Simplex algorithm

From Wikipedia, the free encyclopedia

In mathematical optimization, Dantzig's **simplex algorithm** (or **simplex method**) is a popular algorithm for linear programming.[1][2][3][4][5] The journal *Computing in Science and Engineering* listed it as one of the top 10 algorithms of the twentieth century.[6]

The name of the algorithm is derived from the concept of a simplex and was suggested by T. S. Motzkin.[7] Simplices are not actually used in the method, but one interpretation of it is that it operates on simplicial *cones*, and these become proper simplices with an additional constraint.[8][9][10][11] The simplicial cones in question are the corners (i.e., the neighborhoods of the vertices) of a geometric object called a polytope. The shape of this polytope is defined by the constraints applied to the objective function.

Contents

Overview

The simplex algorithm operates on linear programs in standard form:

Maximize

$$\mathbf{c^T} \cdot \mathbf{x}$$

Subject to

$$\mathbf{Ax} \le \mathbf{b},\ x_i \ge 0$$

with $\mathbf{x} = (x_1, \ldots, x_n)$ the variables of the problem, $\mathbf{c} = (c_1, \ldots, c_n)$ are the coefficients of the objective function, $\mathbf{A}$ is a $p \times n$ matrix, and $\mathbf{b} = (b_1, \ldots, b_p)$ constants with $b_j \geq 0$. There is a straightforward process to convert any linear program into one in standard form so this results in no loss of generality.

In geometric terms, the feasible region defined by all values of $\mathbf{x}$ such that

$$\mathbf{Ax} \leq \mathbf{b}, \, x_i \geq 0$$

is a (possibly unbounded) convex polytope. There is a simple characterization of the extreme points or vertices of this polytope, namely $\mathbf{x} = (x_1, \ldots, x_n)$ is an extreme point if and only if the subset of column vectors A_i corresponding to the nonzero entries of $\mathbf{x}$ ($x_i \neq 0$) are linearly independent.[12] In this context such a point is known as a *basic feasible solution* (BFS).
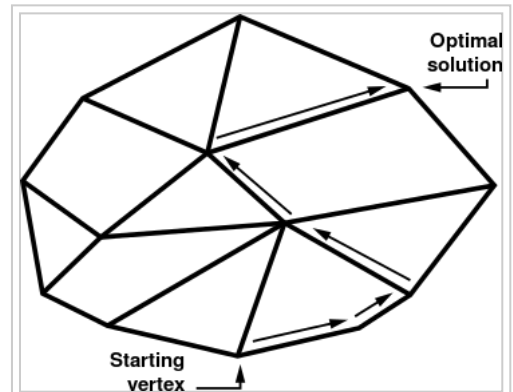
It can be shown that for a linear program in standard form, if the objective function has a maximum value on the feasible region then it has this value on (at least) one of the extreme points.[13] This in itself reduces the problem to a finite computation since there is a finite number of extreme points, but the number of extreme points is unmanageably large for all but the smallest linear programs.[14]

It can also be shown that if an extreme point is not a maximum point of the objective function then there is an edge containing the point so that the objective function is strictly increasing on the edge moving away from the point.[15] If the edge is finite then the edge connects to another extreme point where the objective function has a greater value, otherwise the objective function is unbounded above on the edge and the linear program has no solution. The simplex algorithm applies this insight by walking along edges of the polytope to extreme points with greater and greater objective values. This continues until the maximum value is reached or an unbounded edge is visited, concluding that the problem has no solution. The algorithm always terminates because the number of vertices in the polytope is finite; moreover since we jump between vertices always in the same direction (that of the objective function), we hope that the number of vertices visited will be small.[15]

A system of linear inequalities defines a polytope as a feasible region. The simplex algorithm begins at a starting vertex and moves along the edges of the polytope until it reaches the vertex of the optimum solution.

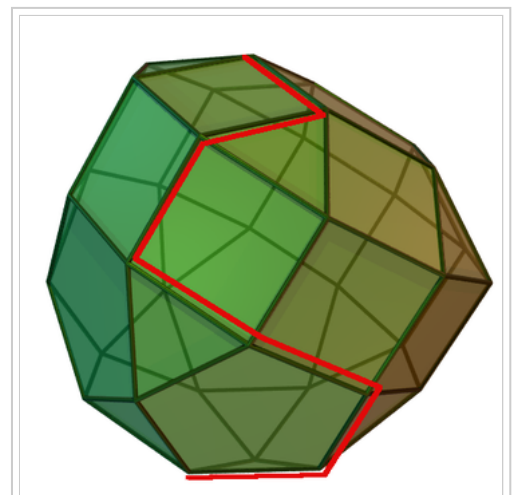Polyhedron of simplex algorithm in 3D

The solution of a linear program is accomplished in two steps. In the first step, known as Phase I, a starting extreme point is found. Depending on the nature of the program this may be trivial, but in general it can be solved by applying the simplex algorithm to a modified version of the original program. The possible results of Phase I are either that a basic feasible solution is found or that the feasible region is empty. In the latter case the linear program is called *infeasible*. In the second step, Phase II, the simplex algorithm is applied using the basic feasible solution found in Phase I as a starting point. The possible results from Phase II are either an optimum basic feasible solution or an infinite edge on which the objective function is unbounded below.[3][16][17]

History

George Dantzig worked on planning methods for the US Army Air Force during World War II using a desk calculator. During 1946 his colleague challenged him to mechanize the planning process in order to entice him into not taking another job. Dantzig formulated the problem as linear inequalities inspired by the work of Wassily Leontief, however, at that time he didn't include an objective as part of his formulation. Without an objective, a vast number of solutions can be feasible, and therefore to find the "best" feasible solution, military-

specified "ground rules" must be used that describe how goals can be achieved as opposed to specifying a goal itself. Dantzig's core insight was to realize that most such ground rules can be translated into a linear objective function that needs to be maximized.[18] Development of the simplex method was evolutionary and happened over a period of about a year.[19]

After Dantzig included an objective function as part of his formulation during mid-1947, the problem was mathematically more tractable. Dantzig realized that one of the unsolved problems that he mistook as homework in his professor Jerzy Neyman's class (and actually later solved), was applicable to finding an algorithm for linear programs. This problem involved finding the existence of Lagrange multipliers for general linear programs over a continuum of variables, each bounded between zero and one, and satisfying linear constraints expressed in the form of Lebesgue integrals. Dantzig later published his "homework" as a thesis to earn his doctorate. The column geometry used in this thesis gave Dantzig insight that made him believe that the Simplex method would be very efficient.[20]

Standard form

The transformation of a linear program to one in standard form may be accomplished as follows.[21] First, for each variable with a lower bound other than 0, a new variable is introduced representing the difference between the variable and bound. The original variable can then be eliminated by substitution. For example, given the constraint

$$x_1 \geq 5$$

a new variable, y_1, is introduced with

$$y_1 = x_1 - 5$$
$$x_1 = y_1 + 5$$

The second equation may be used to eliminate x_1 from the linear program. In this way, all lower bound constraints may be changed to non-negativity restrictions.

Second, for each remaining inequality constraint, a new variable, called a *slack variable*, is introduced to change the constraint to an equality constraint. This variable represents the difference between the two sides of the inequality and is assumed to be nonnegative. For example, the inequalities

$$x_2 + 2x_3 \leq 3$$
$$-x_4 + 3x_5 \geq 2$$

are replaced with

$$x_2 + 2x_3 + s_1 = 3$$
$$-x_4 + 3x_5 - s_2 = 2$$
$$s_1, s_2 \geq 0$$

It is much easier to perform algebraic manipulation on inequalities in this form. In inequalities where $\geq$ appears such as the second one, some authors refer to the variable introduced as a *surplus variable*.

Third, each unrestricted variable is eliminated from the linear program. This can be done in two ways, one is by solving for the variable in one of the equations in which it appears and then eliminating the variable by substitution. The other is to replace the variable with the difference of two restricted variables. For example, if z_1 is unrestricted then write

$$z_1 = z_1^+ - z_1^-$$
$$z_1^+, z_1^- \geq 0$$

The equation may be used to eliminate z_1 from the linear program.

When this process is complete the feasible region will be in the form

$$\mathbf{A}\mathbf{x} = \mathbf{b},\ x_i \geq 0$$

It is also useful to assume that the rank of $\mathbf{A}$ is the number of rows. This results in no loss of generality since otherwise either the system $\mathbf{A}\mathbf{x} = \mathbf{b}$ has redundant equations which can be dropped, or the system is inconsistent and the linear program has no solution.[22]

Simplex tableau

A linear program in standard form can be represented as a *tableau* of the form

$$\begin{bmatrix} 1 & -\mathbf{c}^T & 0 \\ 0 & \mathbf{A} & \mathbf{b} \end{bmatrix}$$

The first row defines the objective function and the remaining rows specify the constraints. (Note, different authors use different conventions as to the exact layout.) If the columns of A can be rearranged so that it contains the identity matrix of order p (the number of rows in A) then the tableau is said to be in *canonical form*.[23] The variables corresponding to the columns of the identity matrix are called *basic variables* while the remaining variables are called *nonbasic* or *free variables*. If the values of the nonbasic variables are set to 0, then the values of the basic variables are easily obtained as entries in b and this solution is a basic feasible solution. The algebraic interpretation here is that the coefficients of the linear equation represented by each row are either 0, 1, or some other number. Each row will have 1 column with value 1, $p-1$ columns with coefficients 0, and the remaining columns with some other coefficients (these other variables represent our non-basic variables). By setting the values of the non-basic variables we ensure in each row that the value of the variable represented by a 1 in its column is equal to the b value at that row.

Conversely, given a basic feasible solution, the columns corresponding to the nonzero variables can be expanded to a nonsingular matrix. If the corresponding tableau is multiplied by the inverse of this matrix then the result is a tableau in canonical form.[24]

Let

$$\begin{bmatrix} 1 & -\mathbf{c}_B^T & -\mathbf{c}_D^T & 0 \\ 0 & I & \mathbf{D} & \mathbf{b} \end{bmatrix}$$

be a tableau in canonical form. Additional row-addition transformations can be applied to remove the coefficients c_B^{T} from the objective function. This process is called *pricing out* and results in a canonical tableau

$$\begin{bmatrix} 1 & 0 & -\bar{\mathbf{c}}_D^T & z_B \\ 0 & I & \mathbf{D} & \mathbf{b} \end{bmatrix}$$

where z_B is the value of the objective function at the corresponding basic feasible solution. The updated coefficients, also known as *relative cost coefficients*, are the rates of change of the objective function with respect to the nonbasic variables.[16]

Pivot operations

The geometrical operation of moving from a basic feasible solution to an adjacent basic feasible solution is implemented as a *pivot operation*. First, a nonzero *pivot element* is selected in a nonbasic column. The row containing this element is multiplied by its reciprocal to change this element to 1, and then multiples of the row are added to the other rows to change the other entries in the column to 0. The result is that, if the pivot element is in row *r*, then the column becomes the *r*-th column of the identity matrix. The variable for this column is now a basic variable, replacing the variable which corresponded to the *r*-th column of the identity matrix before the operation. In effect, the variable corresponding to the pivot column enters the set of basic variables and is called the *entering variable*, and the variable being replaced leaves the set of basic variables and is called the *leaving variable*. The tableau is still in canonical form but with the set of basic variables changed by one element.[3][16]

Algorithm

Let a linear program be given by a canonical tableau. The simplex algorithm proceeds by performing successive pivot operations which each give an improved basic feasible solution; the choice of pivot element at each step is largely determined by the requirement that this pivot improve the solution.

Entering variable selection

Since the entering variable will, in general, increase from 0 to a positive number, the value of the objective function will decrease if the derivative of the objective function with respect to this variable is negative. Equivalently, the value of the objective function is decreased if the pivot column is selected so that the corresponding entry in the objective row of the tableau is positive.

If there is more than one column so that the entry in the objective row is positive then the choice of which one to add to the set of basic variables is somewhat arbitrary and several *entering variable choice rules*[25] such as Devex algorithm[26] have been developed.

If all the entries in the objective row are less than or equal to 0 then no choice of entering variable can be made and the solution is in fact optimal. It is easily seen to be optimal since the objective row now corresponds to an equation of the form

$$z(\mathbf{x}) = z_B + \text{nonnegative terms corresponding to nonbasic variables}$$

Note that by changing the entering variable choice rule so that it selects a column where the entry in the objective row is negative, the algorithm is changed so that it finds the maximum of the objective function rather than the minimum.

Leaving variable selection

Once the pivot column has been selected, the choice of pivot row is largely determined by the requirement that the resulting solution be feasible. First, only positive entries in the pivot column are considered since this guarantees that the value of the entering variable will be nonnegative. If there are no positive entries in the pivot column then the entering variable can take any nonnegative value with the solution remaining feasible. In this case the objective function is unbounded below and there is no minimum.

Next, the pivot row must be selected so that all the other basic variables remain positive. A calculation shows that this occurs when the resulting value of the entering variable is at a minimum. In other words, if the pivot column is *c*, then the pivot row *r* is chosen so that

$$b_r / a_{rc}$$

is the minimum over all r so that $a_{rc} > 0$. This is called the *minimum ratio test*.[25] If there is more than one row for which the minimum is achieved then a *dropping variable choice rule*[27] can be used to make the determination.

Example

Consider the linear program

Minimize

$$Z = -2x - 3y - 4z$$

Subject to

$$3x + 2y + z \leq 10$$
$$2x + 5y + 3z \leq 15$$
$$x, y, z \geq 0$$

With the addition of slack variables s and t, this is represented by the canonical tableau

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 0 & 0 & 0 \\ 0 & 3 & 2 & 1 & 1 & 0 & 10 \\ 0 & 2 & 5 & 3 & 0 & 1 & 15 \end{bmatrix}$$

where columns 5 and 6 represent the basic variables s and t and the corresponding basic feasible solution is

$$x = y = z = 0, \ s = 10, \ t = 15.$$

Columns 2, 3, and 4 can be selected as pivot columns, for this example column 4 is selected. The values of z resulting from the choice of rows 2 and 3 as pivot rows are 10/1 = 10 and 15/3 = 5 respectively. Of these the minimum is 5, so row 3 must be the pivot row. Performing the pivot produces

$$\begin{bmatrix} 1 & -\frac{2}{3} & -\frac{11}{3} & 0 & 0 & -\frac{4}{3} & -20 \\ 0 & \frac{7}{3} & \frac{1}{3} & 0 & 1 & -\frac{1}{3} & 5 \\ 0 & \frac{2}{3} & \frac{5}{3} & 1 & 0 & \frac{1}{3} & 5 \end{bmatrix}$$

Now columns 4 and 5 represent the basic variables z and s and the corresponding basic feasible solution is

$$x = y = t = 0, \ z = 5, \ s = 5.$$

For the next step, there are no positive entries in the objective row and in fact

$$Z = -20 + \frac{2}{3}x + \frac{11}{3}y + \frac{4}{3}t$$

so the minimum value of Z is −20.

Finding an initial canonical tableau

In general, a linear program will not be given in canonical form and an equivalent canonical tableau must be found before the simplex algorithm can start. This can be accomplished by the introduction of *artificial variables*. Columns of the identity matrix are added as column vectors for these variables. If the b value for a constraint equation is negative, the equation is negated before adding the identity matrix columns. This does not change the set of feasible solutions or the optimal solution, and it ensures that the slack variables will

constitute an initial feasible solution. The new tableau is in canonical form but it is not equivalent to the original problem. So a new objective function, equal to the sum of the artificial variables, is introduced and the simplex algorithm is applied to find the minimum; the modified linear program is called the *Phase I* problem.[28]

The simplex algorithm applied to the Phase I problem must terminate with a minimum value for the new objective function since, being the sum of nonnegative variables, its value is bounded below by 0. If the minimum is 0 then the artificial variables can be eliminated from the resulting canonical tableau producing a canonical tableau equivalent to the original problem. The simplex algorithm can then be applied to find the solution; this step is called *Phase II*. If the minimum is positive then there is no feasible solution for the Phase I problem where the artificial variables are all zero. This implies that the feasible region for the original problem is empty, and so the original problem has no solution.[3][16][29]

Example

Consider the linear program

Minimize

$$Z = -2x - 3y - 4z$$

Subject to

$$3x + 2y + z = 10$$
$$2x + 5y + 3z = 15$$
$$x, y, z \geq 0$$

This is represented by the (non-canonical) tableau

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 0 \\ 0 & 3 & 2 & 1 & 10 \\ 0 & 2 & 5 & 3 & 15 \end{bmatrix}$$

Introduce artificial variables u and v and objective function $W = u + v$, giving a new tableau

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & -1 & -1 & 0 \\ 0 & 1 & 2 & 3 & 4 & 0 & 0 & 0 \\ 0 & 0 & 3 & 2 & 1 & 1 & 0 & 10 \\ 0 & 0 & 2 & 5 & 3 & 0 & 1 & 15 \end{bmatrix}$$

Note that the equation defining the original objective function is retained in anticipation of Phase II.

After pricing out this becomes

$$\begin{bmatrix} 1 & 0 & 5 & 7 & 4 & 0 & 0 & 25 \\ 0 & 1 & 2 & 3 & 4 & 0 & 0 & 0 \\ 0 & 0 & 3 & 2 & 1 & 1 & 0 & 10 \\ 0 & 0 & 2 & 5 & 3 & 0 & 1 & 15 \end{bmatrix}$$

Select column 5 as a pivot column, so the pivot row must be row 4, and the updated tableau is

$$\begin{bmatrix} 1 & 0 & \frac{7}{3} & \frac{1}{3} & 0 & 0 & -\frac{4}{3} & 5 \\ 0 & 1 & -\frac{2}{3} & -\frac{11}{3} & 0 & 0 & -\frac{4}{3} & -20 \\ 0 & 0 & \frac{7}{3} & \frac{1}{3} & 0 & 1 & -\frac{1}{3} & 5 \\ 0 & 0 & \frac{2}{3} & \frac{5}{3} & 1 & 0 & \frac{1}{3} & 5 \end{bmatrix}$$

Now select column 3 as a pivot column, for which row 3 must be the pivot row, to get

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & -1 & -1 & 0 \\ 0 & 1 & 0 & -\frac{25}{7} & 0 & \frac{2}{7} & -\frac{10}{7} & -\frac{130}{7} \\ 0 & 0 & 1 & \frac{1}{7} & 0 & \frac{3}{7} & -\frac{1}{7} & \frac{15}{7} \\ 0 & 0 & 0 & \frac{11}{7} & 1 & -\frac{2}{7} & \frac{3}{7} & \frac{25}{7} \end{bmatrix}$$

The artificial variables are now 0 and they may be dropped giving a canonical tableau equivalent to the original problem:

$$\begin{bmatrix} 1 & 0 & -\frac{25}{7} & 0 & -\frac{130}{7} \\ 0 & 1 & \frac{1}{7} & 0 & \frac{15}{7} \\ 0 & 0 & \frac{11}{7} & 1 & \frac{25}{7} \end{bmatrix}$$

This is, fortuitously, already optimal and the optimum value for the original linear program is −130/7.

Advanced topics

Implementation

The tableau form used above to describe the algorithm lends itself to an immediate implementation in which the tableau is maintained as a rectangular (m + 1)-by-(m + n + 1) array. It is straightforward to avoid storing the m explicit columns of the identity matrix that will occur within the tableau by virtue of **B** being a subset of the columns of [**A**, **I**]. This implementation is referred to as the "*standard* simplex algorithm". The storage and computation overhead are such that the standard simplex method is a prohibitively expensive approach to solving large linear programming problems.

In each simplex iteration, the only data required are the first row of the tableau, the (pivotal) column of the tableau corresponding to the entering variable and the right-hand-side. The latter can be updated using the pivotal column and the first row of the tableau can be updated using the (pivotal) row corresponding to the leaving variable. Both the pivotal column and pivotal row may be computed directly using the solutions of linear systems of equations involving the matrix **B** and a matrix-vector product using **A**. These observations motivate the "*revised* simplex algorithm", for which implementations are distinguished by their invertible representation of **B**.[4]

In large linear-programming problems **A** is typically a sparse matrix and, when the resulting sparsity of **B** is exploited when maintaining its invertible representation, the revised simplex algorithm is a much more efficient than the standard simplex method. Commercial simplex solvers are based on the revised simplex algorithm.[4][29][30][31][32]

Degeneracy: stalling and cycling

If the values of all basic variables are strictly positive, then a pivot must result in an improvement in the objective value. When this is always the case no set of basic variables occurs twice and the simplex algorithm must terminate after a finite number of steps. Basic feasible solutions where at least one of the *basic* variables is zero are called *degenerate* and may result in pivots for which there is no improvement in the objective value. In this case there is no actual change in the solution but only a change in the set of basic variables. When several such pivots occur in succession, there is no improvement; in large industrial applications, degeneracy is common and such "*stalling*" is notable. Worse than stalling is the possibility the same set of basic variables occurs twice, in which case, the deterministic pivoting rules of the simplex algorithm will produce an infinite loop, or "cycle". While degeneracy is the rule in practice and stalling is common, cycling is rare in practice. A discussion of an example of practical cycling occurs in Padberg.[29] Bland's rule prevents cycling and thus guarantees that the simplex algorithm always terminates.[29][33][34] Another pivoting algorithm, the criss-cross algorithm never cycles on linear programs.[35]

Efficiency

The simplex method is remarkably efficient in practice and was a great improvement over earlier methods such as Fourier–Motzkin elimination. However, in 1972, Klee and Minty[36] gave an example, the Klee-Minty cube, showing that the worst-case complexity of simplex method as formulated by Dantzig is exponential time. Since then, for almost every variation on the method, it has been shown that there is a family of linear programs for which it performs badly. It is an open question if there is a variation with polynomial time, or even sub-exponential worst-case complexity.[37][38]

Analyzing and quantifying the observation that the simplex algorithm is efficient in practice, even though it has exponential worst-case complexity, has led to the development of other measures of complexity. The simplex algorithm has polynomial-time average-case complexity under various probability distributions, with the precise average-case performance of the simplex algorithm depending on the choice of a probability distribution for the random matrices.[38][39] Another approach to studying "typical phenoma" uses Baire category theory from general topology, and to show that (topologically) "most" matrices can be solved by the simplex algorithm in a polynomial number of steps. Another method to analyze the performance of the simplex algorithm studies the behavior of worst-case scenarios under small perturbation – are worst-case scenarios stable under a small change (in the sense of structural stability), or do they become tractable? Formally, this method uses random problems to which is added a Gaussian random vector ("smoothed complexity").[40]

Other algorithms

Other algorithms for solving linear-programming problems are described in the linear-programming article. Another basis-exchange pivoting algorithm is the criss-cross algorithm.[41][42] There are polynomial-time algorithms for linear programming that use interior point methods: These include Khachiyan's ellipsoidal algorithm, Karmarkar's projective algorithm, and path-following algorithms.[17]

Linear-fractional programming

Linear–fractional programming (LFP) is a generalization of linear programming (LP). In LP the objective function is a linear function, while the objective function of a linear–fractional program is a ratio of two linear functions. In other words, a linear program is a fractional–linear program in which the denominator is the constant function having the value one everywhere. A linear–fractional program can be solved by a variant of the simplex algorithm[43][44][45][46] or by the criss-cross algorithm.[47]

See also

- Criss-cross algorithm

- Cutting-plane method
- Devex algorithm
- Fourier–Motzkin elimination
- Karmarkar's algorithm
- Nelder–Mead simplicial heuristic
- Pivoting rule of Bland, which avoids cycling

Notes

1. Murty, Katta G. (1983). *Linear programming*. New York: John Wiley & Sons Inc. pp. xix+482. ISBN 0-471-09725-X. MR 720547.
2. Richard W. Cottle, ed. *The Basic George B. Dantzig*. Stanford Business Books, Stanford University Press, Stanford, California, 2003. (Selected papers by George B. Dantzig)
3. George B. Dantzig and Mukund N. Thapa. 1997. *Linear programming 1: Introduction*. Springer-Verlag.
4. George B. Dantzig and Mukund N. Thapa. 2003. *Linear Programming 2: Theory and Extensions*. Springer-Verlag.
5. Michael J. Todd (February 2002). "The many facets of linear programming". *Mathematical Programming*. **91** (3): 417–436. doi:10.1007/s101070100261. (Invited survey, from the International Symposium on Mathematical Programming.)
6. *Computing in Science and Engineering*, volume 2, no. 1, 2000 html version (http://www.computer.org/csdl/mags/cs/2000/01/c1022.html)
7. Murty (1983, Comment 2.2)
8. Murty (1983, Note 3.9)
9. Stone, Richard E.; Tovey, Craig A. (1991). "The simplex and projective scaling algorithms as iteratively reweighted least squares methods". *SIAM Review*. **33** (2): 220–237. doi:10.1137/1033049. JSTOR 2031142. MR 1124362.
10. Stone, Richard E.; Tovey, Craig A. (1991). "Erratum: The simplex and projective scaling algorithms as iteratively reweighted least squares methods". *SIAM Review*. **33** (3): 461. doi:10.1137/1033100. JSTOR 2031443. MR 1124362.
11. Strang, Gilbert (1 June 1987). "Karmarkar's algorithm and its place in applied mathematics". *The Mathematical Intelligencer*. New York: Springer. **9** (2): 4–10. doi:10.1007/BF03025891. ISSN 0343-6993. MR '"883185"'.
12. Murty (1983, Theorem 3.1)
13. Murty (1983, Theorem 3.3)
14. Murty (1983, p. 143, Section 3.13)
15. Murty (1983, p. 137, Section 3.8)
16. Evar D. Nering and Albert W. Tucker, 1993, *Linear Programs and Related Problems*, Academic Press. (elementary)
17. Robert J. Vanderbei, *Linear Programming: Foundations and Extensions* (http://www.princeton.edu/~rvdb/LPbook/), 3rd ed., International Series in Operations Research & Management Science, Vol. 114, Springer Verlag, 2008. ISBN 978-0-387-74387-5.
18. "Reminiscences about the origins of linear programming" (PDF). *Operations Research Letter*. **1** (2): 43–48. April 1982. doi:10.1016/0167-6377(82)90043-8.
19. Albers and Reid (1986). "An Interview with George B. Dantzig: The Father of Linear Programming". *College Mathematics Journal*: 292–314.
20. Dantzig, George (May 1987). "Origins of the simplex method" (PDF). *A history of scientific computing*. doi:10.1145/87252.88081. ISBN 0-201-50814-1.
21. Murty (1983, Section 2.2)
22. Murty (1983, p. 173)
23. Murty (1983, section 2.3.2)
24. Murty (1983, section 3.12)
25. Murty (1983, p. 66)
26. Harris, Paula MJ. "Pivot selection methods of the Devex LP code." Mathematical programming 5.1 (1973): 1–28
27. Murty (1983, p. 67)
28. Murty (1983, p. 60)
29. M. Padberg, *Linear Optimization and Extensions*, Second Edition, Springer-Verlag, 1999.
30. Dmitris Alevras and Manfred W. Padberg, *Linear Optimization and Extensions: Problems and Extensions*, Universitext, Springer-Verlag, 2001. (Problems from Padberg with solutions.)
31. Maros, István; Mitra, Gautam (1996). "Simplex algorithms". In J. E. Beasley. *Advances in linear and integer programming*. Oxford Science. pp. 1–46. MR 1438309.
32. Maros, István (2003). *Computational techniques of the simplex method*. International Series in Operations Research & Management Science. **61**. Boston, MA: Kluwer Academic Publishers. pp. xx+325. ISBN 1-4020-7332-1. MR 1960274.
33. Bland, Robert G. (May 1977). "New finite pivoting rules for the simplex method". *Mathematics of Operations Research*. **2** (2): 103–107. doi:10.1287/moor.2.2.103. JSTOR 3689647. MR 459599.
34. Murty (1983, p. 79)
35. There are abstract optimization problems, called oriented matroid programs, on which Bland's rule cycles (incorrectly) while the criss-cross algorithm terminates correctly.

36. Klee, Victor; Minty, George J. (1972). "How good is the simplex algorithm?". In Shisha, Oved. *Inequalities III (Proceedings of the Third Symposium on Inequalities held at the University of California, Los Angeles, Calif., September 1–9, 1969, dedicated to the memory of Theodore S. Motzkin)*. New York-London: Academic Press. pp. 159–175. MR 332165.

37. Christos H. Papadimitriou and Kenneth Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*, Corrected republication with a new preface, Dover. (computer science)

38. Alexander Schrijver, *Theory of Linear and Integer Programming*. John Wiley & sons, 1998, ISBN 0-471-98232-6 (mathematical)

39. The simplex algorithm takes on average *D* steps for a cube. Borgwardt (1987): Borgwardt, Karl-Heinz (1987). *The simplex method: A probabilistic analysis*. Algorithms and Combinatorics (Study and Research Texts). **1**. Berlin: Springer-Verlag. pp. xii+268. ISBN 3-540-17096-0. MR 868467.

40. Spielman, Daniel; Teng, Shang-Hua (2001). "Smoothed analysis of algorithms: why the simplex algorithm usually takes polynomial time". *Proceedings of the Thirty-Third Annual ACM Symposium on Theory of Computing*. ACM. pp. 296–305. arXiv:cs/0111050. doi:10.1145/380752.380813. ISBN 978-1-58113-349-3.

41. Terlaky, Tamás; Zhang, Shu Zhong (1993). "Pivot rules for linear programming: A Survey on recent theoretical developments". *Annals of Operations Research*. Springer Netherlands. 46–47 (1): 203–233. CiteSeerX 10.1.1.36.7658. doi:10.1007/BF02096264. ISSN 0254-5330. MR 1260019.

42. Fukuda, Komei; Terlaky, Tamás (1997). Thomas M. Liebling and Dominique de Werra, eds. "Criss-cross methods: A fresh view on pivot algorithms". *Mathematical Programming: Series B*. **79** (1—3). Amsterdam: North-Holland Publishing Co. pp. 369–395. doi:10.1007/BF02614325. MR 1464775.

43. Murty (1983, Chapter 3.20 (pp. 160–164) and pp. 168 and 179)

44. Chapter five: Craven, B. D. (1988). *Fractional programming*. Sigma Series in Applied Mathematics. **4**. Berlin: Heldermann Verlag. p. 145. ISBN 3-88538-404-3. MR 949209.

45. Kruk, Serge; Wolkowicz, Henry (1999). "Pseudolinear programming". *SIAM Review*. **41** (4): 795–805. doi:10.1137/S0036144598335259. JSTOR 2653207. MR 1723002.

46. Mathis, Frank H.; Mathis, Lenora Jane (1995). "A nonlinear programming algorithm for hospital management". *SIAM Review*. **37** (2): 230–234. doi:10.1137/1037046. JSTOR 2132826. MR 1343214.

47. Illés, Tibor; Szirmai, Ákos; Terlaky, Tamás (1999). "The finite criss-cross method for hyperbolic programming". *European Journal of Operational Research*. **114** (1): 198–214. doi:10.1016/S0377-2217(98)00049-6. ISSN 0377-2217. PDF preprint.

References

- Murty, Katta G. (1983). *Linear programming*. New York: John Wiley & Sons, Inc. pp. xix+482. ISBN 0-471-09725-X. MR 720547.

Further reading

These introductions are written for students of computer science and operations research:

- Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*, Second Edition. MIT Press and McGraw-Hill, 2001. ISBN 0-262-03293-7. Section 29.3: The simplex algorithm, pp. 790–804.
- Frederick S. Hillier and Gerald J. Lieberman: *Introduction to Operations Research*, 8th edition. McGraw-Hill. ISBN 0-07-123828-X
- Rardin, Ronald L. (1997). *Optimization in operations research*. Prentice Hall. p. 919. ISBN 0-02-398415-5.

External links

- An Introduction to Linear Programming and the Simplex Algorithm (http://www.isye.gatech.edu/~spyros/LP/LP.html) by Spyros Reveliotis of the Georgia Institute of Technology.

The Wikibook *Operations Research* has a page on the topic of: *The Simplex Method*

- Greenberg, Harvey J., *Klee-Minty Polytope Shows Exponential Time Complexity of Simplex Method* University of Colorado at Denver (1997) PDF download (http://glossary.computing.society.informs.org/notes/Klee-Minty.pdf)
- Simplex Method (http://www.lokminglui.com/lpch3.pdf) A tutorial for Simplex Method with examples (also two-phase and M-method).
- Example of Simplex Procedure for a Standard Linear Programming Problem (http://math.uww.edu/~mcfarlat/s-prob.htm) by Thomas McFarland of the University of Wisconsin-Whitewater.
- PHPSimplex: online tool to solve Linear Programming Problems (http://www.phpsimplex.com/simplex/simplex.htm?l=en) by Daniel Izquierdo and Juan José Ruiz of the University of Málaga (UMA, Spain)
- simplex-m (http://www.simplex-m.com) Online Simplex Solver

Retrieved from "https://en.wikipedia.org/w/index.php?title=Simplex_algorithm&oldid=747259424"

Categories: Optimization algorithms and methods │ Operations research │ 1947 in computer science │ Exchange algorithms │ Linear programming │ Computer-related introductions in 1947

- This page was last modified on 1 November 2016, at 11:08.
- Text is available under the Creative Commons Attribution-ShareAlike License; additional terms may apply. By using this site, you agree to the Terms of Use and Privacy Policy. Wikipedia® is a registered trademark of the Wikimedia Foundation, Inc., a non-profit organization.