Report of dual simplex method implementation in MATLAB

吴嘉骜 21307130203

2023 年 5 月 31 日

1 Introduction

Dual simplex method is a variant of the simplex method for solving linear programming problems. It is used to solve the dual problem of the original problem, which is equivalent to solving the original problem.
The dual simplex method is similar to the primal simplex method, but it is more efficient in some cases. For example, when the initial basic solution is not primal feasible but dual feasible, the dual simplex method can be used to find a optimal solution by pivoting on the dual variables.

Purpose of the experiment

The purpose of this experiment is to understand the principles behind the dual simplex method and to demonstrate its implementation in MATLAB.

Environment of the experiment

MATLAB R2022a on Windows 11.

Brief overview of the report

We will begin by describing the methodology used to implement the dual simplex method in MATLAB and present the results obtained from the implementation.

Then, we will discuss the implications of the findings and the difficulites confronted with.
Finally, we will draw conclusions and provide recommendations for future improvement.

2 Experiment design and methodology

Dual Simplex

[optsol, optval, runhist, status, iteration number] = dualsimplex (A, b, c, bas index, tol, maxiter, step present) finds the optimal solution and cost of the primal problem by dual simplex method.

Inputs:
A: $m \times n, m < n$, input matrix, with full row rank.
b: $m \times 1$ the right hand of equality constraints.
bas index: a row vector that consists of basis indices satisfying dual feasibility (i.e. reduced costs ≥ 0).
tol: the tolerance to set zero, such as 10^{-10}.
maxiter: maximum iteration numbers.
step present: 0 or 1. $= 1$ if every tableau in each step is presented; $= 0$ otherwise.

Outputs:
optsol: optimal solution of the primal problem.
optval: optimal value.
runhist: cost in each iteration.
status (of the dual problem): optimal or unbounded or infeasible or maxiteration.
iteration number.

Intermediate process:
1. Check if B is with all reduced costs nonnegative.
2. Check if the current basis is optimal. If the components in the zeroth

column of the tableau are all nonnegative, return the optimal solution and cost.
3. If the current basis is not optimal, find the pivot row as well as the exiting variable.
4. Check if the problem is unbounded. If the components in the pivot column are all nonnegative, return the status and terminate.
5. If the problem is not unbounded, find the pivot element by ratio test.
6. Update the tableau.
7. Repeat step 2 to 6.

Main function

[optsol, optval, optbas, runhist, info] = main(A, b, c, bas index, options) is the full implementation of the dual simplex method.
Inputs:
A: $m \times n, m < n$, input matrix, with full row rank.
b: $m \times 1$ the right hand of equality constraints.
c: $n \times 1$ the cost coefficients of the problem.
bas index: a row vector that consists of basis indices satisfying dual feasibility (i.e. reduced costs ≥ 0).
options: a struct that contains: [*tol*, *maxiter*, step present]

Outputs:
optsol: optimal solution of the primal problem (if the problem is infeasible or unbounded, then it will be returned as $NULL$.)
optval: optimal value (if the dual problem is infeasible, it will be returned as $NULL$; if unbounded, it will be returned as ∞.)
optbas: the basis corresponding to the optimal value.
runhist: cost in each iteration.
info: a struct that contains: [status, iteration number].

Intermediate process:
1. Check the input conditions: whether $rank(A) = rank(B) = m$ holds.

2. Carry out dualsimplex.
3. Return the status, optimal value and solution.
4. Print the results in the command window.

3 Numerical results

3.1 A simple case

Consider the following linear programming problem:

$$\begin{aligned}
&\mathit{minimize} \quad 2x_1 + 3x_2 + 3x_3 + x_4 - 2x_5 \\
&\mathit{subject\,to} \quad x_1 + 3x_2 + 4x_4 + x_5 = 2 \\
&\qquad x_1 + 2x_2 - 3x_4 + x_5 = 2 \\
&\qquad - x_1 - 4x_2 + 3x_3 = 1 \\
&\qquad x_1, \dots, x_5 \geq 0.
\end{aligned}$$

The *tol* is set to be 10^{-10}, and the maximum iteration is set to be 1000.
The initial basis is $B = \begin{bmatrix} 1 & 3 & 1 \\ 1 & 2 & 1 \\ -1 & -4 & 0 \end{bmatrix}$.
The output results are:

```
——————Problem 1: Simple Case——————
————iterate 1 tableu————
T0 =
5.9286 2.0714 0 8.7857 0 0
T =
0.0357 -0.0357 0 0.1071 1.0000 0
-0.2500 0.2500 1.0000 -0.7500 0 0
2.6071 0.3929 0 1.8214 0 1.0000
————iterate 2 tableu————
T0 =
3.0000 5.0000 11.7143 0 0 0
T =
0 0 0.1429 0 1.0000 0
0.3333 -0.3333 -1.3333 1.0000 0 0
2.0000 1.0000 2.4286 0 0 1.0000
The primal problem status is optimal.
The optimal cost is -3.000000.
The optimal solution is:
optsol =
0
0
0.3333
0
2.0000
The corresonding basis index is:
optbas =
1 2 5
```

3.2 A primal infeasible case

Consider the following linear programming problem:

$$\begin{aligned} &\textit{minimize} \quad 20x_2 + 13x_4 \\ &\textit{subject to} \quad x_1 + 2x_2 + 2x_4 = 100 \\ &\qquad x_3 + 3x_4 = -2 \\ &\qquad 8x_2 + 6x_4 + x_5 = 5 \\ &\qquad x_1, \ldots, x_5 \geq 0. \end{aligned}$$

The *tol* is set to be 10^{-10}, and the maximum iteration is set to be 1000.

The initial basis is $B = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$.

The output results are:

```
——————Problem 2: Infeasible primal Case——————
The primal problem is infeasible.
```

3.3 A primal unbounded case

Consider the following linear programming problem:

$$\begin{aligned} &\textit{minimize} \quad -2x_2 - 3x_3 + 5x_4 + x_5 + 3x_6 \\ &\textit{subject to} \quad x_1 - x_2 + x_3 + x_4 - x_6 = 5 \\ &\qquad -2x_1 + x_2 + x_5 = 3 \\ &\qquad x_2 - 2x_3 + 2x_4 + x_6 = 7 \\ &\qquad x_1, \ldots, x_6 \geq 0. \end{aligned}$$

The *tol* is set to be 10^{-10}, and the maximum iteration is set to be 1000.

The initial basis is $B = \begin{bmatrix} 1 & 0 & -1 \\ -2 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$.

The output results are:

——————Problem 3: Unbounded primal Case——————

The initial basis is not dual feasible!

3.4 random case

Use MATLAB to randomly generate several linear programming problems with m constrains and n variables. In order to get the initial basis easily, we choose to solve $Ax \leq b, x \geq 0, c \geq 0$. By introducing slack variables, we can get the standard form of the linear programming problem.

The correctness can be verified by MATLAB built-in solver *linprog*. Codes can be found in the appendix.

For instance, we use seed 7130203 to generate the random numbers, and set $m = 100, n = 500$.

The *tol* is set to be 10^{-8}, and the maximum iteration is set to be 10000.

The output results are:

——————Problem 4: Random Case——————

The primal problem status is optimal.

The optimal cost is 1.489685.

The optimal solution is: (omitted for simplicity)

The corresonding basis index is:

optbas = (omitted for simplicity)

4 Discussion

4.1 Analysis of the results

The results are consistent with the theoretical analysis, and are also consistent with the results of the MATLAB solver *linprog*.
For primal feasible cases with finite optimal value, such as Problem 1 and 4, the program can get the optimal solution and the optimal cost.
For primal unbounded cases, such as Problem 3, we know that the dual problem must be infeasible. So in our program, we can never find a dual feasible basis, which means that the program cannot be carried out.
For primal infeasible cases, such as Problem 2, the dual problem can be infeasible or unbounded. But we cannot distinguish them by the program. The key difficulity is that we cannot obtain a dual feasible basis at first by some program. Hence, the dual infeasible case cannot be detected automatically. Also, it is not always easy to find a dual feasible basis by hand.
In all, the program is can work smoothly for primal feasible cases with finite optimal value and a known dual feasible basis. But for other cases, the program may not be satisfactory.

4.2 Conclusion

This experiment preliminarily implements the dual simplex method for linear programming. The results are consistent with the theoretical analysis and the results of the MATLAB solver. This program can be used to solve linear programming problems with a known dual feasible basis. However, it may mot be helpful for other cases.

4.3 Future work

A more general dual simplex method can be implemented, probably a dual two phase method that can find a dual feasible basis automatically. More requirements such as solving the general form of linear programming problems, handling the rank deficient constraints, and so on, can be satisfied in the program. The anticycling strategy should also be taken into consideration. The code style and the efficiency of the program can be improved so as to accelerate the solving process.

5 References

Dimitris Bertsimas and John Tsitsiklis. 1997. Introduction to Linear Optimization (1st. ed.). Athena Scientific.