Linear programming

From Wikipedia, the free encyclopedia

Linear programming (**LP**, or **linear optimization**) is a mathematical method for determining a way to achieve the best outcome (such as maximum profit or lowest cost) in a given mathematical model for some list of requirements represented as linear relationships. Linear programming is a specific case of mathematical programming (mathematical optimization).

More formally, linear programming is a technique for the optimization of a linear objective function, subject to linear equality and linear inequality constraints. Its feasible region is a convex polyhedron, which is a set defined as the intersection of finitely many half spaces, each of which is defined by a linear inequality. Its objective function is a real-valued affine function defined on this polyhedron. A linear programming algorithm finds a point in the polyhedron where this function has the smallest (or largest) value if such point exists.

Linear programs are problems that can be expressed in canonical form:

$$\begin{aligned} \text{maximize} \quad & \mathbf{c}^{\mathrm{T}}\mathbf{x} \\ \text{subject to} \quad & A\mathbf{x} \leq \mathbf{b} \\ \text{and} \quad & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

where **x** represents the vector of variables (to be determined), **c** and **b** are vectors of (known) coefficients and *A* is a (known) matrix of coefficients. The expression to be maximized or minimized is called the *objective function* ($\mathbf{c}^{\mathrm{T}}\mathbf{x}$ in this case). The equations $A\mathbf{x} \leq \mathbf{b}$ are the constraints which specify a convex polytope over which the objective function is to be optimized. (In this context, two vectors are comparable when every entry in one is less-than or equal-to the corresponding entry in the other. Otherwise, they are incomparable.)

Linear programming can be applied to various fields of study. It is used most extensively in business and economics, but can also be utilized for some engineering problems. Industries that use linear programming models include transportation, energy, telecommunications, and manufacturing. It has proved useful in modeling diverse types of problems in planning, routing, scheduling, assignment, and design.

Contents

- 1 History
- 2 Uses
- 3 Standard form
  - 3.1 Example
- 4 Augmented form (slack form)
  - 4.1 Example
- 5 Duality
  - 5.1 Example
  - 5.2 Another example

- 6 Covering-packing dualities
    - 6.1 Examples
- 7 Complementary slackness
- 8 Theory
    - 8.1 Existence of optimal solutions
    - 8.2 Optimal vertices (and rays) of polyhedra
- 9 Algorithms
    - 9.1 Basis exchange algorithms
        - 9.1.1 Simplex algorithm of Dantzig
        - 9.1.2 Criss-cross algorithm
    - 9.2 Interior point
        - 9.2.1 Ellipsoid algorithm, following Khachiyan
        - 9.2.2 Projective algorithm of Karmarkar
        - 9.2.3 Path-following algorithms
    - 9.3 Comparison of interior-point methods versus simplex algorithms
- 10 Open problems and recent work
- 11 Integer unknowns
- 12 Integral linear programs
- 13 Solvers and scripting (programming) languages
- 14 See also
- 15 Notes
- 16 References
- 17 Further reading
- 18 External links

History

The problem of solving a system of linear inequalities dates back at least as far as Fourier, after whom the method of Fourier-Motzkin elimination is named. Linear programming itself was first developed by Leonid Kantorovich, a Russian mathematician, in 1939.[1] It was used during World War II to plan expenditures and returns in order to reduce costs to the army and increase losses to the enemy. The method was kept secret until 1947 when George B. Dantzig published the simplex method and John von Neumann developed the theory of duality as a linear optimization solution, and applied it in the field of game theory. Postwar, many industries found its use in their daily planning.

The linear-programming problem was first shown to be solvable in polynomial time by Leonid Khachiyan in 1979, but a larger theoretical and practical breakthrough in the field came in 1984 when Narendra Karmarkar introduced a new interior-point method for solving linear-programming problems.

Dantzig's original example of finding the best assignment of 70 people to 70 jobs exemplifies the usefulness of linear programming. The computing power required to test all the permutations to select the best assignment is vast; the number of possible configurations exceeds the number of particles in the universe. However, it takes only a moment to find the optimum solution by posing the problem as a linear program and applying the Simplex algorithm. The theory behind linear programming drastically reduces the number of possible

optimal solutions that must be checked.

Uses

Linear programming is a considerable field of optimization for several reasons. Many practical problems in operations research can be expressed as linear programming problems. Certain special cases of linear programming, such as *network flow* problems and *multicommodity flow* problems are considered important enough to have generated much research on specialized algorithms for their solution. A number of algorithms for other types of optimization problems work by solving LP problems as sub-problems. Historically, ideas from linear programming have inspired many of the central concepts of optimization theory, such as *duality, decomposition,* and the importance of *convexity* and its generalizations. Likewise, linear programming is heavily used in microeconomics and company management, such as planning, production, transportation, technology and other issues. Although the modern management issues are ever-changing, most companies would like to maximize profits or minimize costs with limited resources. Therefore, many issues can be characterized as linear programming problems.

Standard form

Standard form is the usual and most intuitive form of describing a linear programming problem. It consists of the following four parts:

- A **linear function to be maximized**

e.g. $\max_{x1,x2} f(x_1, x_2) = c_1 x_1 + c_2 x_2$

- **Problem constraints** of the following form

e.g.

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 &\leq b_1 \\ a_{21}x_1 + a_{22}x_2 &\leq b_2 \\ a_{31}x_1 + a_{32}x_2 &\leq b_3 \end{aligned}$$

- **Non-negative variables**

e.g.

$$\begin{aligned} x_1 &\geq 0 \\ x_2 &\geq 0 \end{aligned}$$

- **Non-negative right hand side constants**

$$b_i \geq 0,\ i = 1, 2, 3$$

The problem is usually expressed in *matrix form*, and then becomes:

$$\max\{c^{\mathrm{T}} x \mid 0 \leq Ax \leq b \wedge x \geq 0\}$$

Other forms, such as minimization problems, problems with constraints on alternative forms, as well as problems involving negative variables can always be rewritten into an equivalent problem in standard form.

Example

Suppose that a farmer has a piece of farm land, say L km^2, to be planted with either wheat or barley or some combination of the two. The farmer has a limited amount of fertilizer, F kilograms, and insecticide, P kilograms. Every square kilometer of wheat requires F_1 kilograms of fertilizer, and P_1 kilograms of insecticide, while every square kilometer of barley requires F_2 kilograms of fertilizer, and P_2 kilograms of insecticide. Let S_1 be the selling price of wheat per square kilometer, and S_2 be the price of barley. If we denote the area of land planted with wheat and barley by x_1 and x_2 respectively, then profit can be maximized by choosing optimal values for x_1 and x_2. This problem can be expressed with the following linear programming problem in the standard form:

Maximize:	$S_1x_1 + S_2x_2$	(maximize the revenue—revenue is the "objective function")
Subject to:	$0 \le x_1 + x_2 \le L$	(limit on total area)
	$0 \le F_1x_1 + F_2x_2 \le F$	(limit on fertilizer)
	$0 \le P_1x_1 + P_2x_2 \le P$	(limit on insecticide)
	$x_1 \ge 0, x_2 \ge 0$	(cannot plant a negative area).

Which in matrix form becomes:

$$\text{maximize} \begin{bmatrix} S_1 & S_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$\text{subject to} \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \le \begin{bmatrix} 1 & 1 \\ F_1 & F_2 \\ P_1 & P_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \le \begin{bmatrix} L \\ F \\ P \end{bmatrix}, \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \ge \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

Augmented form (slack form)

Linear programming problems must be converted into *augmented form* before being solved by the simplex algorithm. This form introduces non-negative *slack variables* to replace inequalities with equalities in the constraints. The problem can then be written in the following block matrix form:

Maximize Z:

$$\begin{bmatrix} 1 & -\mathbf{c}^T & 0 \\ 0 & \mathbf{A} & \mathbf{I} \end{bmatrix} \begin{bmatrix} Z \\ \mathbf{x} \\ \mathbf{x}_s \end{bmatrix} = \begin{bmatrix} 0 \\ \mathbf{b} \end{bmatrix}$$

$\mathbf{x}, \mathbf{x}_S \geq 0$

where $\mathbf{x}_s$ are the newly introduced slack variables, and Z is the variable to be maximized.

Example

The example above is converted into the following augmented form:

Maximize:	$S_1\mathbf{x}_1 + S_2\mathbf{x}_2$	(objective function)
Subject to:	$\mathbf{x}_1 + \mathbf{x}_2 + \mathbf{x}_3 = L$	(augmented constraint)
	$F_1\mathbf{x}_1 + F_2\mathbf{x}_2 + \mathbf{x}_4 = F$	(augmented constraint)
	$P_1\mathbf{x}_1 + P_2\mathbf{x}_2 + \mathbf{x}_5 = P$	(augmented constraint)
	$\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5 \geq 0.$	

where $\mathbf{x}_3$, $\mathbf{x}_4$, $\mathbf{x}_5$ are (non-negative) slack variables, representing in this example the unused area, the amount of unused fertilizer, and the amount of unused insecticide.

In matrix form this becomes:

Maximize Z:

$$\begin{bmatrix} 1 & -S_1 & -S_2 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & F_1 & F_2 & 0 & 1 & 0 \\ 0 & P_1 & P_2 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} Z \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} 0 \\ L \\ F \\ P \end{bmatrix}, \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} \geq 0.$$

Duality

See also: Dual linear program

Every linear programming problem, referred to as a *primal* problem, can be converted into a dual problem, which provides an upper bound to the optimal value of the primal problem. In matrix form, we can express the *primal* problem as:

Maximize $\mathbf{c}^T\mathbf{x}$ subject to $A\mathbf{x} \leq \mathbf{b}$, $\mathbf{x} \geq 0$;

with the corresponding **symmetric** dual problem,

Minimize $\mathbf{b}^T\mathbf{y}$ subject to $A^T\mathbf{y} \geq \mathbf{c}$, $\mathbf{y} \geq 0$.

An alternative primal formulation is:

Maximize $\mathbf{c}^{\mathrm{T}}\mathbf{x}$ subject to $A\mathbf{x} \leq \mathbf{b}$;

with the corresponding **asymmetric** dual problem,

Minimize $\mathbf{b}^{\mathrm{T}}\mathbf{y}$ subject to $A^{\mathrm{T}}\mathbf{y} = \mathbf{c}$, $\mathbf{y} \geq 0$.

There are two ideas fundamental to duality theory. One is the fact that (for the symmetric dual) the dual of a dual linear program is the original primal linear program. Additionally, every feasible solution for a linear program gives a bound on the optimal value of the objective function of its dual. The weak duality theorem states that the objective function value of the dual at any feasible solution is always greater than or equal to the objective function value of the primal at any feasible solution. The strong duality theorem states that if the primal has an optimal solution, $\mathbf{x}^*$, then the dual also has an optimal solution, $\mathbf{y}^*$, such that $\mathbf{c}^{\mathrm{T}}\mathbf{x}^* = \mathbf{b}^{\mathrm{T}}\mathbf{y}^*$.

A linear program can also be unbounded or infeasible. Duality theory tells us that if the primal is unbounded then the dual is infeasible by the weak duality theorem. Likewise, if the dual is unbounded, then the primal must be infeasible. However, it is possible for both the dual and the primal to be infeasible (See also Farkas' lemma).

Example

Revisit the above example of the farmer who may grow wheat and barley with the set provision of some L land, F fertilizer and P insecticide. Assume now that unit prices for each of these means of production (inputs) are set by a planning board. The planning board's job is to minimize the total cost of procuring the set amounts of inputs while providing the farmer with a floor on the unit price of each of his crops (outputs), S_1 for wheat and S_2 for barley. This corresponds to the following linear programming problem:

Minimize:	$L\mathbf{y}_L + F\mathbf{y}_F + P\mathbf{y}_P$	(minimize the total cost of the means of production as the "objective function")
Subject to:	$\mathbf{y}_L + F_1\mathbf{y}_F + P_1\mathbf{y}_P \geq S_1$	(the farmer must receive no less than S_1 for his wheat)
	$\mathbf{y}_L + F_2\,\mathbf{y}_F + P_2\mathbf{y}_P \geq S_2$	(the farmer must receive no less than S_2 for his barley)
	$\mathbf{y}_L \geq 0$, $\mathbf{y}_F \geq 0$, $\mathbf{y}_P \geq 0$	(prices cannot be negative).

Which in matrix form becomes:

Minimize: $\begin{bmatrix} L & F & P \end{bmatrix} \begin{bmatrix} y_L \\ y_F \\ y_P \end{bmatrix}$

Subject to: $\begin{bmatrix} 1 & F_1 & P_1 \\ 1 & F_2 & P_2 \end{bmatrix} \begin{bmatrix} y_L \\ y_F \\ y_P \end{bmatrix} \geq \begin{bmatrix} S_1 \\ S_2 \end{bmatrix}, \quad \begin{bmatrix} y_L \\ y_F \\ y_P \end{bmatrix} \geq 0.$

The primal problem deals with physical quantities. With all inputs available in limited quantities, and assuming the unit prices of all outputs is known, what quantities of outputs to produce so as to maximize total revenue? The dual problem deals with economic values. With floor guarantees on all output unit prices, and assuming the available quantity of all inputs is known, what input unit pricing scheme to set so as to minimize total expenditure?

To each variable in the primal space corresponds an inequality to satisfy in the dual space, both indexed by output type. To each inequality to satisfy in the primal space corresponds a variable in the dual space, both indexed by input type.

The coefficients that bound the inequalities in the primal space are used to compute the objective in the dual space, input quantities in this example. The coefficients used to compute the objective in the primal space bound the inequalities in the dual space, output unit prices in this example.

Both the primal and the dual problems make use of the same matrix. In the primal space, this matrix expresses the consumption of physical quantities of inputs necessary to produce set quantities of outputs. In the dual space, it expresses the creation of the economic values associated with the outputs from set input unit prices.

Since each inequality can be replaced by an equality and a slack variable, this means each primal variable corresponds to a dual slack variable, and each dual variable corresponds to a primal slack variable. This relation allows us to complementary slackness.

Another example

Sometimes, one may find it more intuitive to obtain the dual program without looking at program matrix. Consider the following linear program:

$$\text{minimize} \quad \sum_{i=1}^{m} c_i x_i + \sum_{j=1}^{n} d_j t_j$$

$$\text{subject to} \quad \sum_{i=1}^{m} a_{ij} x_i + e_j t_j \geq g_j \quad , \ 1 \leq j \leq n$$

$$f_i x_i + \sum_{j=1}^{n} b_{ij} t_j \geq h_i \quad , \ 1 \leq i \leq m$$

$$x_i \geq 0,\, t_j \geq 0 \qquad , \ 1 \leq i \leq m, 1 \leq j \leq n$$

We have *m* + *n* conditions and all variables are non-negative. We shall define *m* + *n* dual variables: $\mathbf{y}_j$ and $\mathbf{s}_i$. We get:

$$\text{minimize} \quad \sum_{i=1}^{m} c_i x_i + \sum_{j=1}^{n} d_j t_j$$

$$\text{subject to} \qquad\qquad\qquad\qquad\qquad\qquad , \ 1 \leq j \leq n$$

$$\sum_{i=1}^{m} a_{ij} x_i \cdot y_n + e_j t_j \cdot y_j \geq g_j \cdot y_j$$
$$f_i x_i \cdot s_i + \sum_{j=1}^{} b_{ij} t_j \cdot s_i \geq h_i \cdot s_i \quad , \ 1 \leq i \leq m$$

$$x_i \geq 0,\ t_j \geq 0 \quad , \ 1 \leq i \leq m, 1 \leq j \leq n$$

$$y_j \geq 0,\ s_i \geq 0 \quad , \ 1 \leq j \leq n, 1 \leq i \leq m$$

Since this is a minimization problem, we would like to obtain a dual program that is a lower bound of the primal. In other words, we would like the sum of all right hand side of the constraints to be the maximal under the condition that for each primal variable the sum of its coefficients do not exceed its coefficient in the linear function. For example, $\mathbf{x}_1$ appears in $n + 1$ constraints. If we sum its constraints' coefficients we get $a_{1,1}\mathbf{y}_1 + a_{1,2}\mathbf{y}_2 + \ldots + a_{1,n}\mathbf{y}_n + f_1\mathbf{s}_1$. This sum must be at most $\mathbf{c}_1$. As a result we get:

maximize $\sum_{j=1}^{n} g_j y_j + \sum_{i=1}^{m} h_i s_i$

subject to $\sum_{j=1}^{n} a_{ij} y_j + f_i s_i \leq c_i \quad , \ 1 \leq i \leq m$

$$e_j y_j + \sum_{i=1}^{m} b_{ij} s_i \leq d_j \quad , \ 1 \leq j \leq n$$

$$y_j \geq 0,\ s_i \geq 0 \quad , \ 1 \leq j \leq n, 1 \leq i \leq m$$

Note that we assume in our calculations steps that the program is in standard form. However, any linear program may be transformed to standard form and it is therefore not a limiting factor.

Covering-packing dualities

A covering LP is a linear program of the form:

Minimize: $\mathbf{b}^T\mathbf{y}$,
Subject to: $A^T\mathbf{y} \geq \mathbf{c}$, $\mathbf{y} \geq 0$,

such that the matrix A and the vectors $\mathbf{b}$ and $\mathbf{c}$ are non-negative.

Covering-packing dualities

Covering problems	Packing problems
Minimum set cover	Maximum set packing
Minimum vertex cover	Maximum matching
Minimum edge cover	Maximum independent set

The dual of a covering LP is a packing LP, a linear program of the form:

Maximize: $\mathbf{c}^T\mathbf{x}$,
Subject to: $A\mathbf{x} \leq \mathbf{b}$, $\mathbf{x} \geq 0$,

such that the matrix A and the vectors $\mathbf{b}$ and $\mathbf{c}$ are non-negative.

Examples

Covering and packing LPs commonly arise as a linear programming relaxation of a combinatorial problem and are important in the study of approximation algorithms.[2] For example, the LP relaxations of the set packing problem, the independent set problem, and the matching problem are packing LPs. The LP relaxations of the set cover problem, the vertex cover problem, and the dominating set problem are also covering LPs.

Finding a fractional coloring of a graph is another example of a covering LP. In this case, there is one constraint for each vertex of the graph and one variable for each independent set of the graph.

Complementary slackness

It is possible to obtain an optimal solution to the dual when only an optimal solution to the primal is known using the complementary slackness theorem. The theorem states:

Suppose that $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$ is primal feasible and that $\mathbf{y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_m)$ is dual feasible. Let $(\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_m)$ denote the corresponding primal slack variables, and let $(\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n)$ denote the corresponding dual slack variables. Then $\mathbf{x}$ and $\mathbf{y}$ are optimal for their respective problems if and only if

- $\mathbf{x}_j\mathbf{z}_j = 0$, for $j = 1, 2, \ldots, n$, and
- $\mathbf{w}_i\mathbf{y}_i = 0$, for $i = 1, 2, \ldots, m$.

So if the i-th slack variable of the primal is not zero, then the i-th variable of the dual is equal zero. Likewise, if the j-th slack variable of the dual is not zero, then the j-th variable of the primal is equal to zero.

This necessary condition for optimality conveys a fairly simple economic principle. In standard form (when maximizing), if there is slack in a constrained primal resource (i.e., there are "leftovers"), then additional quantities of that resource must have no value. Likewise, if there is slack in the dual (shadow) price non-negativity constraint requirement, i.e., the price is not zero, then there must be scarce supplies (no "leftovers").

Theory

Existence of optimal solutions

Geometrically, the linear constraints define the feasible region, which is a convex polyhedron. A linear function is a convex function, which implies that every local minimum is a global minimum; similarly, a linear function is a concave function, which implies that every local maximum is a global maximum.

Optimal solution need not exist, for two reasons. First, if two constraints are inconsistent, then no feasible solution exists: For instance, the constraints $\mathbf{x} \geq 2$ and $\mathbf{x} \leq 1$ cannot be satisfied jointly; in this case, we say that the LP is *infeasible*. Second, when the polytope is unbounded in the direction of the gradient of the objective function (where the gradient of the objective function is the vector of the coefficients of the objective function), then no optimal value is

attained.

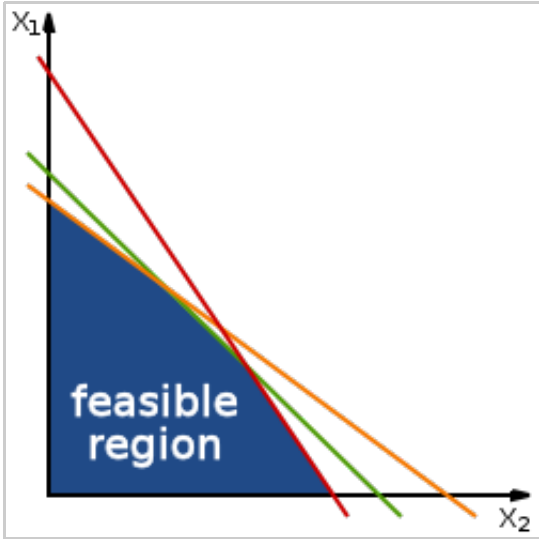
Optimal vertices (and rays) of polyhedra

Otherwise, if a feasible solution exists and if the (linear) objective function is bounded, then the optimum value is always attained on the boundary of optimal level-set, by the *maximum principle* for *convex functions* (alternatively, by the *minimum* principle for *concave functions*): Recall that linear functions are both convex and concave. However, some problems have distinct optimal solutions: For example, the problem of finding a feasible solution to a system of linear inequalities is a linear programming problem in which the objective function is the zero function (that is, the constant function taking the value zero everywhere): For this feasibility problem with the zero-function for its objective-function, if there are two distinct solutions, then every convex combination of the solutions is a solution.

The vertices of the polytope are also called *basic feasible solutions*. The reason for this choice of name is as follows. Let d denote the number of variables. Then the fundamental theorem of linear inequalities implies (for feasible problems) that for every vertex $\mathbf{x}^*$ of the LP feasible region, there exists a set of d (or fewer) inequality constraints from the LP such that, when we treat those d constraints as equalities, the unique solution is $\mathbf{x}^*$. Thereby we can study these vertices by means of looking at certain subsets of the set of all constraints (a discrete set), rather than the continuum of LP solutions. This principle underlies the simplex algorithm for solving linear programs.

Algorithms

Basis exchange algorithms



A series of linear constraints on two variables produces a region of possible values for those variables. Solvable problems will have a feasible region in the shape of a simple polygon.

Simplex algorithm of Dantzig

The simplex algorithm, developed by George Dantzig in 1947, solves LP problems by constructing a feasible solution at a vertex of the polytope and then walking along a path on the edges of the polytope to vertices with non-decreasing values of the objective function until an optimum is reached. In many practical problems, "stalling" occurs: Many pivots are made with no increase in the objective function.[3][4] In rare practical problems, the usual versions of the simplex algorithm may actually "cycle".[4] To avoid cycles, researchers developed new pivoting rules .[5][6][3][4][7][8]

In practice, the simplex algorithm is quite efficient and can be guaranteed to find the global optimum if certain precautions against *cycling* are taken. The simplex algorithm has been proved to solve "random" problems efficiently, i.e. in a cubic number of steps,[9] which is similar to its behavior on practical problems.[3][10]

However, the simplex algorithm has poor worst-case behavior: Klee and Minty constructed a

family of linear programming problems for which the simplex method takes a number of steps exponential in the problem size.[3][6][7] In fact, for some time it was not known whether the linear programming problem was solvable in polynomial time (complexity class P).

Criss-cross algorithm

Like the simplex algorithm of Dantzig, the criss-cross algorithm is a basis-exchange algorithm that pivots between bases. However, the criss-cross algorithm need not maintain feasibility, but can pivot rather from a feasible basis to an infeasible basis. The criss-cross algorithm does not have polynomial time-complexity for linear programming. Both algorithms visit all 2^D corners of a (perturbed) cube in dimension D, the Klee–Minty cube (after Victor Klee and George J. Minty), in the worst case.[8][11]

Interior point

Ellipsoid algorithm, following Khachiyan

This is the first worst-case polynomial-time algorithm for linear programming. To solve a problem which has n variables and can be encoded in L input bits, this algorithm uses $O(n^4L)$ pseudo-arithmetic operations on numbers with $O(L)$ digits. Khachiyan's algorithm and his long standing issue was resolved by Leonid Khachiyan in 1979 with the introduction of the ellipsoid method. The convergence analysis have (real-number) predecessors, notably the iterative methods developed by Naum Z. Shor and the approximation algorithms by Arkadi Nemirovski and D. Yudin.

Projective algorithm of Karmarkar

Khachiyan's algorithm was of landmark importance for establishing the polynomial-time solvability of linear programs. The algorithm was not a computational break-through, as the simplex method is more efficient for all but specially constructed families of linear programs.

However, Khachiyan's algorithm inspired new lines of research in linear programming. In 1984, N. Karmarkar proposed a projective method for linear programming. Karmarkar's algorithm improved on Khachiyan's worst-case polynomial bound (giving $O(n^{3.5}L)$). Karmarkar claimed that his algorithm was much faster in practical LP than the simplex method, a claim that created great interest in interior-point methods. Its projective geometry is interesting.[12]

Path-following algorithms

In contrast to the simplex algorithm, which finds an optimal solution by traversing the edges between vertices on a polyhedral set, interior-point methods move through the interior of the feasible region. Since then, many interior-point methods have been proposed and analyzed. Early successful implementations were based on *affine scaling* variants of the method. For both theoretical and practical purposes, barrier function or path-following methods have been the most popular since the 1990s.[13]

Comparison of interior-point methods versus simplex algorithms

The current opinion is that the efficiency of good implementations of simplex-based methods

and interior point methods are similar for routine applications of linear programming.[13] However, for specific types of LP problems, it may be that one type of solver is better than another (sometimes much better).

LP solvers are in widespread use for optimization of various problems in industry, such as optimization of flow in transportation networks.[14]

Open problems and recent work

There are several open problems in the theory of linear programming, the solution of which would represent fundamental breakthroughs in mathematics and potentially major advances in our ability to solve large-scale linear programs.

Unsolved problems in computer science

Does linear programming admit a strongly polynomial-time algorithm?	

- Does LP admit a strongly polynomial-time algorithm?
- Does LP admit a strongly polynomial algorithm to find a strictly complementary solution?
- Does LP admit a polynomial algorithm in the real number (unit cost) model of computation?

This closely related set of problems has been cited by Stephen Smale as among the 18 greatest unsolved problems of the 21st century. In Smale's words, the third version of the problem "is the main unsolved problem of linear programming theory." While algorithms exist to solve linear programming in weakly polynomial time, such as the ellipsoid methods and interior-point techniques, no algorithms have yet been found that allow strongly polynomial-time performance in the number of constraints and the number of variables. The development of such algorithms would be of great theoretical interest, and perhaps allow practical gains in solving large LPs as well.

Although the Hirsch conjecture was recently disproved for higher dimensions, it still leaves the following questions open.

- Are there pivot rules which lead to polynomial-time Simplex variants?
- Do all polytopal graphs have polynomially-bounded diameter?

These questions relate to the performance analysis and development of Simplex-like methods. The immense efficiency of the Simplex algorithm in practice despite its exponential-time theoretical performance hints that there may be variations of Simplex that run in polynomial or even strongly polynomial time. It would be of great practical and theoretical significance to know whether any such variants exist, particularly as an approach to deciding if LP can be solved in strongly polynomial time.

The Simplex algorithm and its variants fall in the family of edge-following algorithms, so named because they solve linear programming problems by moving from vertex to vertex along edges of a polytope. This means that their theoretical performance is limited by the maximum number of edges between any two vertices on the LP polytope. As a result, we are interested in knowing the maximum graph-theoretical diameter of polytopal graphs. It has been proved that all polytopes have subexponential diameter. The recent disprove of the Hirsch conjecture is the first step to prove whether any polytope has superpolynomial diameter. If any such polytopes exist, then no edge-following variant can run in polynomial

time. Questions about polytope diameter are of independent mathematical interest.

Simplex pivot methods preserve primal (or dual) feasibility. On the other hand, criss-cross pivot methods do not preserve (primal or dual) feasibility—they may visit primal feasible, dual feasible or primal-and-dual infeasible bases in any order. Pivot methods of this type have been studied since the 1970s. Essentially, these methods attempt to find the shortest pivot path on the arrangement polytope under the linear programming problem. In contrast to polytopal graphs, graphs of arrangement polytopes are known to have small diameter, allowing the possibility of strongly polynomial-time criss-cross pivot algorithm without resolving questions about the diameter of general polytopes.[8]

Integer unknowns

If the unknown variables are all required to be integers, then the problem is called an integer programming (IP) or **integer linear programming** (ILP) problem. In contrast to linear programming, which can be solved efficiently in the worst case, integer programming problems are in many practical situations (those with bounded variables) NP-hard. **0-1 integer programming** or **binary integer programming** (BIP) is the special case of integer programming where variables are required to be 0 or 1 (rather than arbitrary integers). This problem is also classified as NP-hard, and in fact the decision version was one of Karp's 21 NP-complete problems.

If only some of the unknown variables are required to be integers, then the problem is called a **mixed integer programming** (MIP) problem. These are generally also NP-hard.

There are however some important subclasses of IP and MIP problems that are efficiently solvable, most notably problems where the constraint matrix is totally unimodular and the right-hand sides of the constraints are integers.

Advanced algorithms for solving integer linear programs include:

- cutting-plane method
- branch and bound
- branch and cut
- branch and price
- if the problem has some extra structure, it may be possible to apply delayed column generation.

Such integer-programming algorithms are discussed by Padberg and in Beasley.

Integral linear programs

A linear program in real variables is said to be **integral** if it has at least one optimal solution which is integral. Likewise, a polyhedron $P = \{x \mid Ax \geq 0\}$ is said to be **integral** if for all bounded feasible objective functions *c*, the linear program $\{\max cx \mid x \in P\}$ has an optimum x^* with integer coordinates. As observed by Edmonds and Giles in 1977, one can equivalently say that a polyhedron is integral if for every bounded feasible integral objective function *c*, the optimal *value* of the linear progam $\{\max cx \mid x \in P\}$ is an integer.

Integral linear programs are of central importance in the polyhedral aspect of combinatorial

optimization since they provide an alternate characterization of a problem. Specifically, for any problem, the convex hull of the solutions is an integral polyhedron; if this polyhedron has a nice/compact description, then we can efficiently find the optimal feasible solution under any linear objective. Conversely, if we can prove that a linear programming relaxation is integral, then it is the desired description of the convex hull of feasible (integral) solutions.

Note that terminology is not consistent throughout the literature, so one should be careful to distinguish the following two concepts,

- in an *integer linear program,* described in the previous section, variables are forcibly constrained to be integers, and this problem is NP-hard in general,
- in an *integral linear program,* described in this section, variables are not constrained to be integers but rather one has proven somehow that the continuous problem always has an integral optimal value (assuming *c* is integral), and this optimal value may be found efficiently since all polynomial-size linear programs can be solved in polynomial time.

One common way of proving that a polyhedron is integral is to show that it is totally unimodular. There are other general methods including the integer decomposition property and total dual integrality. Other specific well-known integral LPs include the matching polytope, lattice polyhedra, submodular flow polyhedra, and the intersection of 2 generalized polymatroids/*g*-polymatroids --- e.g. see Schrijver 2003.

A bounded integral polyhedron is sometimes called a convex lattice polytope, particularly in two dimensions.

Solvers and scripting (programming) languages

Free open-source permissive licenses:

Name	License	Brief info
OpenOpt	BSD	Universal cross-platform numerical optimization framework, see its LP (http://openopt.org/LP) page and other problems (http://openopt.org/Problems) involved
pulp-or (http://pulp-or.googlecode.com)	BSD	Python module for modeling and solving linear programming problems
Pyomo (https://software.sandia.gov/pyomo)	BSD	Python module for formulating linear programming problems with abstract models

Free open-source copyleft (reciprocal) licenses:

Name	License	Brief info
LP_Solve (http://lpsolve.sourceforge.net/5.5/)	LGPL	User-friendly linear and integer programming solver. Also provides DLL for program integration.

Cassowary constraint solver	LGPL	an incremental constraint solving toolkit that efficiently solves systems of linear equalities and inequalities.
CVXOPT (http://abel.ee.ucla.edu/cvxopt/)	GPL	general purpose convex optimization solver written in Python, with a C API, and calls external routines (e.g. BLAS, LAPACK, FFTW) for numerical computations. Has its own solvers, but can also call glpk or MOSEK[15] if installed
glpk	GPL	GNU Linear Programming Kit, a free LP/MILP solver. Uses GNU MathProg modelling language.
Qoca	GPL	a library for incrementally solving systems of linear equations with various goal functions
CBC (http://www.coin-or.org/projects/Cbc.xml)	CPL	a MIP solver from COIN-OR
CLP	CPL	an LP solver from COIN-OR
R-Project	GPL	a programming language and software environment for statistical computing and graphics
CVX (http://cvxr.com/cvx)	GPL	MATLAB based modeling system for convex optimization, including linear programs; calls either SDPT3 or SeDuMi as a solver
CVXMOD (http://cvxmod.net/)	GPL	Python based modeling system, similar to CVX. It calls CVXOPT as its solver. It is still in alpha release, as of 2009
SDPT3 (http://www.math.nus.edu.sg/~mattohkc/sdpt3.html)	GPL	MATLAB based convex optimization solver
SeDuMi (http://sedumi.ie.lehigh.edu/)	GPL	MATLAB based convex optimization solver

MINTO (Mixed Integer Optimizer, an integer programming solver which uses branch and bound algorithm) has publicly available source code[16] but not open source.

Proprietary:

Name	Brief info
APMonitor	
AIMMS	
AMPL	A popular modeling language for large-scale linear, mixed integer and nonlinear optimisation with a free student version available.
Analytica (http://www.lumina.com/products/analytica-optimizer/)	Optimization modeling software that incorporates state-of-the-art algorithms for linear and nonlinear optimization. Supports LP, NLP, QP, continuous and integer optimization.
CPLEX	Popular solver with an API for several programming languages, and also has a modelling language and works with AIMMS, AMPL, GAMS, MPL, OpenOpt, OPL Development Studio, and TOMLAB
EXCEL Solver Function	
FortMP	
GAMS	
GIPALS	
Gurobi	Solver with parallel algorithms for large-scale linear programs, quadratic programs and mixed-integer programs. Free for academic use.
IMSL Numerical Libraries	Collections of math and statistical algorithms available in C/C++, Fortran, Java and C#/.NET. Optimization routines in the IMSL Libraries include unconstrained, linearly and nonlinearly constrained minimizations, and linear programming algorithms.
Lingo	
LPL	
LiPS (http://sourceforge.net/projects/lipside/) (freeware)	Linear Program Solver (LiPS) is intended for solving linear programming problems. Main features: easy to use graphical interface, sensitivity analysis, goal and mixed integer programming solver. LiPS supports MPS and simple LP format (like lpsolve).
	A general-purpose and matrix-oriented

MATLAB	programming-language for numerical computing. Linear programming in MATLAB requires the Optimization Toolbox in addition to the base MATLAB product; available routines include BINTPROG and LINPROG
Mathematica	A general-purpose programming-language for mathematics, including symbolic and numerical capabilities.
MOPS	
MOSEK	A solver for large scale optimization with API for several languages (C++,java,.net, Matlab and python).
NMath Stats	A general-purpose .NET statistical library containing a simplex solver.[17]
OptimJ	A Java-based modeling language for optimization with a free version available.[18][19]
SAS	
SCIP	A general-purpose constraint integer programming solver with an emphasis on MIP. Free for academic use and available in source code.
Solver Foundation	A .NET platform for modeling, scheduling, and optimization.
SoPlex (http://soplex.zib.de/)	The Sequential object-oriented simPlex: a general-purpose LP solver. Free for academic use and available in source code.
SuanShu	A Java-based math library that supports linear programming and other kinds of numerical optimization.[20]
TOMLAB	
VisSim	A visual block diagram language for simulation of dynamical systems.
Xpress	

See also

- Mathematical programming
- Nonlinear programming
- Convex programming
- Dynamic programming
- Simplex algorithm, used to solve LP problems
- Quadratic programming, a superset of linear programming
- Shadow price

- MPS file format
- nl file format
- MIP example, job shop problem
- Linear-fractional programming (LFP)
- Oriented matroid

Notes

1. **^** See his 1940 paper listed below
2. **^** Vazirani (2001, p. 112)
3. ^ ***a b c d*** Dantzig & Thapa (2003)
4. ^ ***a b c*** Padberg (1999)
5. **^** Bland (1977)
6. ^ ***a b*** Murty (1983)
7. ^ ***a b*** Papadimitriou (Steiglitz)
8. ^ ***a b c*** Fukuda & Terlaky (1997): Fukuda, Komei; Terlaky, Tamás (1997). "Criss-cross methods: A fresh view on pivot algorithms" (http://www.cas.mcmaster.ca/~terlaky/files/crisscross.ps) . *Mathematical Programming: Series B* (Amsterdam: North-Holland Publishing Co.) **79** (Papers from the 16th International Symposium on Mathematical Programming held in Lausanne, 1997): pp. 369–395. doi:10.1016/S0025-5610(97)00062-2 (http://dx.doi.org/10.1016%2FS0025-5610%2897%2900062-2) . MR1464775 (http://www.ams.org/mathscinet-getitem?mr=1464775) . http://www.cas.mcmaster.ca/~terlaky/files/crisscross.ps.
9. **^** Borgwardt (1987)
10. **^** Todd (2002)
11. **^** Roos (1990): Roos, C. (1990). "An exponential example for Terlaky's pivoting rule for the criss-cross simplex method". *Mathematical Programming*. Series A **46** (1): 79. doi:10.1007/BF01585729 (http://dx.doi.org/10.1007%2FBF01585729) . MR1045573 (http://www.ams.org/mathscinet-getitem?mr=1045573) .
12. **^** 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 (http://dx.doi.org/10.1007%2FBF03025891) . ISSN 0343-6993 (http://www.worldcat.org/issn/0343-6993) . MR**883185** "'883185'" (http://www.ams.org/mathscinet-getitem?mr=) .
13. ^ ***a b*** Gondzio & Terlaky (1996)
14. **^** For solving network-flow problems in transportation networks, specialized implementations of the simplex algorithm can dramatically improve its efficiency. Dantzig & Thapa (2003)
15. **^** http://www.mosek.com/
16. **^** http://coral.ie.lehigh.edu/~minto/download.html
17. **^** Linear programming page at CenterSpace Software (http://www.centerspace.net/landing.php?id=lp)
18. **^** http://www.in-ter-trans.eu/resources/Zesch_Hellingrath_2010_Integrated+Production-Distribution+Planning.pdf OptimJ used in an optimization model for mixed-model assembly lines, University of Münster
19. **^** http://www.aaai.org/ocs/index.php/AAAI/AAAI10/paper/viewFile/1769/2076 OptimJ used in an Approximate Subgame-Perfect Equilibrium Computation Technique for Repeated Games
20. **^** http://www.numericalmethod.com

References

- L.V. Kantorovich: *A new method of solving some classes of extremal problems*,

Doklady Akad Sci USSR, 28, 1940, 211-214.
- G.B Dantzig: *Maximization of a linear function of variables subject to linear inequalities*, 1947. Published pp. 339–347 in T.C. Koopmans (ed.):*Activity Analysis of Production and Allocation*, New York-London 1951 (Wiley & Chapman-Hall)
- J. E. Beasley, editor. *Advances in Linear and Integer Programming*. Oxford Science, 1996. (Collection of surveys)
- R. G. Bland, New finite pivoting rules for the simplex method, *Math. Oper. Res.* 2 (1977) 103–107.
- Karl-Heinz Borgwardt, *The Simplex Algorithm: A Probabilistic Analysis*, Algorithms and Combinatorics, Volume 1, Springer-Verlag, 1987. (Average behavior on random problems)
- Richard W. Cottle, ed. *The Basic George B. Dantzig*. Stanford Business Books, Stanford University Press, Stanford, California, 2003. (Selected papers by George B. Dantzig)
- George B. Dantzig and Mukund N. Thapa. 1997. *Linear programming 1: Introduction*. Springer-Verlag.
- George B. Dantzig and Mukund N. Thapa. 2003. *Linear Programming 2: Theory and Extensions*. Springer-Verlag. (Comprehensive, covering e.g. pivoting and interior-point algorithms, large-scale problems, decomposition following Dantzig-Wolfe and Benders, and introducing stochastic programming.)
- Edmonds, J. and Giles, R., "A min-max relation for submodular functions on graphs," Ann. Discrete Math., v1, pp. 185–204, 1977
- Fukuda, Komei; Terlaky, Tamás (1997). "Criss-cross methods: A fresh view on pivot algorithms" (http://www.cas.mcmaster.ca/~terlaky/files/crisscross.ps) . *Mathematical Programming: Series B* (Amsterdam: North-Holland Publishing Co.) **79** (Papers from the 16th International Symposium on Mathematical Programming held in Lausanne, 1997): pp. 369–395. doi:10.1016/S0025-5610(97)00062-2 (http://dx.doi.org/10.1016%2FS0025-5610%2897%2900062-2) . MR1464775 (http://www.ams.org/mathscinet-getitem?mr=1464775) . http://www.cas.mcmaster.ca/~terlaky/files/crisscross.ps.
- Gondzio, Jacek; Terlaky, Tamás (1996). "3 A computational view of interior point methods" (http://www.maths.ed.ac.uk/~gondzio/CV/oxford.ps) . In J. E. Beasley. *Advances in linear and integer programming*. Oxford Lecture Series in Mathematics and its Applications. **4**. New York: Oxford University Press. pp. 103–144. MR1438311 (http://www.ams.org/mathscinet-getitem?mr=1438311) . Postscript file at website of Gondzio (http://www.maths.ed.ac.uk/~gondzio/CV/oxford.ps) and at McMaster University website of Terlaky (http://www.cas.mcmaster.ca/~terlaky/files/dut-twi-94-73.ps.gz) . http://www.maths.ed.ac.uk/~gondzio/CV/oxford.ps.
- Murty, Katta G. (1983). *Linear programming*. New York: John Wiley & Sons, Inc.. pp. xix+482. ISBN 0-471-09725-X. MR720547 (http://www.ams.org/mathscinet-getitem?mr=720547) . (comprehensive reference to classical approaches).
- Evar D. Nering and Albert W. Tucker, 1993, *Linear Programs and Related Problems*, Academic Press. (elementary)
- M. Padberg, *Linear Optimization and Extensions*, Second Edition, Springer-Verlag, 1999. (carefully written account of primal and dual simplex algorithms and projective algorithms, with an introduction to integer linear programming --- featuring the traveling salesman problem for Odysseus.)
- Christos H. Papadimitriou and Kenneth Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*, Corrected republication with a new preface, Dover. (computer science)

- Michael J. Todd (February 2002). "The many facets of linear programming". *Mathematical Programming* **91** (3). (Invited survey, from the International Symposium on Mathematical Programming.)
- Vazirani, Vijay V. (2001). *Approximation Algorithms*. Springer-Verlag. ISBN 3-540-65367-8. (Computer science)

Further reading

A reader may consider beginning with Nering and Tucker, with the first volume of Dantzig and Thapa, or with Williams.

- Dmitris Alevras and Manfred W. Padberg, *Linear Optimization and Extensions: Problems and Extensions*, Universitext, Springer-Verlag, 2001. (Problems from Padberg with solutions.)
- Mark de Berg, Marc van Kreveld, Mark Overmars, and Otfried Schwarzkopf (2000). *Computational Geometry* (2nd revised ed.). Springer-Verlag. ISBN 3-540-65620-0. Chapter 4: Linear Programming: pp. 63–94. Describes a randomized half-plane intersection algorithm for linear programming.
- Michael R. Garey and David S. Johnson (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman. ISBN 0-7167-1045-5. A6: MP1: INTEGER PROGRAMMING, pg.245. (computer science, complexity theory)
- Bernd Gärtner, Jiří Matoušek (2006). *Understanding and Using Linear Programming*, Berlin: Springer. ISBN 3-540-30697-8 (elementary introduction for mathematicians and computer scientists)
- Cornelis Roos, Tamás Terlaky, Jean-Philippe Vial, *Interior Point Methods for Linear Optimization*, Second Edition, Springer-Verlag, 2006. (Graduate level)
- Alexander Schrijver (2003). *Combinatorial optimization: polyhedra and efficiency*. Springer.
- Alexander Schrijver, *Theory of Linear and Integer Programming*. John Wiley & sons, 1998, ISBN 0-471-98232-6 (mathematical)
- 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. (An on-line second edition was formerly available. Vanderbei's site still contains extensive materials.)
- H. P. Williams, *Model Building in Mathematical Programming*, Third revised Edition, 1990. (Modeling)
- Stephen J. Wright, 1997, *Primal-Dual Interior-Point Methods*, SIAM. (Graduate level)
- Yinyu Ye, 1997, *Interior Point Algorithms: Theory and Analysis*, Wiley. (Advanced graduate-level)
- Ziegler, Günter M., Chapters 1–3 and 6–7 in *Lectures on Polytopes*, Springer-Verlag, New York, 1994. (Geometry)

External links

- Guidance on Formulating LP problems (http://people.brunel.ac.uk/~mastjjb/jeb/or/lp.html)
- Mathematical Programming Glossary (http://glossary.computing.society.informs.org/)
- The linear programming FAQ (http://wiki.mcs.anl.gov/NEOS/index.php/Linear_Programming_FAQ)

- 2009 Linear Programming Software Survey (http://www.lionhrtpub.com/orms/surveys/LP/LP-survey.html) - *OR/MS Today*
- George Dantzig (http://www.stanford.edu/group/SOL/dantzig.html)

Categories: Linear programming | Convex optimization | Operations research | Geometric algorithms | P-complete problems | Mathematical and quantitative methods (economics)
