Implementation of blending problem algorithm into information system

Pavel Kolman[1]

Abstract. Optimal feed mix is very important part of cattle nutrition problem. For filling in of missing ingredients are used food addictions. They are produced according to needs of concrete breeder. All the ingredients in animal feed addictions must occur in exactly given amount. This requirement is determinative for breeder, for feed addictions producer are most important minimal costs. To find optimal ratio of nutrients to get all ingredients in required amount (or in given bound), it is used blending problem. It belongs to linear programming and is easily solvable with simplex method.

This paper describes problematic of blending problem implementation, when finding optimal nutrition ingredients ratio in producing company. Requirement of company is to have blending problem (include simplex method algorithm, solving this problem) implemented in their information system, so that they were able to optimize nutrient ratio of each food addiction they produce.

Keywords: Nutrition problem, Blending problem, Simplex method, cycling in the Simplex method, Delphi, Programming.

JEL Classification: C44, C61
AMS Classification: 46N10, 90C05, 90C08

1 Introduction

Costs minimization is a very important factor for company which is blending feed addictions. That is the main reason, why there is used optimization in a decision process, which nutrients and in what amount will be used for food addictions mixing process. Character of solved problem allows creating mathematical model solvable with linear programming methods. For optimization it is used the simplex method. In this paper there will be described problematic of simplex method algorithm implementation into information system. This information system is being developed for producing company. The concrete name of developing company, nor producing company will not be mentioned, from competitive reasons.

2 Material and methods

When finding optimal composition of food addictions, in the first step it will be necessary to make a mathematical model of described problem. The variables in mathematical model will be searched amounts of nutrients in given units (they will differ according to used nutrient). Constraints in mathematical model will describe amounts of ingredients (minerals, proteins etc.) in minimal, maximal amount or given bound. With regard to mathematical model character of blending problem there will be always used simplex method.

The simplex method will be applied according to generally known rules, with exception to pivot selection. Because the simplex algorithm is very well known, it will be described very briefly. In the first step, mathematical model will be converted into standard form, i.e. all the inequalities will be changed to equations by adding additional variables (slack or surplus). Then follows making of canonical form of LP problem, where there will added artificial variables so that there was identity submatrix. This canonical form is the initial basic solution. The next steps consist in individual iterations, considering from optimality test, pivot selection and basis change. While for optimality test and basis change will be used well known rule as described in literature, pivot selection rule will differ.

Reason for different pivot change rule is elimination of potential cycling in solved mathematical model. It can occur only in degenerated problems, but this is a quite often phenomenon in blending problem models. As a cycling we understand situation, when after several iterations in simplex algorithm we get to the same basis, we have already been before. As a necessary condition for cycling occurrence is zero objective function change, because its value within individual iterations cannot worse. If there would be used generally known pivot selection rule according to Jablonský [4] or Stevenson [5], some of problems (e.g. Todd [2]) would get into a cycle. For cycle elimination there will be used Blend's pivot selection rule.

Blend's pivot selection rule chooses as a variable entering the basis that with its minimal index. The algorithm describes e.g. Blend [1].

[1] Mendel University in Brno, Faculty of Business and Economics, Department of Statistics and Operation Analysis, Zemědělská 1, 60300 Brno, xkolman@mendelu.cz.

When the application has been developed, for pivot selection there was used the combination of both approaches. Because cycling can occur only in situation, when objective function change is 0, the newly defined pivot selection algorithm was following:

- Pivot was selected with according to the original simplex algorithm. When its selection has improved objective function, was used for basis change.
- When the objective function improvement was zero, pivot was chosen according to Blend's algorithm.

It is obvious, that in this modification cannot occur cycling, because in all potential cycle occurrences it is used Blend's anti-cycling rule which excludes this eventuality. Textbook example of LP problem where can occur cycling is visible on Figure 2**,** describing input data format. It was gained from Todd's [2] demonstration of cycling simplex method. For terminating of algorithm calculation is sufficient optimality criterion fulfillment. If obtained solution is feasible or nor will be decided considering to blending problem specifics directly in information system regarding to structural variables values.

3 Results

Before description of proper program and its functions, it should be mentioned reason why the program was developed in Delphi as a console application. The information system itself is being developed in Microsoft Dynamics NAV, also known as Navision. Navision is not suitable for large problems calculations, but this is typical property of blending problem models. From this reason it was agreed, that communication interface will be solved in Navision and own calculations will be realized by console application. The application will be available to information system in exactly specified directory and information system will be allowed to run it always when necessary. Advantage of this approach is a fact that whole console application takes after compilation only 100 Kbytes of memory! The reason of such small size is mainly caused by absence of graphical interface in application.

3.1 Program structure

The own program, that will be executed from information system, will be console application programmed in Delphi. For own calculations will be run with 2 parameters. The parameters itself will be input file with mathematical model in required format (this model will be information system output and console application input) and output file will be optimal solution of LP problem (this file will be the other way around console application output and information system input). Because the program is made as a console application, it can be run parametrically from command line, or directly from some programs. This is very important requirement, mainly from reason, that information system will run the application automatically, without user notification. The execution of program with two parameters will be following:

```
simplexka.exe parameter1 parameter2,
```

where `simplexka.exe` is program name, parameter1 contains name of file with mathematical model (alternatively with path) and parameter2 is name of file, where the optimal solution will be saved. After execution with parameters the mathematical model is loaded, solved and optimal solution saved to file from parameter2. Here should be mentioned 2 facts. First of all, if file from parameter2 exists, is automatically (without warning) overwritten by newly created file. Second of all, if program has no access rights to directory with optimal solution file, the file with optimal solution will not be saved. For verification, if solved problem has feasible solution or no, is intended program exitcode. In case that optimal solution is found, the exitcode value is set as 0, otherwise 1. Verification, which requirements has not been fulfilled and if those breakings are important or no (e.g. if non fulfillment of constraint is in thousandths of percent, constraint can be although considered as fulfilled) is depending on information system developers.

The main application source code is displayed in Figure 1 and is relatively short. The main reason is that all functions needed for successful problem solving are included in `simplexka.pas` unit. This unit (library) is a necessary part of program and was programmed to be usable in other applications where simplex method is needed. Unit simplexka.pas detailed description transcends range of this paper, therefore will not be furthermore described. So what happens, when the program is executed? First of all, the parameters are load. The input file is saved to input variable, output file to output variable. Then the mathematical model saved in input file is load to LP variable. It is structured data type containing whole linear programming problem. Afterwards, the

`vyresit_ulohu_LP` function is run. The function has 2 parameters: the first parameter is mathematical model contained in LP variable, the second is LP problem solution vector. It is a dynamic data field of real48 data type. After its termination the function returns Boolean value true in case, that optimal solution of mathematical model in LP variable was found. If obtained solution is not feasible, return value of function is false. According to function return value, application exitcode is assigned. The last function of main program is `zapis_reseni`

function with 2 parameters. The first parameter is dynamic data field containing optimal values of structural variables the second one is file, where this optimal solution will be saved. The main application source code is visible on Figure 1.

```
program krmiva;
{$APPTYPE CONSOLE}
uses
  SysUtils,
  simplexka in 'simplexka.pas';
var
  input, output: string; { input and output file }
  lp: TLPproblem; { LP model }
  reseni: Treseni; { LP model solution }
begin
  try
    { load input and output file from parameters }
    input := ParamStr(1);
    output := ParamStr(2);
    { loads model to LP variable }
    lp := nacti_ulohu(input);
    { solves LP problem and results if solution is feasible or no }
    if vyresit_ulohu_LP(lp, reseni) then
      exitcode := 0
    else
      exitcode := 1;
    { writes solution to file }
    zapis_reseni(output, reseni);
  except
    on E: Exception do
      Writeln(E.ClassName, ': ', E.Message);
  end;
end.
```

Figure 1 The main application source code.

3.2 Input data

Because the information system is being developed in Navision and application solving the blending problem is executable *.exe file, it was necessary to agree on communication between information system and program. With information system developers it was agreed, that mathematical model will be put together in information system in a specific format, and saved to exactly specified place. Therefrom it will be load from application, solved and saved to previously specified place. At the same time, the problem solution will be load from information system and processed. Mathematical model construction will be implemented directly in information system. Its correct construction will handle person responsible for food addictions blending. The main application task is correct solving of LP problem and return of result in specified format to information system. Because the program works in Czech computer environment, the decimal separator is comma. Example of mathematical model in specified format (in detail will be described below) is on Figure 2.

```
[Objective function]¶
Z MAX¶
10 → -57 → -9 → -24¶
[Binding constraints]¶
0,5 → -5,5 → -2,5 → 9 → <= → 0¶
0,5 → -1,5 → -0,5 → 1 → <= → 0¶
1 → → → → <= → 1¶
```

Figure 2 Example of cycling LP problem in input file format.

The first row contains comment, that objective function follows and has only informative meaning for user. It can contain any text and during import is only loaded, but not processed. From second row, the LP type is loaded. Although it is assumed that there will be solved only minimization problems (resulting from blending problem properties), regarding to program universality there remained this option. When the solved problem is maximization type, text string contains "Z_MAX" value, for minimization problems "Z_MIN". On the third row there

are objective function coefficients separated by tab sign (#9 in ASCII code) and ended by end of line sign (CR-LF in Windows operation system, in ASCII code represented by #13#10). On the fourth row, there begins loading of constraints. This row has only informative meaning, the same as first row. Since fifth row, the constraints are founded, each constraint on one row. The structural coefficients separator is in the same way as in objective function the tab sign. The constraint is in form structural coefficients, relation and right-hand side, all separated by tabulators. Example of mathematical model in described format is visible on Figure 1. The main advantage of previously described format is its universality. It is very easy to take any mathematical model and in any text editor or table processor, prepare it to given format.

In this situation here should be mentioned case, where a structural or objective function coefficient is zero valued. Regarding easy transport from or to MS excel (mainly through clipboard), it is possible instead of zero values write empty strings (separated with tabulators). Then, zero valued coefficients are recognized according to two consecutive tabulators, e.g. see last row of Figure 2 example, where there are 3 zero valued structural coefficients.

The question of correct mathematical model loading is very important for practical use of program. It is not acceptable to load wrong any constraint or, not to load it. Therefore there should be mentioned, how does the application identifies real number of variables and constraints in a model. When the mathematical model is being loaded, the first row is ignored. Number of structural variables is specified from 4^{th} non-empty row. It is the row, where first constraint occurs. Number of structural variables is specified as number of tab (#9) signs before first occurrence of "<" or "=" or ">" sign. Number of constraints corresponds to number of non-empty rows, decreased by four. From these four rows, first three are related to objective function and fourth row informs user that from following row there are individual constraints. Although this format is not absolutely resistant to wrong model loading, when there is used not correct data format, after consultations with information system developers was considered as sufficient. Because the mathematical model construction will be in information system realized automatically, the probability of making such model is extremely low. This problem can occur (with relatively high probability), when the mathematical model is constructed manually by user.

3.3 Output data format

Regarding to a fact, that program output (i.e. optimal solution) will be processed in information system, it was necessary to agree with developers on output format. It is very simple: optimal solution values of all structural variables (include non-basic variables, i.e. zero valued variables) are written to output file, each variable on one row with precision of 8 decimal positions. Mathematical data format is not allowed. This format guarantees easy and correct loading of optimal solution into information system.

4 Discussion

Development of the application is up to date finished. With cooperation with information system developers, the small error eliminations and improvements are under way. During the developing process, the great accent was focused out on absolute possible results precision and elimination of cycling problem.

The maximal precision of results was solved by using of real48 data type. According to Delphi-basics [3], it is the floating point type with the highest capacity and precision.

The more difficult problem to solve was elimination of potential cycling. In a practical use, this could cause serious problems. This problem was solved by alternative way of pivot selection. It was used combination of original and Blend's rule. When there was used standard pivot selection and the solved problem was degenerate, the algorithm cycled relatively often. By small change (when objective function change is 0, we choose pivot column breaking optimality criterion with lowest index) was occurrence of cycling eliminated. The program itself was tested on examples, where there is a high probability of cycling: assignment problems solved as LP problems by simplex method. Those problems are strongly degenerate and therefore here very often occur cycling. During testing the program was solving of different assignment problems up to 30x30 dimensions (reason, why there were not tested larger problems consists from fact, that solving time grew exponentially with growing dimensions of matrix). While in original way of pivot selection the cycles occurred often in 7x7 matrix and program had to be terminated manually, improved version never cycled.

The last serious problem, that has not been solved yet, but will have to be solved together with information system developers and blending company employees, is infeasibility tolerance. In blending problem regularly occurs situation, that optimal solution of given blending problem does not exist. But when one or more of the constraints is softened, problem has feasible solution. The plan for future is to let program automatically decide, when small breach of any constraint is yet "all right" and when no. Now, the decision process fully depends on program user.

5 Conclusion

Although the program development has not been finished, the project state goes to finish and is program is prepared to be used in information system. It is possible that in practical use there occur problems with program they will have to be solved. But, those problems are not known yet. During testing the program works stable and program results correspond with results obtained from commercial optimization software. But, for real practical experience it will be necessary to wait.

References

[1] Blend, R. J.: *New finite pivoting rules for the Simplex method*. Mathematics in operations research, Binghamton, 1977. Available on http://sci.sut.ac.ir/People%5CCourses%5C42%5Cbland-rule.pdf.

[2] Todd M. J.: *Cycling in the Simplex method*. Available on http://people.orie.cornell.edu/~miketodd/or630/SimplexCyclingExample.pdf

[3] *Delphi basics*. Delphi manual, available on http://www.delphibasics.co.uk/RTL.asp?Name=Real48

[4] Jablonský, J. *Operační výzkum : kvantitativní modely pro ekonomické rozhodování.* 3. vyd. Praha: Professional Publishing, 2007. 323 s. ISBN 978-80-86946-44-3.

[5] Stevenson, W. J. -- Ozgur, C. Introduction to management science with spreadsheets. Boston: McGraw-Hill/Irwin, 2007. 812 s. ISBN 978-0-07-325290-2.