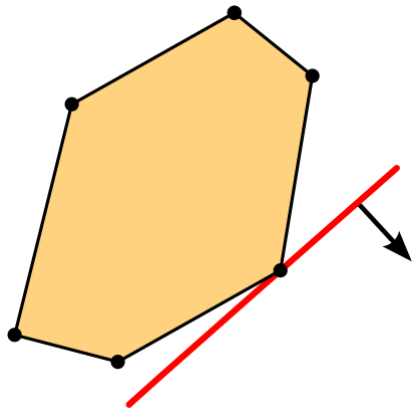
WIKIPEDIA

Linear programming

Linear programming (**LP**, also called **linear optimization**) is a method to achieve the best outcome (such as maximum profit or lowest cost) in a mathematical model whose requirements are represented by linear relationships. Linear programming is a special case of mathematical programming (also known as 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 polytope, 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 (linear) function defined on this polyhedron. A linear programming algorithm finds a point in the polytope where this function has the smallest (or largest) value if such a point exists.



A pictorial representation of a simple linear program with two variables and six inequalities. The set of feasible solutions is depicted in yellow and forms a polygon, a 2-dimensional polytope. The linear cost function is represented by the red line and the arrow: The red line is a level set of the cost function, and the arrow indicates the direction in which we are optimizing.

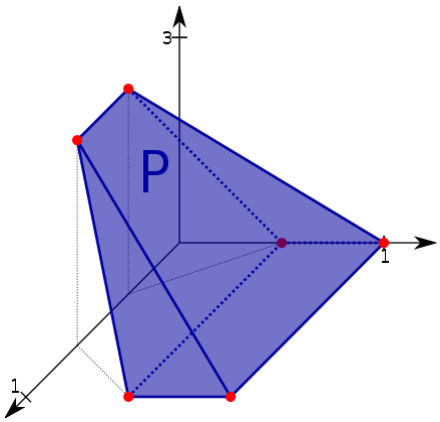
Linear programs are problems that can be expressed in canonical form as

$$\begin{array}{ll} \textbf{Find a vector} & \mathbf{x} \\ \textbf{that maximizes} & \mathbf{c}^T\mathbf{x} \\ \textbf{subject to} & A\mathbf{x} \leq \mathbf{b} \\ \textbf{and} & \mathbf{x} \geq \mathbf{0}. \end{array}$$

Here the components of $\mathbf{x}$ are the variables to be determined, $\mathbf{c}$ and $\mathbf{b}$ are given vectors (with $\mathbf{c}^T$ indicating that the coefficients of $\mathbf{c}$ are used as a single-row matrix for the purpose of forming the matrix product), and A is a given matrix. The function whose value is to be maximized or minimized ($\mathbf{x} \mapsto \mathbf{c}^T\mathbf{x}$ in this case) is called the objective function. The inequalities $A\mathbf{x} \leq \mathbf{b}$ and $\mathbf{x} \geq \mathbf{0}$ 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 they have the same dimensions. If every entry in the first is less-than or equal-to the corresponding entry in the second, then it can be said that the first vector is less-than or equal-to the second vector.

Linear programming can be applied to various fields of study. It is widely used in mathematics, and to a lesser extent in business, economics, and for some engineering problems. Industries that use linear programming models include transportation, energy, telecommunications, and manufacturing. It has proven useful in modeling diverse types of problems in planning, routing, scheduling, assignment, and design.

Contents

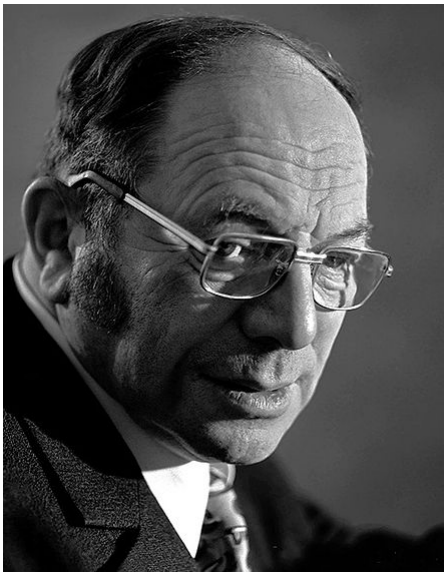


A closed feasible region of a problem with three variables is a convex polyhedron. The surfaces giving a fixed value of the objective function are planes (not shown). The linear programming problem is to find a point on the polyhedron that is on the plane with the highest possible value.

History

The problem of solving a system of linear inequalities dates back at least as far as Fourier, who in 1827 published a method for solving them,[1] and after whom the method of Fourier–Motzkin elimination is named.

In 1939 a linear programming formulation of a problem that is equivalent to the general linear programming problem was given by the Soviet mathematician and economist Leonid Kantorovich, who also proposed a method for solving it.[2] It is a way he developed, during World War II, to plan expenditures and returns in order to reduce costs of the army and to increase losses imposed on the enemy. Kantorovich's work was initially neglected in the USSR.[3] About the same time as Kantorovich, the Dutch-American economist T. C. Koopmans formulated classical economic problems as linear programs. Kantorovich and Koopmans later shared the 1975 Nobel prize in economics.[1] In 1941, Frank Lauren Hitchcock also formulated transportation problems as linear programs and gave a solution very similar to the later simplex method.[2] Hitchcock had died in 1957 and the Nobel prize is not awarded posthumously.


Leonid Kantorovich

During 1946–1947, George B. Dantzig independently developed general linear programming formulation to use for planning problems in the US Air Force.[4] In 1947, Dantzig also invented the simplex method that for the first time efficiently tackled the linear programming problem in most cases.[4] When Dantzig arranged a meeting with John von Neumann to discuss his simplex method, Neumann immediately conjectured the theory of duality by realizing that the problem he had been working in game theory was equivalent.[4] Dantzig provided formal proof in an unpublished report "A Theorem on Linear Inequalities" on January 5, 1948.[3] Dantzig's work was made available to public in 1951. In the post-war years, many industries applied it in their daily planning.


John von Neumann

Dantzig's original example was to find the best assignment of 70 people to 70 jobs. 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 observable 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 solutions that must be checked.

The linear programming problem was first shown to be solvable in polynomial time by Leonid Khachiyan in 1979,[5] 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.[6]

Uses

Linear programming is a widely used field of optimization for several reasons. Many practical problems in operations research can be expressed as linear programming problems.[3] 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 was heavily used in the early formation of microeconomics and it is currently utilized in 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 and minimize costs with limited resources. Google uses linear programming to stabilize YouTube videos.[7] 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 three parts:

- A **linear function to be maximized**

  e.g. $f(x_1, x_2) = c_1 x_1 + c_2 x_2$

- **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.

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

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

$$\max\{\, \mathbf{c}^{\mathrm{T}} \mathbf{x} \mid \mathbf{x} \in \mathbb{R}^n \wedge A\mathbf{x} \leq \mathbf{b} \wedge \mathbf{x} \geq 0 \,\}$$

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 pesticide, P kilograms. Every square kilometer of wheat requires F_1 kilograms of fertilizer and P_1 kilograms of pesticide, while every square kilometer of barley requires F_2 kilograms of fertilizer and P_2 kilograms of pesticide. Let S_1 be the selling price of wheat per square kilometer, and S_2 be the selling

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:	$S_1 \cdot x_1 + S_2 \cdot x_2$	(maximize the revenue (the total wheat sales plus the total barley sales) – revenue is the "objective function")
Subject to:	$x_1 + x_2 \leq L$	(limit on total area)
	$F_1 \cdot x_1 + F_2 \cdot x_2 \leq F$	(limit on fertilizer)
	$P_1 \cdot x_1 + P_2 \cdot x_2 \leq P$	(limit on pesticide)
	$x_1 \geq 0, x_2 \geq 0$	(cannot plant a negative area).

In matrix form this becomes:

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

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

Augmented form (slack form)

Linear programming problems can be converted into an *augmented form* in order to apply the common form of the simplex algorithm. This form introduces non-negative *slack variables* to replace inequalities with equalities in the constraints. The problems can then be written in the following block matrix form:

Maximize z:

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

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

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

Example

The example above is converted into the following augmented form:

Maximize:	$S_1 \cdot x_1 + S_2 \cdot x_2$	(objective function)
subject to:	$x_1 + x_2 + x_3 = L$	(augmented constraint)
	$F_1 \cdot x_1 + F_2 \cdot x_2 + x_4 = F$	(augmented constraint)
	$P_1 \cdot x_1 + P_2 \cdot x_2 + x_5 = P$	(augmented constraint)
	$x_1, x_2, x_3, x_4, x_5 \geq 0.$	

where x_3, x_4, x_5 are (non-negative) slack variables, representing in this example the unused area, the amount of unused fertilizer, and the amount of unused pesticide.

In matrix form this becomes:

Maximize z:

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

Duality

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

with the corresponding **symmetric** dual problem,

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

An alternative primal formulation is:

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

with the corresponding **asymmetric** dual problem,

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

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, $\mathbf{y}^*$, and $\mathbf{c}^{\mathsf{T}}\mathbf{x}^* = \mathbf{b}^{\mathsf{T}}\mathbf{y}^*$.

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 dual linear program for details and several more examples.

Variations

Covering/packing dualities

A covering LP is a linear program of the form:

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

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

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

Maximize: $\mathbf{c}^{\mathsf{T}}\mathbf{x}$,
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.[8] 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 to 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.

An optimal solution need not exist, for two reasons. First, if the 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 because it is always possible to do better than any finite value of the objective function.

Optimal vertices (and rays) of polyhedra

Otherwise, if a feasible solution exists and if the constraint set is bounded, then the optimum value is always attained on the boundary of the constraint set, by the *maximum principle* for *convex functions* (alternatively, by the *minimum* principle for *concave functions*) since 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

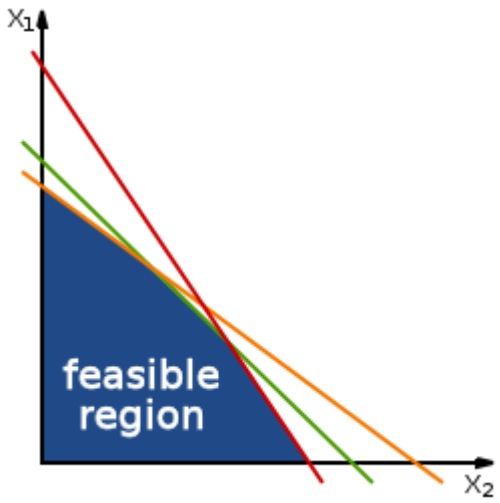
Basis exchange algorithms

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 for sure. In many practical problems, "stalling" occurs: many pivots are made with no increase in the objective function.[9][10] In rare practical problems, the usual versions of the simplex algorithm may actually "cycle".[10] To avoid cycles, researchers developed new pivoting rules.[11][12][9][10][13][14]

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,[15] which is similar to its behavior on practical problems.[9][16]

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.[9][12][13] In fact, for some time it was not known whether the linear programming problem was solvable in polynomial time, i.e. of complexity class P.



In a linear programming problem, a series of linear constraints produces a convex feasible region of possible values for those variables. In the two-variable case this region is in the shape of a convex simple polygon.

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, in the worst case.[14][17]

Interior point

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.

Ellipsoid algorithm, following Khachiyan

This is the first worst-case polynomial-time algorithm ever found for linear programming. To solve a problem which has n variables and can be encoded in L input bits, this algorithm runs in $O(n^6 L)$ time.[5] Leonid Khachiyan solved this long-standing complexity issue in 1979 with the introduction of the ellipsoid method. The convergence analysis has (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[6] improved on Khachiyan's[5] 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.[18] Since Karmarkar's discovery, many interior-point methods have been proposed and analyzed.

Vaidya's 87 algorithm

In 1987, Vaidya proposed an algorithm that runs in $O(n^3)$ time.[19]

Vaidya's 89 algorithm

In 1989, Vaidya developed an algorithm that runs in $O(n^{2.5})$ time.[20] Formally speaking, the algorithm takes $O((n+d)^{1.5}nL)$ arithmetic operations in the worst case, where d is the number of constraints, n is the number of variables, and L is the number of bits.

Input sparsity time algorithms

In 2015, Lee and Sidford showed that, it can be solved in $\tilde{O}((nnz(A)+d^2)\sqrt{d}L)$ time,[21] where $nnz(A)$ represents the number of non-zero elements, and it remains taking $O(n^{2.5}L)$ in the worst case.

Current matrix multiplication time algorithm

In 2019, Cohen, Lee and Song improved the running time to $\tilde{O}((n^\omega + n^{2.5-\alpha/2} + n^{2+1/6})L)$ time, ω is the exponent of matrix multiplication and α is the dual exponent of matrix multiplication.[22] α is (roughly) defined to be the largest number such that one can multiply an $n \times n$ matrix by a $n \times n^\alpha$ matrix in $O(n^2)$ time. In a followup work by Lee, Song and Zhang, they reproduce the same result via a different method.[23] These two algorithms remain $\tilde{O}(n^{2+1/6}L)$ when $\omega = 2$ and $\alpha = 1$. The result due to Jiang, Song, Weinstein and Zhang improved $\tilde{O}(n^{2+1/6}L)$ to $\tilde{O}(n^{2+1/18}L)$.[24]

Comparison of interior-point methods and simplex algorithms

The current opinion is that the efficiencies of good implementations of simplex-based methods and interior point methods are similar for routine applications of linear programming. However, for specific types of LP problems, it may be that one type of solver is better than another (sometimes much better), and that the structure of the solutions generated by interior point methods versus simplex-based methods are significantly different with the support set of active variables being typically smaller for the latter one.[25]

Open problems and recent work

? **Unsolved problem in computer science**:
Does linear programming admit a strongly polynomial-time algorithm?
(more unsolved problems in computer science)

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.

- Does LP admit a strongly polynomial-time algorithm?
- Does LP admit a strongly polynomial-time algorithm to find a strictly complementary solution?
- Does LP admit a polynomial-time 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 disproof 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.[14]

Integer unknowns

If all of the unknown variables are 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 because they are even more general than ILP programs.

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 or – more general – where the system has the total dual integrality (TDI) property.

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 the polyhedron P is integral if for every bounded feasible integral objective function c, the optimal *value* of the linear program $\{\max cx \mid x \in P\}$ 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.

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 two generalized polymatroids/g-polymatroids – e.g. see Schrijver 2003.

Solvers and scripting (programming) languages

Permissive licenses:

Name	License	Brief info
Gekko	MIT License	Open-source library for solving large-scale LP, QP, QCQP, NLP, and MIP optimization
GLOP	Apache v2	Google's open-source linear programming solver
Pyomo	BSD	An open-source modeling language for large-scale linear, mixed integer and nonlinear optimization
SuanShu	Apache v2	an open-source suite of optimization algorithms to solve LP, QP, SOCP, SDP, SQP in Java

Copyleft (reciprocal) licenses:

Name	License	Brief info
Cassowary constraint solver	LGPL	an incremental constraint solving toolkit that efficiently solves systems of linear equalities and inequalities
CLP	CPL	an LP solver from COIN-OR
glpk	GPL	GNU Linear Programming Kit, an LP/MILP solver with a native C API and numerous (15) third-party wrappers for other languages. Specialist support for flow networks. Bundles the AMPL-like GNU MathProg modelling language and translator.
Qoca	GPL	a library for incrementally solving systems of linear equations with various goal functions
R-Project	GPL	a programming language and software environment for statistical computing and graphics

MINTO (Mixed Integer Optimizer, an integer programming solver which uses branch and bound algorithm) has publicly available source code[26] but is not open source.

Proprietary licenses:

Name	Brief info
AIMMS	A modeling language that allows to model linear, mixed integer, and nonlinear optimization models. It also offers a tool for constraint programming. Algorithm, in the forms of heuristics or exact methods, such as Branch-and-Cut or Column Generation, can also be implemented. The tool calls an appropriate solver such as CPLEX or similar, to solve the optimization problem at hand. Academic licenses are free of charge.
AMPL	A popular modeling language for large-scale linear, mixed integer and nonlinear optimisation with a free student limited version available (500 variables and 500 constraints).
APMonitor	API to MATLAB and Python. Solve example Linear Programming (LP) problems through MATLAB, Python, or a web-interface.
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. Free for academic use.
Excel Solver Function	A nonlinear solver adjusted to spreadsheets in which function evaluations are based on the recalculating cells. Basic version available as a standard add-on for Excel.
FortMP	
GAMS	
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.
LINDO	Solver with an API for large scale optimization of linear, integer, quadratic, conic and general nonlinear programs with stochastic programming extensions. It offers a global optimization procedure for finding guaranteed globally optimal solution to general nonlinear programs with continuous and discrete variables. It also has a statistical sampling API to integrate Monte-Carlo simulations into an optimization framework. It has an algebraic modeling language (LINGO) and allows modeling within a spreadsheet (What'sBest).
Maple	A general-purpose programming-language for symbolic and numerical computing.
MATLAB	A general-purpose and matrix-oriented programming-language for numerical computing. Linear programming in MATLAB requires the Optimization Toolbox in addition to the base MATLAB product; available routines include INTLINPROG and LINPROG
Mathcad	A WYSIWYG math editor. It has functions for solving both linear and nonlinear optimization problems.
Mathematica	A general-purpose programming-language for mathematics, including symbolic and numerical capabilities.
MOSEK	A solver for large scale optimization with API for several languages (C++,java,.net, Matlab and python).
NAG Numerical Library	A collection of mathematical and statistical routines developed by the Numerical Algorithms Group for multiple programming languages (C, C++, Fortran, Visual Basic, Java and C#) and packages (MATLAB, Excel, R, LabVIEW). The Optimization chapter of the NAG Library includes routines for linear programming problems with both sparse and non-sparse linear constraint matrices, together with routines for the optimization of quadratic, nonlinear, sums of squares of linear or nonlinear functions with nonlinear, bounded or no constraints. The NAG Library has routines for both local and global optimization, and for continuous or integer problems.
OptimJ	A Java-based modeling language for optimization with a free version available.[27][28]
SAS/OR	A suite of solvers for Linear, Integer, Nonlinear, Derivative-Free, Network, Combinatorial and Constraint Optimization; the Algebraic modeling language OPTMODEL; and a variety of vertical solutions aimed at specific problems/markets, all of which are fully integrated with the SAS System.
SCIP	A general-purpose constraint integer programming solver with an emphasis on MIP. Compatible with Zimpl modelling language. Free for academic use and available in source code.
XPRESS	Solver for large-scale linear programs, quadratic programs, general nonlinear and mixed-integer programs. Has API for several programming languages, also has a modelling language Mosel and works with AMPL, GAMS. Free for academic use.

VisSim	A visual block diagram language for simulation of dynamical systems.

See also

- Convex programming
- Dynamic programming
- Expected shortfall § Optimization of expected shortfall
- Input–output model
- Job shop scheduling
- Least-squares spectral analysis
- Linear algebra
- Linear production game
- Linear-fractional programming (LFP)
- LP-type problem
- Mathematical programming
- Nonlinear programming
- Oriented matroid
- Quadratic programming, a superset of linear programming
- Semidefinite programming
- Shadow price
- Simplex algorithm, used to solve LP problems

Notes

1. Gerard Sierksma; Yori Zwols (2015). *Linear and Integer Optimization: Theory and Practice* (3rd ed.). CRC Press. p. 1. ISBN 978-1498710169.
2. Alexander Schrijver (1998). *Theory of Linear and Integer Programming*. John Wiley & Sons. pp. 221–222. ISBN 978-0-471-98232-6.
3. George B. Dantzig (April 1982). "Reminiscences about the origins of linear programming" (http://www.dtic.mil/cgi-bin/GetTRDoc?Location=U2&doc=GetTRDoc.pdf&AD=ADA112060). *Operations Research Letters*. **1** (2): 43–48. doi:10.1016/0167-6377(82)90043-8 (https://doi.org/10.1016%2F0167-6377%2882%2990043-8).
4. Dantzig, George B.; Thapa, Mukund Narain (1997). *Linear programming*. New York: Springer. p. xxvii. ISBN 0387948333. OCLC 35318475 (https://www.worldcat.org/oclc/35318475).
5. Leonid Khachiyan (1979). "A Polynomial Algorithm for Linear Programming". *Doklady Akademii Nauk SSSR*. **224** (5): 1093–1096.
6. Narendra Karmarkar (1984). "A New Polynomial-Time Algorithm for Linear Programming". *Combinatorica*. **4** (4): 373–395. doi:10.1007/BF02579150 (https://doi.org/10.1007%2FBF02579150). S2CID 7257867 (https://api.semanticscholar.org/CorpusID:7257867).
7. (PDF) https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/37041.pdf (https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/37041.pdf). {{cite web}}: Missing or empty |title= (help)
8. Vazirani (2001, p. 112)
9. Dantzig & Thapa (2003)
10. Padberg (1999)
11. Bland (1977)
12. Murty (1983)
13. Papadimitriou & Steiglitz

14. Fukuda, Komei; Terlaky, Tamás (1997). Thomas M. Liebling; Dominique de Werra (eds.). "Criss-cross methods: A fresh view on pivot algorithms". *Mathematical Programming, Series B*. **79** (1–3): 369–395. CiteSeerX 10.1.1.36.9373 (https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.36.9373). doi:10.1007/BF02614325 (https://doi.org/10.1007%2FBF02614325). MR 1464775 (https://www.ams.org/mathscinet-getitem?mr=1464775). S2CID 2794181 (https://api.semanticscholar.org/CorpusID:2794181).
15. Borgwardt (1987)
16. Todd (2002)
17. Roos, C. (1990). "An exponential example for Terlaky's pivoting rule for the criss-cross simplex method". *Mathematical Programming*. Series A. **46** (1): 79–84. doi:10.1007/BF01585729 (https://doi.org/10.1007%2FBF01585729). MR 1045573 (https://www.ams.org/mathscinet-getitem?mr=1045573). S2CID 33463483 (https://api.semanticscholar.org/CorpusID:33463483).
18. Strang, Gilbert (1 June 1987). "Karmarkar's algorithm and its place in applied mathematics". *The Mathematical Intelligencer*. **9** (2): 4–10. doi:10.1007/BF03025891 (https://doi.org/10.1007%2FBF03025891). ISSN 0343-6993 (https://www.worldcat.org/issn/0343-6993). MR 0883185 (https://www.ams.org/mathscinet-getitem?mr=0883185). S2CID 123541868 (https://api.semanticscholar.org/CorpusID:123541868).
19. Vaidya, Pravin M. (1987). *An algorithm for linear programming which requires* $O(((m+n)n^2+(m+n)^{1.5}n)L)$ *arithmetic operations*. 28th Annual IEEE Symposium on Foundations of Computer Science. FOCS.
20. Vaidya, Pravin M. (1989). *Speeding-up linear programming using fast matrix multiplication*. 30th Annual Symposium on Foundations of Computer Science. FOCS. doi:10.1109/SFCS.1989.63499 (https://doi.org/10.1109%2FSFCS.1989.63499).
21. Lee, Yin-Tat; Sidford, Aaron (2015). *Efficient inverse maintenance and faster algorithms for linear programming*. FOCS '15 Foundations of Computer Science. arXiv:1503.01752 (https://arxiv.org/abs/1503.01752).
22. Cohen, Michael B.; Lee, Yin-Tat; Song, Zhao (2018). *Solving Linear Programs in the Current Matrix Multiplication Time*. 51st Annual ACM Symposium on the Theory of Computing. STOC'19. arXiv:1810.07896 (https://arxiv.org/abs/1810.07896).
23. Lee, Yin-Tat; Song, Zhao; Zhang, Qiuyi (2019). *Solving Empirical Risk Minimization in the Current Matrix Multiplication Time*. Conference on Learning Theory. COLT'19. arXiv:1905.04447 (https://arxiv.org/abs/1905.04447).
24. Jiang, Shunhua; Song, Zhao; Weinstein, Omri; Zhang, Hengjie (2020). *Faster Dynamic Matrix Inverse for Faster LPs*. arXiv:2004.07470 (https://arxiv.org/abs/2004.07470).
25. Illés, Tibor; Terlaky, Tamás (2002). "Pivot versus interior point methods: Pros and cons" (https://strathprints.strath.ac.uk/9200/). *European Journal of Operational Research*. **140** (2): 170. CiteSeerX 10.1.1.646.3539 (https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.646.3539). doi:10.1016/S0377-2217(02)00061-9 (https://doi.org/10.1016%2FS0377-2217%2802%2900061-9).
26. "COR@L – Computational Optimization Research At Lehigh" (http://coral.ie.lehigh.edu/~minto/download.html). *lehigh.edu*.
27. 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
28. 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

References

- Kantorovich, L. V. (1940). "Об одном эффективном методе решения некоторых классов экстремальных проблем" [A new method of solving some classes of extremal problems]. *Doklady Akad Sci SSSR*. **28**: 211–214.
- F. L. Hitchcock: *The distribution of a product from several sources to numerous localities (https://onlinelibrary.wiley.com/doi/abs/10.1002/sapm1941201224)*, Journal of Mathematics and Physics, 20, 1941, 224–230.
- G.B Dantzig: *Maximization of a linear function of variables subject to linear inequalities (https://books.google.com/books?hl=en&lr=&id=ZpYca36h464C&oi=fnd&pg=PA24&dq=%22Maximization+of+a+linear+function+of+variables+subject+to+linear+inequalities%22&ots=0viWRKQVGk&sig=25NCv3tDYjTLYxCxn9deMWBn8VE)*, 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)
- Bland, Robert G. (1977). "New Finite Pivoting Rules for the Simplex Method". *Mathematics of Operations Research*. **2** (2): 103–107. doi:10.1287/moor.2.2.103 (https://doi.org/10.1287%2Fmoor.2.2.103). JSTOR 3689647 (https://www.jstor.org/stable/3689647).
- 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, Jack; Giles, Rick (1977). "A Min-Max Relation for Submodular Functions on Graphs". *Studies in Integer Programming*. Annals of Discrete Mathematics. Vol. 1. pp. 185–204. doi:10.1016/S0167-5060(08)70734-9 (https://doi.org/10.1016%2FS0167-5060%2808%2970734-9). ISBN 978-0-7204-0765-5.
- Fukuda, Komei; Terlaky, Tamás (1997). Thomas M. Liebling; Dominique de Werra (eds.). "Criss-cross methods: A fresh view on pivot algorithms". *Mathematical Programming, Series B*. **79** (1–3): 369–395. CiteSeerX 10.1.1.36.9373 (https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.36.9373). doi:10.1007/BF02614325 (https://doi.org/10.1007%2FBF02614325). MR 1464775 (https://www.ams.org/mathscinet-getitem?mr=1464775). S2CID 2794181 (https://api.semanticscholar.org/CorpusID:2794181).
- 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 (ed.). *Advances in linear and integer programming*. Oxford Lecture Series in Mathematics and its Applications. Vol. 4. New York: Oxford University Press. pp. 103–144. MR 1438311 (https://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).
- Murty, Katta G. (1983). *Linear programming*. New York: John Wiley & Sons, Inc. pp. xix+482. ISBN 978-0-471-09725-9. MR 0720547 (https://www.ams.org/mathscinet-getitem?mr=0720547). (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): 417–436. doi:10.1007/s101070100261 (https://doi.org/10.1007%2Fs101070100261). S2CID 6464735 (https://api.semanticscholar.org/CorpusID:6464735). (Invited survey, from the International Symposium on Mathematical Programming.)
- Vanderbei, Robert J. (2001). *Linear Programming: Foundations and Extensions*. Springer Verlag.
- Vazirani, Vijay V. (2001). *Approximation Algorithms*. Springer-Verlag. ISBN 978-3-540-65367-7. (Computer science)

Further reading

- Dmitris Alevras and Manfred W. Padberg, *Linear Optimization and Extensions: Problems and Solutions (https://books.google.com/books?id=RAUyB8NDHJwC&printsec=frontcover#v=onepage&q&f=false)*, Universitext, Springer-Verlag, 2001. (Problems from Padberg with solutions.)
- Mark de Berg, Marc van Kreveld, Mark Overmars, and Otfried Schwarzkopf (2000). *Computational Geometry* (https://archive.org/details/computationalgeo00berg) (2nd revised ed.). Springer-Verlag. ISBN 978-3-540-65620-3. 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 978-0-7167-1045-5. A6: MP1: INTEGER PROGRAMMING, pg.245. (computer science, complexity theory)
- Gärtner, Bernd; Matoušek, Jiří (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)
- Gerard Sierksma; Yori Zwols (2015). *Linear and Integer Optimization: Theory and Practice*. CRC Press. ISBN 978-1-498-71016-9.
- Gerard Sierksma; Diptesh Ghosh (2010). *Networks in Action; Text and Computer Exercises in Network Optimization*. Springer. ISBN 978-1-4419-5512-8. (linear optimization modeling)
- H. P. Williams, *Model Building in Mathematical Programming (https://books.google.com/books?id=YJRh0tOes7UC&printsec=frontcover#v=onepage&q&f=false)*, Fifth Edition, 2013. (Modeling)
- Stephen J. Wright, 1997, *Primal-Dual Interior-Point Methods (https://books.google.com/books?id=oQdBzXhZeUkC&printsec=frontcover#v=onepage&q&f=false)*, 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://lpsolve.sourceforge.net/4.0/LinearProgrammingFAQ.htm)
- Benchmarks For Optimisation Software (http://plato.asu.edu/bench.html)
