot element. However, due to the corresponding b_5 is 0, it should avoid choosing the element a_{51} in this row as the pivot element. And at the same time, the sparseness degree of vector b is 0.5, which is bigger than 0.4. The sparsity prevention pivoting method is inserted in this step to make the b vector density. The second smallest element in the reduced cost vector is $\sigma_3 = -0.35$, and it corresponding non-basic variable is x_3. According to the rule of selecting the pivot element in section II, $a_{83} = 3$ will be used as the pivot element and the value of the element b_3 in the b vector corresponding to a_{83} is non-zero. Therefore, x_3 becomes the basic variable, x_8 becomes the non-basic variable, and $a_{83} = 3$ is chosen as the pivot element. The result is shown in Table 3.

TABLE 3. The simplex tableau after one step calculation using the sparsity prevention pivoting method.

	x_1	x_2	x_4	x_8	b
x_5	31/30	-29/15	151/30	1/30	1/3
x_6	211/300	-22/75	23/60	1/300	1/30
x_7	2/3	1/3	2/3	-1/3	5/3
x_3	1/3	2/3	1/3	1/3	10/3
	-833/60	757/30	1207/60	7/60	-7/6

TABLE 4. The simplex tableau using the sparsity prevention pivoting method.

	x_2	x_4	x_6	x_8	b
x_5	-317/211	4716/1055	-310/211	6/211	60/211
x_1	-88/211	115/211	300/211	1/211	10/211
x_7	129/211	64/211	-200/211	-71/211	345/211
x_3	170/211	32/211	-100/211	70/211	700/211
	1711/88	5509/199	4165/211	25/137	-385/211

As shown in the Table 3, the b vector becomes dense and the value of the objective function decreases after selecting x_8 to become basic and x_3 to become non-basic. The main reason is that the value of the element in the b vector corresponding to the pivot element is non-zero. When adding or subtracting a non-zero element to other elements, the probability that other elements become non-zero will be high. The objective function will be improved at each step in subsequent calculations. Of course, another important reason why b vector becomes all non-zero is that the columns represented by x_3 in Table 1 are all non-zero. If there are more zero elements in this column, the b vector may still be sparse after one step of the operation on the simplex tableau.

Still use the original simplex algorithm to complete the calculation of linear programming problems in other steps. The $a_{61} = 211/300$ will be selected as the pivot element and Table 4 will be obtained.

All numbers in the reduced cost vector are greater than 0 and the optimal solution is obtained in this step. The optimal solution of the problem is $x = (x_1, x_2, x_3, x_4) = (10/211, 0, 700/211, 0)$, the optimal value is $-385/211$. The number of steps required for a calculation using the sparsity prevention pivoting method is only 2 steps.

TABLE 5. Iterations of the first 50 LP problems.

Problem number	CLS	SPPM	Problem number	CLS	SPPM
LP problem 1	138	125	LP problem 26	86	85
LP problem 2	453	433	LP problem 27	5304	150
LP problem 3	8368	460	LP problem 28	156	110
LP problem 4	loop	244	LP problem 29	357	344
LP problem 5	225	108	LP problem 30	802	346
LP problem 6	418	408	LP problem 31	167	154
LP problem 7	199	158	LP problem 32	742	738
LP problem 8	1286	1212	LP problem 33	2842	424
LP problem 9	256	259	LP problem 34	138	131
LP problem 10	456	430	LP problem 35	75	73
LP problem 11	237	237	LP problem 36	2331	199
LP problem 12	119	102	LP problem 37	118	105
LP problem 13	62	63	LP problem 38	799	130
LP problem 14	2031	2031	LP problem 39	4943	4927
LP problem 15	206	163	LP problem 40	538	210
LP problem 16	201	172	LP problem 41	222	222
LP problem 17	147	140	LP problem 42	402	114
LP problem 18	126	115	LP problem 43	207	200
LP problem 19	421	393	LP problem 44	403	380
LP problem 20	84	82	LP problem 45	4709	137
LP problem 21	144	135	LP problem 46	125	125
LP problem 22	357	358	LP problem 47	217	223
LP problem 23	275	275	LP problem 48	212	198
LP problem 24	loop	311	LP problem 49	468	59
LP problem 25	408	308	LP problem 50	64	62

TABLE 6. The average iterations of 1000 linear programming problems with 3000 variables and 20 constraints

SD	CLS	MOA	SPPM	$\frac{\text{CLS}}{\text{SPPM}}$	$\frac{\text{MOA}}{\text{SPPM}}$
0-0.1	189.316	158.172	187.889	1.0076	0.8418
0.1-0.2	169.103	142.727	160.871	1.0512	0.8872
0.2-0.3	142.104	121.665	133.052	1.0680	0.9144
0.3-0.4	140.823	120.084	125.434	1.1227	0.9574
0.4-0.5	161.382	138.347	132.073	1.2219	1.0475
0.5-0.6	177.981	154.070	142.472	1.2492	1.0814
0.6-0.7	194.262	169.085	153.369	1.2666	1.1025
0.7-0.8	244.588	214.344	181.536	1.3473	1.1807
0.8-0.9	294.105	260.639	190.316	1.5454	1.3695
Total	1713.666	1479.134	1407.012	1.2179	1.0513

IV. CASE STUDY

In this section, a numerical experiment is carried out to gain an insight into the behavior of this new method. By using mathematical tools, the proposed sparsity prevention pivoting method in this paper and the two-phase simplex algorithm are implemented. 1000 linear programming problems with 3000 variables and 20 constraints are constructed. Using CLS represents the two-phase simplex algorithm and SPPM represents the sparsity prevention pivoting method, the iterations of the first 50 linear programming problems is shown in Table 5:

It is seen that the iterations of the sparsity prevention pivoting method proposed in this paper is less than that of the two-phase simplex algorithm. The specific improvement effect is related to the properties of the matrix of linear programming, which is specifically related to the sparseness degree of the matrix A and the vector b.

Using the matrix sparseness degree standard mentioned in section II equations (5), all 1000 problems are classified according to the sparseness degree of the matrix A and the vector b. The average iterations of the sparsity prevention pivoting method and that of the two-phase simplex algorithm are shown in Table 6. To compare with other pivoting methods in terms of computational efficiency, the results of the calculation using the most-obtuse angle method (MOA) are also shown in Table 6.

The results show that the sparsity prevention pivoting method proposed in this paper can reduce the iterations required for linear programming. When the sparseness degree of the matrix A and that of the vector b are low, the calculation efficiency is higher than that of two-phase simplex algorithm, but it is lower than that of most-obtuse angle method, the efficiency of calculation is not improved obviously. When the sparseness degree of the matrices is greater than 0.4, calculation efficiency of the sparsity prevention pivoting method begins to exceed that of most-obtuse angle method.

TABLE 7. The average iterations of 1000 linear programming problems with 2000 variables and 20 constraints

SD	CLS	MOA	SPPM	CLS / SPPM	MOA / SPPM
0-0.1	200.378	168.060	199.277	1.0055	0.8433
0.1-0.2	180.233	153.051	175.758	1.0255	0.8708
0.2-0.3	157.683	133.845	144.619	1.0903	0.9255
0.3-0.4	153.826	132.552	131.835	1.1668	1.0054
0.4-0.5	161.235	139.393	135.824	1.1871	1.0263
0.5-0.6	164.940	143.965	135.222	1.2198	1.0647
0.6-0.7	192.144	168.696	152.154	1.2628	1.1087
0.7-0.8	234.771	205.507	168.829	1.3906	1.2173
0.8-0.9	295.906	261.194	187.375	1.5792	1.3940
Total	1741.117	1506.261	1430.893	1.2168	1.0527

And the more sparsity of the matrix is, the more efficiency of the sparsity prevention pivoting method is. Compared with the two-phase simplex algorithm and the most-obtuse angle method respectively, the highest calculation efficiency can be increased by 54.5% and 36.9%.

Similarly, 1000 linear programming problems with 2000 variables and 20 constraints are constructed. The two-phase simplex algorithm, the most-obtuse angle method and the sparsity prevention pivoting method are used to calculate the results respectively. The result is shown in Table 7.

It is seen that the result is basically the same as those descriptions mentioned above.

V. CONCLUSIONS

In this paper, a sparsity prevention pivoting method is proposed to reduce the zero-length calculation step in linear programming. One step is inserted to reselect the pivot element in the existing linear programming algorithm in this method, when the sparseness degree of b vector is greater than 0.4. The maximum number of allowed insertion steps is determined as one step in each calculation. It can avoid choosing the row which the value of the element in the b vector in this row is zero as the row of the pivot element, which makes b vector density after the inserted step and reduces the appearance probability of the zero-length calculation step.

Through the case study, it is shown that the iterations required for linear programming can be reduced effectively by inserting one step to reselect the pivot element using the proposed sparsity prevention pivoting method. And the higher the sparseness degree of the matrix is, the better the effect of the algorithm is. This method can be widely used in the situation of sparse matrix linear programming, such as the node admittance matrix of the power grid.

REFERENCES

[1] M. H. F. Wen and V. O. K. Li, "Optimal phasor data compression unit installation for wide-area measurement systems—An integer linear programming approach," *IEEE Trans. Smart Grid*, vol. 7, no. 6, pp. 2644–2653, Nov. 2016.

[2] S. Mashayekh, M. Stadler, G. Cardoso, and M. Heleno, "A mixed integer linear programming approach for optimal DER portfolio, sizing, and placement in multi-energy microgrids," *Appl. Energy*, vol. 187, pp. 154–168, Feb. 2017.

[3] J. Martinovic and G. Scheithauer, "Integer linear programming models for the skiving stock problem," *Eur. J. Oper. Res.*, vol. 251, no. 2, pp. 356–368, Jun. 2016.

[4] J. Sun, "On methods for solving nonlinear semidefinite optimization problems," *Numer. Algebra Control Optim.*, vol. 1, no. 1, pp. 1–14, Mar. 2011.

[5] Y. Abdelsadek, F. Herrmann, I. Kacem, and B. Otjacques, "Branch-and-bound algorithm for the maximum triangle packing problem," *Comput. Ind. Eng.*, vol. 81, pp. 147–157, Mar. 2015.

[6] M. S. Bazaraa, J. J. Jarvis, and H. D. Sherali, *Linear Programming and Network Flows*, 3th ed. New York, NY, USA: Wiley, 2009, pp. 1–6.

[7] G. B. Dantzig, "Maximization of a linear function of variables subject to linear inequalities," in *Activity Analysis of Production and Allocation*. New York, NY, USA: Wiley, 1951, pp. 339–347.

[8] H. Nabli, "An overview on the simplex algorithm," *Appl. Math. Comput.*, vol. 210, no. 2, pp. 479–489, Apr. 2009.

[9] S. A. Edalatpanah and S. Shahabi, "A new two-phase method for the fuzzy primal simplex algorithm," *Int. Rev. Pure Appl. Math.*, vol. 8, no. 2, pp. 157–164, Jan. 2012.

[10] M. Soleimani-Damaneh, "Modified big-M method to recognize the infeasibility of linear programming models," *Knowl.-Based Syst.*, vol. 21, no. 5, pp. 377–382, Jul. 2008.

[11] K. Paparrizos, N. Samaras, and G. Stephanides, "A new efficient primal dual simplex algorithm," *Comput. Oper. Res.*, vol. 30, no. 9, pp. 1383–1399, Jul. 2003.

[12] L. G. Kachiyan, "A polynomial time algorithm for linear programming," *Soviet Math Dokl*, vol. 4, no. 4, pp. 373–395, 1979.

[13] N. Karmarkar, "A new polynomial-time algorithm for linear programming," *Combinatorica*, vol. 4, no. 4, pp. 373–395, Dec. 1984.

[14] P. Zörnig, "Systematic construction of examples for cycling in the simplex method," *Comput. Oper. Res.*, vol. 33, no. 8, pp. 2247–2262, Aug. 2006.

[15] E. M. L. Beale, "Cycling in the dual simplex algorithm," *Naval Res. Logistics Quart.*, vol. 2, no. 4, pp. 269–275, Dec. 1955.

[16] R. G. Bland, "New finite pivoting rules for the simplex method," *Math. Oper. Res.*, vol. 2, no. 2, pp. 103–107, 1977.

[17] W. Li, P. Guerrero-García, and A. Santos-Palomo, "A basis-deficiency-allowing primal phase-I algorithm using the most-obtuse-angle column rule," *Comput. Math. Appl.*, vol. 51, nos. 6–7, pp. 903–914, Mar. 2006.

[18] Y. Ma, L. Zhang, and P. Pan, "Criss-cross algorithm based on the most-obtuse-angle rule and deficient basis," *Appl. Math. Comput.*, vol. 268, no. 9, pp. 439–449, Oct. 2015.

[19] P.-Q. Pan, "A primal deficient-basis simplex algorithm for linear programming," *Appl. Math. Comput.*, vol. 196, no. 2, pp. 898–912, Mar. 2008.

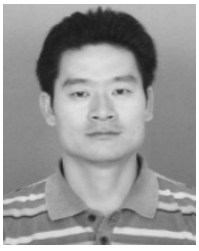
[20] P.-Q. Pan, "A dual projective simplex method for linear programming," *Comput. Math. Appl.*, vol. 35, no. 6, pp. 119–135, May 1998.

[21] Y. Yang, J. Di, and Y. Gao, "Research on goal programming algorithm based on lingo," *J. Converg. Inf. Technol.*, vol. 8, no. 7, pp. 1009–1016, Apr. 2013.

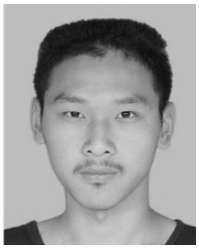
[22] J. C. Zavala-Díaz, M. A. Cruz-Chávez, and M. H. Cruz-Rosales, "Mathematical Multi-objective Model for the selection of a portfolio of investment in the Mexican Stock Market," *Adv. Inf. Sci. Service Sci.*, vol. 2, no. 2, pp. 67–76, Jun. 2010.

[23] G. B. Dantzig, A. Orden, and P. Wolfe, "The generalized simplex method for minimizing a linear form under linear inequality restraints," *Pacific J. Math.*, vol. 5, no. 2, pp. 183–195, 1955.

[24] H. S. Najafi and S. A. Edalatpanah, "Homotopy perturbation method for linear programming problems," *Appl. Math. Model.*, vol. 38, nos. 5–6, pp. 1607–1611, Mar. 2014.



PEIQIANG LI was born in Shanxi, China, in 1975. He received the B.S. degree in electrical engineering from Changsha Electric Power University, Changsha, China, in 1997, and the master's and Dr.Eng. degrees in electrical engineering from Hunan University in 2004 and 2009, respectively. He is currently an Associate Professor with Hunan University. His research interests include load modeling and electric power automation.



QIYUAN LI was born in Liaoning, China, in 1994. He received the B.E. degree in electrical engineering from Hunan University, Changsha, China, in 2016, where he is currently pursuing the M.S. degree with the College of Electrical and Information Engineering. His major research interests include data mining and optimization.

CANBING LI (M'06–SM'13) was born in Hunan, China, in 1979. He received the B.S. and Ph.D. degrees in electrical engineering from Tsinghua University, Beijing, China, in 2001 and 2006, respectively. He is currently a Professor with the College of Electrical and Information Engineering, Hunan University, Changsha, China. His research interests include smart grid, energy efficiency, and energy policy.

BIN ZHOU (S'11–M'13–SM'17) was born in Hunan, China, in 1984. He received the B.S. degree in electrical engineering from Zhengzhou University, Zhengzhou, China, in 2006, the M.S. degree in electrical engineering from the South China University of Technology, Guangzhou, China, in 2009, and the Ph.D. degree from The Hong Kong Polytechnic University, Hong Kong, in 2013.He was a Research Associate and subsequently a Post-Doctoral Fellow with the Department of Electrical Engineering, The Hong Kong Polytechnic University. He is currently an Associate Professor with the College of Electrical and Information Engineering, Hunan University, Changsha, China. His research interests include smart grid operation and planning, renewable energy generation, and energy efficiency.

YIJIA CAO (M'98–SM'13) was born in Hunan, China, in 1969. He received the B.S. degree in electrical engineering from Xi'an Jiaotong University, Xi'an, China, in 1988, and the Ph.D. degree from the Huazhong University of Science and Technology, Wuhan, China, in 1994. He is currently a Chair Professor and Vice President with the College of Electrical and Information Engineering, Hunan University, Changsha, China. His research interests include power system cascading failure, smart grid information technology, smart grid operation, and optimization.

QIUWEI WU (M'08–SM'15) received the B.Eng. and M.Eng. degrees in power system and its automation from the Nanjing University of Science and Technology, Nanjing, China, in 2000 and 2003, respectively, and the Ph.D. degree in power system engineering from Nanyang Technological University, Singapore, in 2009.

He was a Senior Research and Development Engineer with VESTAS Technology R&D Singapore Pte Ltd from 2008 to 2009. He has been with the Department of Electrical Engineering, Technical University of Denmark, since 2009 (Post-Doctoral Fellow from 2009 to 2010, an Assistant Professor from 2010 to 2013, and an Associate Professor since 2013). He was a Visiting Scholar with the Department of Industrial Engineering and Operations Research, University of California at Berkeley, in 2012 funded by the Danish Agency for Science, Technology and Innovation, Denmark. He was a Visiting Professor named by Y. Xue, an Academician of the Chinese Academy of Engineering, at Shandong University, China, from 2015 to 2017. He is currently a Visiting Scholar with the School of Engineering and Applied Science, Harvard University.

His research area is power system operation and control with high renewables, including wind power modeling and control, active distribution networks, and integrated energy systems. He is an Editor of the IEEE TRANSACTIONS ON SMART GRID and the IEEE POWER ENGINEERING LETTERS. He is also an Associate Editor of the *International Journal of Electrical Power and Energy Systems*, the *Journal of Modern Power Systems and Clean Energy*, and the *IET Renewable Power Generation*.

BALING FANG (S'13–M'17) received the M.B.A. degree from Jilin University, Jilin, China, in 2009, and the Ph.D. degree in electrical engineering from Hunan University, Changsha, China, in 2017. His research interests include smart grid and cloud computing.