L

Lagrange, Joseph-Louis

TANIA QUERIDO, DUKWON KIM
University Florida, Gainesville, USA

MSC2000: 01A99

Article Outline

Keywords

Calculus of variations; Lagrange multipliers

J.L. Lagrange (1736–1813) made significant contributions to many branches of mathematics and physics, among them the theory of numbers, the theory of equations, ordinary and partial differential equations, the calculus of variations, analytic geometry and mechanics. By his outstanding discoveries he threw the first seeds of thought that later nourished C.F. Gauss and N.H. Abel.

During the first thirty years of his life he lived in Turin (France, now Italy) and, as a boy, his tastes were more classical than scientific. His interest in mathematics began while he was still in school when he read a paper by E. Halley on the uses of algebra in optics. He then began a course of independent study, and excelled so rapidly in the field of mathematical analysis that by the age of nineteen he was appointed professor at the Royal Artillery School and helped to found the Royal Academy of Science in 1757. His ideas had greatly impressed L. Euler, one of the giants of European mathematics. Euler and Lagrange, together, would join the first rank of the eighteenth century mathematicians, and their careers and research where often related [5].

In 1759 Lagrange focused his research in analysis and mechanics and wrote 'Sur la Propagation du son dans les fluides', a very difficult issue for that time [4]. From 1759 to 1761 he had his first publications in the 'Miscellanea of the Turin Academy'. His reputation was established.

Lagrange developed a new calculus which would enrich the sciences, called *calculus of variations*. In its simplest form the subject seeks to determine a functional relationship $y = f(x)$ such that an $\int_a^b g(x, y)\, dx$ could produce a maximum or a minimum. It was viewed as a mathematical study of economy or the 'best income' [4]. That was Lagrange's earliest contribution to the optimization area.

In 1766, Lagrange was appointed the Head of the Berlin Academy of Science, succeeding Euler. In offering this appointment, Frederick the Great wanted to turn his Academy into one of the best institutes of its day, proclaiming that the 'greatest mathematician in Europe' should live near the 'greatest king in Europe' [1]. During this period, he had a prosperous time, developing important works in the field of calculus, introducing the strictness in the calculus differential and integral. Later (1767) he published a memoir on the approximation of roots of polynomial equations by means of continued fractions; in 1770 he wrote a paper considering the *solvability of equations* in terms of permutations on their roots.

After Frederick's death, Lagrange left Berlin and became a member of the Paris Academy of Science by the invitation of Louis XVI (1787). He remained in Paris for the rest of his career, making a lengthy treatise on

the numerical solution of equations, representing a significant portion of his mathematical research. His papers on solution of third - and fourth-degree polynomial equations by radicals received considerable attention. His methods, laid in the early development of group theory to solving polynomials, were later taken by E. Galois. Lagrange's name was attached to one of the most important theorems of group theory [3]:

Theorem 1 *If o is the order of a subgroup g of a group G of order O, then o is a factor of O.*

In 1788 he published his masterpiece, the treatise 'Méchanique Analytique', which summarized and unified all the work done in the field of general mechanics since the time of I. Newton. This work, notable for its use of theory of differential equations, transformed mechanics into a branch of mathematical analysis. As W. Hamilton later said, 'he made a kind of scientific poem' [6].

In 1793, Lagrange headed a commission, which included P.S. Laplace and A. Lavoisier, to devise a new system of weights and measures. Out of this came the metric system.

Lagrange developed the method of variation of parameters in the solution of nonhomogeneous linear differential equations. In the determination of maxima and minima of a function, say $f(x, y, z, w)$, subject to constraints such as $g(x, y, z, w) = 0$ and $h(x, y, z, w) = 0$, he suggested the use of *Lagrange multipliers* to provide an elegant algorithm. By this method two undetermined constants λ and μ are introduced, forming the function $F \equiv f + \lambda g + \mu h$, from the related equations $F_x = 0$, $F_y = 0$, $F_z = 0$, $F_w = 0$, $g = 0$, and $h = 0$, the multipliers λ and μ are then eliminated, and the problem is solved. This procedure and its variations have emerged as a very important class of optimization method [1,2].

One can characterize Lagrange's contribution to optimization as his formalist foundation. Most of his results were retained and developed further by the following generations, who gave to his theory a different and practical course.

By the end of his life, Lagrange could not think futuristically for the mathematics. He felt that other sciences such as chemistry, physics and biology would attract the ablest minds of the future. His pessimism was unfounded. Much more was to be forthcoming with Gauss and his successors, making the nineteenth century the richest in the history of mathematics.

See also

- ▶ Decomposition Techniques for MILP: Lagrangian Relaxation
- ▶ Integer Programming: Lagrangian Relaxation
- ▶ Lagrangian Multipliers Methods for Convex Programming
- ▶ Multi-objective Optimization: Lagrange Duality

References

1. Bertsekas DP (1986) Constrained optimization and Lagrange multiplier methods. Athena Sci., Belmont, MA
2. Boyer CB (1968) A history of mathematics. Wiley, New York
3. Fraser CG (1990) Lagrange's analytical mathematics, its Cartesian origins and reception in Comte's positive philosophy. Studies History Philos Sci 21(2):243–256
4. Julia G (1942-1950) La vie et l'oeuvre de J.-L. Lagrange. Enseign Math 39:9–21
5. Koetsier T (1986) Joseph Louis Lagrange (1736-1813): His life, his work and his personality. Nieuw Arch Wisk (4) 4(3):191–205
6. Simmons GF (1972) Differential equations, with applications and historical notes. McGraw-Hill, New York

Lagrange-Type Functions

A. M. Rubinov[1], X. Q. Yang[2]

[1] School of Information Technology and Mathematical Sciences, The University of Ballarat, Ballarat, Australia

[2] Department of Applied Mathematics, The Hong Kong Polytechnic University, Kowloon, Hong Kong

MSC2000: 90C30, 90C26, 90C46

Article Outline

Keywords and Phrases

Lagrange-type function; IPH functions; Multiplicative inf-convolution; Zero duality gap; Exact penalty function

Lagrange and penalty function methods provide a powerful approach, both as a theoretical tool and a computational vehicle, for the study of constrained optimization problems. However, for a nonconvex constrained optimization problem, the classical Lagrange primal-dual method may fail to find a minimum as a zero duality gap is not always guaranteed. A large penalty parameter is, in general, required for classical quadratic penalty functions in order that minima of penalty problems are a good approximation to those of the original constrained optimization problems. It is well-known that penalty functions with too large parameters cause an obstacle for numerical implementation. Thus the question arises how to generalize classical Lagrange and penalty functions, in order to obtain an appropriate scheme for reducing constrained optimization problems to unconstrained ones that will be suitable for sufficiently broad classes of optimization problems from both the theoretical and computational viewpoints.

One of the approaches for such a scheme is as follows: an unconstrained problem is constructed, where the objective function is a convolution of the objective and constraint functions of the original problem. While a linear convolution leads to a classical Lagrange function, different kinds of nonlinear convolutions lead to interesting generalizations. We shall call functions that appear as a convolution of the objective function and the constraint functions, *Lagrange-type functions*. It can be shown that these functions naturally arise as a result of a nonlinear separation of the image set of the problem and a cone in the image-space of the problem under consideration (see [4]). The class of Lagrange-type functions includes also augmented Lagrangians, corresponding to the so-called canonical dualizing parameterization. However, augmented Lagrangians constructed by means of some general dualizing parameterizations cannot be included in this scheme.

Consider the following problem $P(f,g)$:

$$\min f(x) \quad \text{subject to} \quad x \in X, g(x) \leq 0\,,$$

where X is a metric space, f is a real-valued function defined on X, and g maps X into $\mathbb{R}^m$, that is, $g(x) = (g_1(x), \ldots, g_m(x))$, where g_i are real-valued functions defined on X. We assume that the set of feasible solutions $X_0 = \{x \in X\colon g(x) \leq 0\}$ is nonempty and that the objective function f is bounded from below on X.

Let Ω be a set of parameters and $h\colon \mathbb{R}^{1+m} \times \Omega \to \mathbb{R}$ be a function. Let $\eta \in \mathbb{R}$. Then the function

$$L(x,\omega) = h(f(x) - \eta, g(x); \omega) + \eta, \quad x \in X, \omega \in \Omega\,, \tag{1}$$

is called a Lagrange-type function for problem $P(f,g)$ corresponding to h and η, and h is called a convolution function.

If h is linear with respect to the first variable, more specifically:

$$h(u, v; \omega) = u + \chi(v; \omega)\,,$$

where $\chi\colon \mathbb{R}^m \times \Omega \to \mathbb{R}$ is a real-valued function, then the parameter η can be omitted. Indeed, for each $\eta \in \mathbb{R}$, we have

$$L(x,\omega) = f(x) + \chi(g(x); \omega)\,.$$

However in general nonlinear situation the presence of η is important and different η lead to Lagrange-type functions with different properties.

One of the possible choices of the number η is $\eta = f(x_*)$ where x_* is a reference point, in particular x_* is a solution of $P(f,g)$ (see [4]). Then the Lagrange-type function has the form

$$L(x,\omega) = h(f(x) - f(x_*), g(x); \omega) + f(x_*)\,, \quad x \in X, \omega \in \Omega\,.$$

The Lagrange-type function (1) is a very general scheme and includes linear Lagrange functions, classical penalty functions, and augmented Lagrange functions as special cases.

Let $\Omega = \mathbb{R}^m_+$ and p be a real-valued function defined on $\mathbb{R}^{1+m}$. Define

$$h(u, v; \omega) = p(u, \omega_1 v_1, \ldots, \omega_m v_m)\,. \tag{2}$$

The Lagrange-type function has the form

$$L_p(x,\omega) = p(f(x) - \eta, \omega_1 g_1(x), \ldots, \omega_m g_m(x)) + \eta\,.$$

We can obtain fairly good results if the function p enjoys some properties. In particular, we assume that

(i) p is increasing;

(ii) $p(u, 0_m) \leq u$, for all $u \in \mathbb{R}$. (Here 0_m is the origin of $\mathbb{R}^m$.) One more assumption is useful for applications.

(iii) p is positively homogeneous ($p(\lambda x) = \lambda p(x)$ for $\lambda > 0$).

If both (i) and (iii) hold, then p is called an IPH function. Let p be a real-valued function defined on $\mathbb{R}^{1+m}$ and h be a convolution function defined by (2). Then

(a) If p enjoys properties (i) and (ii), then

$$\sup_{w\in\Omega} h(u,v;w) \le u, \quad \forall u \in \mathbb{R}, v \in \mathbb{R}^m_- .$$

(b) If p is an IPH function and $p(u, e_i) > 0$, where e_i is the i-th unit vector, $i = 1, \dots, m$, then

$$\sup_{w\in\Omega} h(u,v;w) = +\infty, \quad \forall v \notin \mathbb{R}^m_- .$$

We now give some examples of Lagrange-type functions. First two examples correspond to functions of the form (2).

1) Let $p(u,v) = u + \sum_{i=1}^m v_i$. Then $L_p(x,\omega) = f(x) + \sum_{i=1}^m \omega_i g_i(x)$ coincides with the classical Lagrange function.
2) Let $p(u,v) = u + \sum_{i=1}^m v_i^+$ where $v^+ = \max(v,0)$. Then $L_p(x,\omega) = f(x) + \sum_{i=1}^m \omega_i g_i(x)^+$ coincides with the classical (linear) penalty function. If $p(u,v) = u + \sum_{i=1}^m (v_i^+)^2$, then $L_p(x,\omega) = f(x) + \sum_{i=1}^m \omega_i (g_i(x)^+)^2$ is a quadratic penalty function.

We now give the definition of a penalty-type function. Let Ω be a set of parameters and $h: \mathbb{R}^{1+m} \times \Omega \to \mathbb{R}$ be a convolution function with the property:

$$h(u,v;\omega) = u, \quad u \in \mathbb{R}, v \in \mathbb{R}^m_-, \omega \in \Omega .$$

Then the Lagrange-type function $L(x,\omega)$, corresponding to h, is called a penalty-type function.

Next two examples cannot be presented in the form (2).

3) Augmented Lagrangians

Let $\sigma: \mathbb{R}^m \to \mathbb{R}$ be an augmenting function, i. e., $\sigma(0) = 0$ and $\sigma(z) > 0$, for $z \neq 0$, and $\Omega \subset \{(y,r): y \in \mathbb{R}^m, r \ge 0\}$ be a set of parameters satisfying $(0,0) \in \Omega$ and $(y,r) \in \Omega$ implying $(y,r') \in \Omega$, for all $r' \ge r$. Let $h: \mathbb{R}^m \times \Omega \to \mathbb{R}$ be the convolution function defined by

$$\begin{aligned} h(u,v;(y,r)) &= \inf_{z+v\le 0} (u - [y,z] + r\sigma(z)) \\ &= u + \inf_{z+v\le 0} (-[y,z] + r\sigma(z)) . \end{aligned}$$

Then the Lagrange-type function, corresponding to $\eta = 0$, coincides with the augmented Lagrangian [5], that is,

$$\begin{aligned} L(x,(y,r)) &= h(f(x), g(x); (y,r)) \\ &= \inf_{z+g(x)\le 0} (f(x) - [y,z] + r\sigma(z)) . \end{aligned}$$

4) Morrison-type functions. Let $\Omega = \mathbb{R}_+$ and

$$h(u,v,\omega) = ((u-\omega)^+)^2 + \sigma(v_1^+, \dots, v_m^+) ,$$

where σ is an augmenting function. Then the Lagrange-type function corresponding to $\eta = 0$ has the form

$$L(x,\omega) = ((f(x)-\omega)^+)^2 + \sigma(g_1(x)^+, \dots, g_m(x)^+) .$$

Functions of this kind have been introduced by Morrison [6].

Consider problem $P(f,g)$, a convolution function $h: \mathbb{R}^{1+m} \times \Omega \to \mathbb{R}$ and the corresponding Lagrange-type function

$$L(x,\omega) = h(f(x) - \eta, g(x); \omega) + \eta .$$

The dual function $q: \Omega \to \bar{\mathbb{R}} = \mathbb{R} \cup \{-\infty, +\infty\}$ of $P(f,g)$ with respect to h and η is defined by

$$q(\omega) = \inf_{x\in X} h(f(x) - \eta, g(x); \omega) + \eta, \quad \omega \in \Omega .$$

Consider the dual problem to $P(f,g)$ with respect to h and η:

$$\max q(\omega), \quad \text{subject to} \quad \omega \in \Omega .$$

We are interested in the following questions: Find conditions under which

1) the weak duality holds, i. e.,

$$M(f,g) := \inf_{x\in X_0} f(x) \ge \sup_{\omega\in\Omega} q(\omega) := M^*(f,g) ;$$

2) the zero duality gap property holds, i. e.,

$$\inf_{x\in X_0} f(x) = \sup_{\omega\in\Omega} q(\omega) ;$$

3) an exact Lagrange parameter exists, i. e., the weak duality holds and there exists $\bar{\omega} \in \Omega$ such that

$$\inf_{x\in X_0} f(x) = \inf_{x\in X} L(x,\bar{\omega}) ;$$

4) a strong exact parameter exists: there exists an exact parameter $\bar{\omega} \in \Omega$ such that

$$\begin{aligned}\operatorname{argmin} P(f,g) &:= \operatorname{argmin}_{x\in X_0} f(x)\\ &= \operatorname{argmin}_{x\in X} L(x,\bar{\omega})\,;\end{aligned}$$

5) a saddle point exists and generates a solution of $P(f,g)$. The first part of this question means that there exists $(x_*,\omega_*) \in X\times\Omega$ such that

$$\begin{aligned}L(x,\omega_*) \le L(x_*,\omega_*) \le L(x_*,\omega),\\ x\in X\,,\ \omega\in\Omega\,.\end{aligned} \tag{3}$$

The second part means that (3) implies $x_* \in \operatorname{argmin} P(f,g)$.

The weak duality allows one to estimate from below the optimal value $M(f,g)$ by solving the unconstrained problem $\inf_{x\in X} L(x,\omega)$. The zero duality gap property allows one to find $M(f,g)$ by solving a sequence of unconstrained problems $\inf_{x\in X} L(x,\omega_t)$ where $\{\omega_t\}\subset\Omega$. The existence of an exact Lagrange parameter $\bar{\omega}$ means that $M(f,g)$ can be found by solving one unconstrained problem $\inf_{x\in X} L(x,\bar{\omega})$. The existence of a strong exact parameter $\bar{\omega}$ means that the solution set of $P(f,g)$ is the same as that of $\min_{x\in X} L(x,\bar{\omega})$.

Let $h : \mathbb{R}^{1+m}\times\Omega\to\mathbb{R}$ be a convolution function such that

$$\sup_{\omega\in\Omega} h(u,v;\omega) \le u, \quad \text{for all } (u,v)\in\mathbb{R}\times\mathbb{R}^m_-\,. \tag{4}$$

Then the weak duality holds.

Condition (4) can be guaranteed if

$$\begin{aligned}h(u,v;w) = p(u,w_1v_1,\dots,w_mv_m)\,,\\ (u,v)\in\mathbb{R}^{1+m}, w\in\mathbb{R}^m_+\,,\end{aligned}$$

and $p\colon \mathbb{R}^{1+m}\to\mathbb{R}$ is an IPH function satisfying

$$p(1,0_m)\le 1, \quad p(-1,0_m)\le -1\,.$$

Assume that η is a lower estimate of the function f over the set X, i. e., $f(x)-\eta \ge b>0$, for all $x\in X$. Then, in order to establish the weak duality, we need only to consider convolution functions defined on $[b,+\infty)\times\mathbb{R}^m\times\Omega$ such that

$$\sup_{\omega\in\Omega} h(u,v;\omega)\le u, \quad \forall (u,v)\in[b,+\infty)\times\mathbb{R}^m_-\,. \tag{5}$$

To investigate the zero duality gap property, we further assume that, for any $\epsilon\in(0,b)$, there exists $\delta>0$ such that

$$\inf_{\omega\in\Omega} h(u,v;\omega)\ge u-\epsilon, \quad \forall u\ge b, r(v)\le\delta\,; \tag{6}$$

and that, for each $c>0$, there exists $\bar{\omega}\in\Omega$ such that

$$h(u,v;\bar{\omega})\ge cr(v), \quad \forall u\ge b, v\in\mathbb{R}^m\,, \tag{7}$$

where $r\colon\mathbb{R}^m\to\mathbb{R}$ is such that $r(v)\le 0 \iff v\in\mathbb{R}^m_-$.

Assume further that

(f_1) The function f is uniformly positive on X_0, i. e.,

$$\inf_{x\in X_0} f(x) = M(f,g)>0\,;$$

(f_2) The function f is uniformly continuous on an open set containing the set X_0;

(g) The mapping g is continuous and the set-valued mapping

$$D(\delta) = \{x\in X\colon r(g(x))\le\delta\}$$

is upper semi-continuous at the point $\delta=0$.

Theorem 1 *Under the assumptions (5)–(7) and (f_1), (f_2) and (g), the zero duality gap property holds for P(f,g) with respect to the Lagrange-type function $L(x,\omega)$, corresponding to h and $\eta=0$.*

Let $b\ge 0$. Define a convolution function $h\colon[b,+\infty)\times\mathbb{R}^m\to\mathbb{R}$ by

$$h(u,v;\omega) = p(u,\omega_1v_1,\dots,\omega_mv_m)\,,$$

where $p\colon\mathbb{R}_+\times\mathbb{R}^m\to\mathbb{R}$ is an increasing function satisfying

$$p(u,0_m)\le u, \quad \text{for all } u\ge 0\,. \tag{8}$$

Consider the $P(f,g)$ with uniformly positive objective function f on X. Let L be the Lagrange-type function defined by

$$L(x,\omega) = p(f(x),\omega_1g_1(x),\dots,\omega_mg_m(x))\,,$$

where p is defined on $\mathbb{R}_+\times\mathbb{R}^m$. Define the perturbation function $\beta(y)$ of $P(f,g)$ by

$$\beta(y) = \inf\{f(x)\colon x\in X, g(x)\le y\}, \quad y\in\mathbb{R}^m\,.$$

Theorem 2 *Let p be a continuous increasing function satisfying (7). Let the zero duality gap property with respect to p holds. Then the perturbation function β is lower semi-continuous at the origin.*

Further assume that p satisfies the following property: there exist positive numbers $a_1, \ldots, a_m$ such that, for all $u > 0$, $(v_1, \ldots, v_m) \in \mathbb{R}^m$, we have

$$p(u, v_1, \ldots, v_m) \geq \max(u, a_1 v_1, \ldots, a_m v_m)\,. \qquad (9)$$

Theorem 3 *Assume that p is an increasing convolution function that possesses properties (8) and (9). Let perturbation function β of problem P(f,g) be lower semi-continuous at the origin. Then the zero duality gap property with respect to p holds.*

Remark 1 The perturbation function β depends on $P(f,g)$ and doesn't depend on the exogenous function p. It is worth noting that Theorems 2 and 3 establish equivalence relations between the zero duality gap property with respect to different p from a broad class of convolution functions.

Remark 2 If p is a linear function, then the lower semicontinuity does not imply the zero duality gap property, so we need to impose a condition that does not hold for linear functions. This is the role of (9). The results similar to Theorem 2 and Theorem 3 hold also for penalty type functions, where $p(u,v)$ is a function defined on $\mathbb{R}_+ \times \mathbb{R}^m_+$ and $L(x,\omega) = p(f(x), \omega_1 g_1(x)^+, \ldots, g_m(x)^+)$. In such a case (9) should be valid only for $u > 0$, $v \in \mathbb{R}^m_+$. This requirement is very weak and is valid for many increasing functions including the function $p(u,v) = u + \sum_{i=1}^m v_i$.

Let the Lagrange-type function be of the following form

$$L(x,\omega) = f(x) + \chi(g(x);\omega), \quad x \in X, \omega \in \Omega\,.$$

Consider set K of functions $\chi\colon \mathbb{R}^m \times \Omega \to \mathbb{R}$ with the following two properties

(i) $\chi(\cdot,\omega)$ is lower semi-continuous for all $\omega \in \Omega$;

(ii) $\sup_{\omega\in\Omega} \chi(v;\omega) = 0$, for all $v \in \mathbb{R}^m_-$.

Consider a point $(x_*, \omega_*) \in X \times \Omega$ such that

$$L(x_*, \omega_*) = \min_{x\in X} L(x, \omega_*)\,, \qquad (10)$$

$$\chi(g(x_*);\omega_*) = 0\,. \qquad (11)$$

Theorem 4 *Let $\chi \in K$. If (10) and (11) hold for $x_* \in X_0$ and $\omega_* \in \Omega$, then ω_* is an exact Lagrange parameter.*

The most advanced theory has been developed for two special classes of Lagrange-type functions. One of them is augmented Lagrangians (see article in encyclopedia). The other class consists of penalty-type functions for problems with a positive objective and a single constraint. This penalty-type functions are composed by convolutions functions of the form (2) with IPH functions p.

Remark 3 Consider problem $P(f,g)$ with m constraints $g_1, \ldots, g_m$. We can convert these constraints to a single one by many different ways. In particular, the system $g_1(x) \leq 0, \ldots, g_m(x) \leq 0$ is equivalent to the single inequality $f_1(x) := \sum_{i=1}^m g_i^+(x) \leq 0$. The function f_1 is non-smooth. If all functions $g_i(x)$ are smooth then a smoothing procedure can be applied to f_1 (see [13] for details). Problems with a single constraint are convenient to be dealt with from many points of view.

Let $P(f,f_1)$ be a problem with a positive objective f and a single constraint f_1. We consider here only IPH functions s_k defined on $\mathbb{R}^2_+$ by:

$$s_k(u,v) = (u^k + v^k)^{1/k}, \quad u, v \geq 0\,. \qquad (12)$$

(Many results that are valid for s_k can be extended also for IPH functions $p\colon \mathbb{R}^2_+ \to \mathbb{R}_+$ with properties $p(1,0) = 1$, $\lim_{u\to+\infty} p(1,u) = +\infty$.)

A penalty-type function L_k^+ corresponding to s_k has the form $L_k^+(x,d) = (f(x)^k + d^k f_1^+(x)^k)^{1/k}$. Here d^k is a penalty parameter. It can be shown that the exact parameter does not exist if $k > 1$ for the 'regular' problems in a certain sense, so we will here consider only the classical penalty function with k=1 and lower order penalty functions with $k < 1$. It can be shown that the existence of an exact parameter for $k \leq 1$ implies the existence of exact parameters for k' with $0 < k' < k$. One of the main questions that can be studied in the framework of this class of penalty-type functions is the size of exact penalty parameters. Generally speaking, we can diminish the size of exact parameter using the choice of k and some simple reformulations of the problem $P(f,f_1)$ in hand.

For the function L_k^+ an explicit value of the least exact penalty parameter can be expressed through the

perturbation function. Let $\beta(y)$ be the perturbation function of the problem $P(f,f_1)$. Note that $\beta(0) = M(f,f_1)$ and β is a decreasing function, so $\beta(y) \leq M(f,f_1)$. For the sake of simplicity, we assume that $\beta(y) < M(f,f_1)$ for all $y > 0$. Let

$$\bar{d}_k = \sup_{y>0} \frac{(M(f,f_1)^k - \beta^k(y))^{1/k}}{y}. \tag{13}$$

Then the least exact parameter exists if and only if the supremum in (13) is finite and the least exact parameter is equal to $\bar{d}_k$. For $k=1$ the existence easily follows from the calmness results of Burke [1].

Let $f^c(x) = f(x) + c$ with $c > 0$ and $d_{c,k}$ be the least exact parameter for problem $P(f^c, f_1)$. Then it can be proved that $d_{c,k} \to 0$ as $c \to +\infty$.

Assume that functions f and f_1 are Lipschitz. Since $k < 1$ the function L_k^+ is not locally Lipschitz at points x where $f_1(x) = 0$, so we need to have a special smoothing procedure in order to apply numerical method for the unconstrained minimization of this function. Such a procedure is described in [14]. This procedure can be applied for different types of lower order penalty functions.

Another approach for constructing a Lipschitz penalty function with a small exact parameter is also of interest (see [11] and references therein).

Let σ be a strictly increasing continuous concave function defined on $[a, +\infty)$ where $a > 0$. Assume that $\sigma(a) \geq 0$ and $\lim_{y\to+\infty} \sigma(y) = 0$ where σ'_+ is the right derivative of the concave function σ. Consider the function $f^{\sigma,c}(x) = \sigma(f(x) + c)$ and the classical penalty function for $L^+_{1,\sigma,c}(x,d) = \sigma(f(x)+c) + df_1(x)$ for the problem $P(f^{\sigma,c}, f_1)$. Let $d_{\sigma,c}$ be the least exact parameter of $L^+_{1,\sigma,c}$ (assuming that this parameter exists). Then we can assert that $d_{\sigma,c} \to 0$ as $c \to 0$ under very mild assumptions.

References

1. Burke JV (1991) Calmness and exact penalization. SIAM J Control Optim 29:493–497
2. Burke JV (1991) An exact penalization viewpoint of constrained optimization. SIAM J Control Optim 29:968–998
3. Clarke FH (1983) Optimization and nonsmooth analysis. Wiley, New York
4. Giannessi F (2005) Constrained optimization and image space analysis, vol 1. Separation of sets and optimality conditions. Mathematical concepts and methods in science and engineering, vol 49. Springer, New York, p 395
5. Huang XX, Yang XQ (2003) A unified augmented lagrangian approach to duality and exact penalization. Math Oper Res 28(3):533–552
6. Morrison DD (1968) Optimization by least squares. SIAM J Numer Anal 5:83–88
7. Rockafellar RT, Wets RJ-B (1998) Variational analysis. Springer, Berlin
8. Rubinov AM (2000) Abstract convexity and global optimization. Kluwer, Dordrecht
9. Rubinov AM, Glover BM, Yang XQ (2000) Decreasing functions with application to penalization. SIAM J Optim 10:289–313
10. Rubinov AM, Glover BM, Yang XQ (1999) Modified lagrangian and penalty functions in continuous optimization. Optimization 46:327–351
11. Rubinov AM, Yang XQ (2003) Lagrange-type functions in constrained non-convex optimization. Kluwer, Dordrecht
12. Rubinov AM, Yang XQ, Bagirov AM (2002) Nonlinear penalty functions with a small penalty parameter. Optim Methods Softw 17:931–964
13. Teo KL, Goh CJ, Wong KH (1991) A unified computational approach to optimal control problems. In: Pitman monographs and surveys in pure and applied mathematics, vol 55. Longman Scientific and Technical, Harlow, p 329
14. Wu ZY, Bai FS, Yang XQ, Zhang LS (2004) An exact lower order penalty function and its smoothing in nonlinear programming. Optimization 53:51–68
15. Yang XQ, Huang XX (2001) A nonlinear lagrangian approach to constrained optimization problems. SIAM J Optim 14:1119–1144
16. Yevtushenko YG, Zhadan VG (1990) Exact auxiliary functions in optimization problems. USSR Comput Math Math Phys 30:31–42

Lagrangian Duality: BASICS

Donald W. Hearn[1], Timothy J. Lowe[2]

[1] University Florida, Gainesville, USA

[2] University Iowa, Iowa City, USA

MSC2000: 90C30

Article Outline

Keywords

Primal optimization problem; Dual optimization problem; Subgradient; Duality gap; Constraint qualification

The Primal Problem and the Lagrangian Dual Problem

For a given *primal optimization problem* (P) it is possible to construct a related *dual* problem which depends on the same data and often facilitates the analysis and solution of (P). This section focuses on the *Lagrangian dual*, a particular form of dual problem which has proven to be very useful in many optimization applications.

A general form of *primal problem* is

$$(\mathrm{P}) \quad \begin{cases} \min & f(x) \\ \text{s.t.} & g(x) \le 0, \\ & h(x) = 0, \\ & x \in S, \end{cases}$$

where f is a scalar function of the n-dimensional vector x, and g and h are vector functions of x. S is a nonempty subset of $\mathbf{R}^n$. It is convenient to associate *dual variables* with the constraints as follows: components of the *dual vector* u correspond to components of the vector constraint $g(x) \le 0$, and similarly the components of v are associated with components of the constraint $h(x) = 0$.

There is a great deal of flexibility in defining problem (P). For example, any or all of the explicit constraints $g(x) \le 0$ and $h(x) = 0$ could be incorporated in the definition of the set S. This, of course, governs the number and type of dual variables. As will be seen in the examples, defining S is the first step in defining a Lagrangian dual of (P). To illustrate the basic duality results, certain assumptions regarding the functions f, g and h and the set S will be made to simplify the presentation below. For more thorough treatments, see the references.

Given (P), define the *Lagrangian function* $L(x, u, v) = f(x) + u^{\mathsf{T}} g(x) + v^{\mathsf{T}} h(x)$. The *Lagrangian dual problem* is then

$$(\mathrm{D}) \quad \begin{cases} \max & \theta(u, v) \\ \text{s.t.} & u \ge 0, \end{cases}$$

where, for fixed (u, v), the dual function θ is defined in terms of the *infimum of the Lagrangian function* with respect to $x \in S$:

$$\theta(u, v) = \inf_{x \in S} L(x, u, v).$$

Below are five examples of primal problems and their duals. The first is a geometrical example, three are classes of optimization problems: linear programs, convex programs, and quadratic programs, and the final example is an integer program.

Example 1 (geometrical problem) In this two-variable example, a linear function is to be minimized over the intersection of the unit disk and the nonnegative orthant. The optimal solution is at the origin with objective value zero.

$$(\mathrm{P1}) \quad \begin{cases} \min & x_1 + x_2 \\ \text{s.t.} & x_1^2 + x_2^2 \le 1, \\ & -x_1 \le 0, \\ & -x_2 \le 0, \end{cases}$$

Letting $S = \{(x_1, x_2): x_1^2 + x_2^2 \le 1\}$, the dual problem is

$$(\mathrm{D1}) \quad \begin{cases} \max & \theta(u) \\ \text{s.t.} & u \ge 0, \end{cases}$$

where

$$\theta(u) = \min_{x_1^2 + x_2^2 \le 1} (1 - u_1)x_1 + (1 - u_2)x_2.$$

Note that min replaces inf in the definition of θ since it is clear that the infimum exists and is finite for this example.

Example 2 (linear programming) Duality is an important topic in any treatment of linear programming. This example shows that the Lagrangian dual of a primal linear program is equivalent to the *dual linear program* as

it is usually formulated in textbooks. Letting the primal be

$$\text{(P2)} \quad \begin{cases} \min & c^\top x \\ \text{s.t.} & b - Ax \le 0, \\ & x \ge 0, \end{cases}$$

and choosing $S = \{x: x \ge 0\}$, the Lagrangian dual is

$$\text{(D2)} \quad \begin{cases} \max & \theta(u) \\ \text{s.t.} & u \ge 0 \end{cases}$$

where

$$\theta(u) = \inf_{x \ge 0} c^\top x + u^\top (b - Ax).$$

This reduces to

$$\theta(u) = b^\top u + \begin{cases} 0 & \text{if } (c - A^\top u) \ge 0, \\ -\infty & \text{otherwise.} \end{cases}$$

Assuming there are nonnegative values of u such that $c \ge A^\intercal u$, these would be the only viable choices for the maximization of $\theta(u)$ and therefore (D2) takes the form familiar from linear programming duality:

$$\begin{cases} \max & b^\top u \\ \text{s.t.} & A^\top u \le c, \\ & u \ge 0. \end{cases}$$

Example 3 (differentiable convex programming) One of the first nonlinear duals was developed by P. Wolfe [27] for the primal problem

$$\text{(P3)} \quad \begin{cases} \min & f(x) \\ \text{s.t.} & g(x) \le 0, \\ & x \in S, \end{cases}$$

where S is an open convex set, and f and g are differentiable convex functions defined on S. The Lagrangian function is $L(x, u) = f(x) + u^\intercal\, g(x)$, and it is further assumed that $\theta(u) \ne -\infty$ for all $u \ge 0$. With these assumptions the Lagrangian function is convex in x and has a minimum where its gradient is zero. That is, the requirement $\theta\ (u) = \min_{x \in S} L(x, u)$ is the same as requiring $\nabla_x L(x, u) = 0$. Thus the dual problem may be written

$$\text{(D3)} \quad \begin{cases} \max & L(x, u) \\ \text{s.t.} & \nabla_x L(x, u) = 0, \\ & u \ge 0. \end{cases}$$

Example 4 (convex quadratic programming) An important special case of the preceding example is the problem

$$\text{(P4)} \quad \begin{cases} \min & \frac{1}{2} x^\top H x + d^\top x \\ \text{s.t.} & Ax \le b, \\ & x \in \mathbb{R}^n, \end{cases}$$

where H is a given symmetric positive definite $n \times n$ matrix and d is a given vector in $\mathbf{R}^n$. Applying the results for (P3) above and using the equality constraints of (D3) to eliminate x, the dual of can be written

$$\text{(D4)} \quad \begin{cases} \max & \theta(u) = -\frac{1}{2} u^\top (AH^{-1}A^\top) u \\ & \quad -u^\top (b + AH^{-1}d) - \frac{1}{2} d^\top H^{-1} d \\ \text{s.t.} & u \ge 0. \end{cases}$$

Thus, the dual of (P4) is also a quadratic program in the dual variables u.

Example 5 (integer program) The following numerical example of a linear problem with binary variables will be used to illustrate various dual properties in the following sections.

$$\text{(P5)} \quad \begin{cases} \min & 20 - x_1 - 5x_2 - 7x_3 \\ \text{s.t.} & x_1 + 3x_2 + 4x_3 \le 5, \\ & x_j \in \{0, 1\}, \quad j = 1, 2, 3. \end{cases}$$

For this problem, let S be defined by the binary restrictions on the components of x. Then $L(x, u) = 20 - x_1 - 5x_2 - 7x_3 + u(x_1 + 3x_2 + 4x_3 - 5)$ and the dual problem is

$$\text{(D5)} \quad \begin{cases} \max & \theta(u) \\ \text{s.t.} & u \ge 0, \end{cases}$$

where

$$\begin{cases} \theta(u) = \min & (u - 1)x_1 + (3u - 5)x_2 \\ & \quad +(4u - 7)x_3 - 5u + 20 \\ \text{s.t.} & x_j \in \{0, 1\}, \quad j = 1, 2, 3. \end{cases}$$

Weak and Strong Duality

For a given primal problem (P) and associated dual problem (D), a fundamental relationship showing that

the two objective function values bound each other is given by the following *weak duality* result:

Theorem 6 *If $\overline{x}$ is feasible to (P) and $(\overline{u}, \overline{v})$ is feasible to (D), then*

$$f(\overline{x}) \geq \theta(\overline{u}, \overline{v}).$$

Proof

$$\begin{aligned}\theta(u, v) &= \inf_{x \in S} L(x, u, v) \\ &\leq f(\overline{x}) + \overline{u}^{\top} g(\overline{x}) + \overline{v}^{\top} h(\overline{x}) \leq f(\overline{x}).\end{aligned}$$

The first inequality follows since $\overline{x} \in S$ and the second from $\overline{u}^{\top} g(\overline{x}) \leq 0$ and $h(h(\overline{x}) = 0$. □

If the optimal primal and dual objective values are equal, *strong duality* is said to hold for the primal and dual pair. The following theorem illustrates such a result for the the pair (P3) and (D3).

Theorem 7 *Let x^* be an optimal solution for (P3) and assume the function g satisfies some constraint qualification. Then there exists a vector u^* such that (x^*, u^*) solves (D3) and*

$$f(x^*) = L(x^*, u^*).$$

Proof Under the assumptions there exists a $u^* \geq 0$ such that (x^*, u^*) satisfies the *Karush–Kuhn–Tucker conditions*:

$$\begin{aligned}&\nabla_x L(x^*, u^*) = 0, \\ &u^{*T} g(x^*) = 0,\end{aligned}$$

from which it follows that

$$f(x^*) = L(x^*, u^*)$$

and that (x^*, u^*) is feasible to (D3). Using this and the weak duality theorem gives

$$L(x^*, u^*) \geq L(x, u)$$

for any (x, u) satisfying the constraints of (D3). The results of the theorem follow. □

The references contain additional strong duality results, including cases where differentiability is not required. However, as will be seen in examples below, it often happens that there is a difference, known as the *duality gap*, between the optimal values of the primal and dual objective functions.

Properties of the Lagrangian Dual Function

The Lagrangian dual function enjoys two useful properties: it is concave and, although it is not necessarily differentiable, it is relatively straightforward to compute a *subgradient* at any dual feasible point.

Theorem 8 *θ (u, v) is concave.*

Proof For fixed x, $L(x, u, v)$ is linear in (u, v) and thus $\theta(u, v)$ is the infimum of a (possibly infinite) collection of functions linear in (u, v). □

It is important to note that the above result is true under very general conditions. In particular, it is true when the set S is discrete.

Since θ (u, v) is concave, it is known that at least one *linear supporting function* exists at each (u, v). Collectively, the gradients of all linear supports at (u, v) is called the set of *subgradients* of θ at (u, v).

For any (u, v) for which $\theta(u, v)$ is finite, denote $S(u, v)$ as the solution set of the minimization defining $\theta(u, v)$.

Theorem 9 *For fixed $(\overline{u}, \overline{v})$, let $\overline{x} \in S(\overline{u}, \overline{v})$. Then $(g(\overline{x}), h(\overline{x}))$ is a subgradient of θ at $(\overline{u}, \overline{v})$.*

Proof For any (u, v)

$$\begin{aligned}\theta(u, v) &= \inf_{x \in S} f(x) + u^{\top} g(x) + v^{\top} h(x) \\ &\leq f(\overline{x}) + u^{\top} g(\overline{x}) + v^{\top} h(\overline{x}) \\ &= f(\overline{x}) + (u - \overline{u})^{\top} g(\overline{x}) + \overline{u}^{\top} g(\overline{x}) \\ &\quad + (v - \overline{v})^{\top} h(\overline{x}) + \overline{v}^{\top} h(\overline{x}).\end{aligned}$$

Hence

$$\theta(u, v) \leq \theta(\overline{u}, \overline{v}) + g(\overline{x})^{\top}(u - \overline{u}) + h(\overline{x})^{\top}(v - \overline{v}).$$

□

If $S(\overline{u}, \overline{v})$ is a single point $\overline{x}$, then there is only one subgradient of θ at $(\overline{u}, \overline{v})$ in which case θ is differential at $(\overline{u}, \overline{v})$, i. e., $\nabla\theta(\overline{u}, \overline{v}) = (g(\overline{x}), h(\overline{x}))$.

From the above, θ is always concave and it is relatively easy to calculate a slope at any point. Much use of this is made in algorithms for large scale integer programs. Also, the fact that the maximum value of the dual provides a lower bound to the optimal objective function value in methods (such as *branch and bound*) for solving the primal problem. While strong duality

generally holds for convex programs, this is rarely true for integer programs.

Revisiting the examples of the first section, for Example 1 the Karush–Kuhn–Tucker conditions can be employed to derive

$$\theta(u) = -\sqrt{(1-u_1)^2 + (1-u_2)^2}.$$

There is no duality gap for this problem, the dual maximum occurs at $(u_1, u_2) = (1, 1)$ where θ is zero, in agreement with the primal minimum. The dual function is differentiable except at its maximizing point. The dual function of Example 2, a linear program, is linear and thus it is concave and differentiable everywhere. Similarly, in Example 4, since H is positive definite, H^{-1} is also positive definite and the dual function is again concave and differentiable everywhere. For Example 5, the integer program, values of u feasible to the dual problem, $S(u)$ and $\theta(u)$ are given in Table 1. $S(u)$ is the triple $(x_1(u), x_2(u), x_3(u))$.

Figure 1 is a graph of the function $\theta(u)$. Again, $\theta(u)$ is a concave function and it is differentiable except at

$$u = 1, \frac{5}{3}, \frac{7}{4}.$$

The maximum dual value is

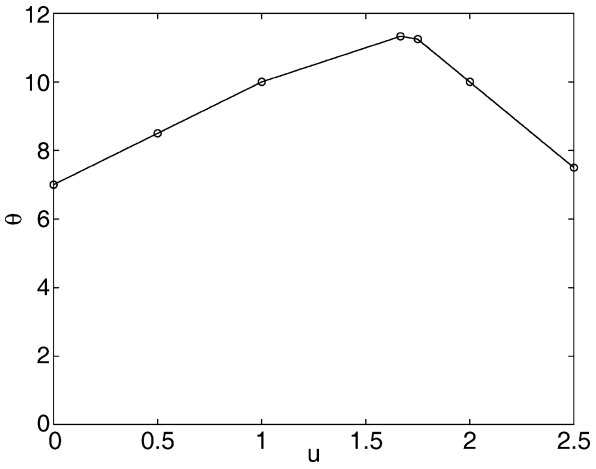
$$\theta\left(\frac{5}{3}\right) = 11\frac{1}{3},$$

which indicates a duality gap of size 2/3 since the optimal value of (P5) is $f(1, 0, 1) = 12$.

By contrast, Theorem 8 does not apply in Example 3 because the objective of (D3), a Lagrangian function, depends on both x and u, rather than the dual variables alone. Lagrangian functions are generally not concave.

Lagrangian Duality: BASICS, Table 1
Values of the dual function for Example 5

u	$S(u)$	$\theta(u)$
$0 < u < 1$	$(1, 1, 1)$	$7 + 3u$
1	$\{(1, 1, 1) \cup (0, 1, 1)\}$	$8 + 2u$
$1 < u < 5/3$	$(0, 1, 1)$	$8 + 2u$
5/3	$\{(0, 1, 1) \cup (0, 0, 1)\}$	$13 - u$
$5/3 < u < 7/4$	$(0, 0, 1)$	$13 - u$
7/4	$\{(0, 0, 1) \cup (0, 0, 0)\}$	$20 - 5u$
$7/4 < u$	$(0, 0, 0)$	$20 - 5u$



Lagrangian Duality: BASICS, Figure 1
***θ (u)* for Example 5**

Geometrical Interpretations of Lagrangian Duality

The Resource-Payoff Space

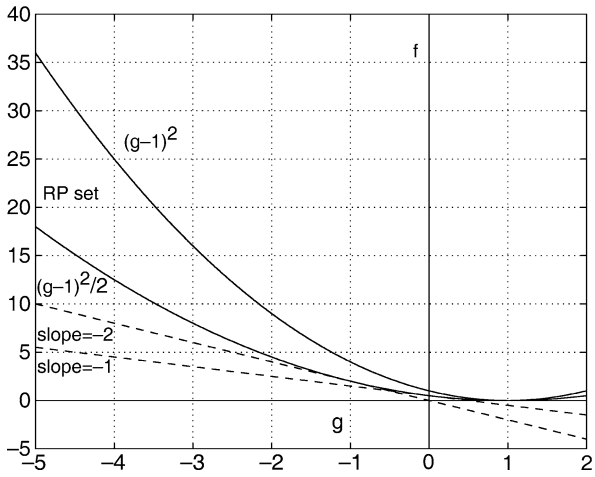
One interpretation of the dual problem is provided via the resource-payoff set RP for problem (P). To illustrate geometrically, assume that (P) has just one inequality constraint $g(x) \leq 0$ and there are no explicit equality constraints. Then the resource-payoff set for the problem is the set of points defined by

$$\mathrm{RP} = \{(g(x), f(x)) : \ x \in S)\}.$$

That is, RP is a mapping of all $x \in S$ into the (g, f)-plane. In this plane, the Lagrangian equated to a constant θ has the form $f + ug = \theta$, which defines a line of slope $-u$ and intercept θ. For any $u \geq 0$, the dual function $\theta(u)$ is defined by minimizing $f(x) + ug(x)$ over $x \in S$. Thus $\theta\ (u)$ is the intercept of a linear support to the resource-payoff set at $\{(g(x), f(x)) : x \in S(u)\}$. To illustrate, consider the problem

$$\text{(P6)} \quad \begin{cases} \min & x_1^2 + x_2^2 \\ \text{s.t.} & 1 - x_1 - x_2 \leq 0, \\ & -x_1 \leq 0, \\ & -x_2 \leq 0. \end{cases}$$

The optimal solution is $x_1^* = x_2^* = 1/2$ and $f(x_1^*, x_2^*) = 1/2$. Letting $S = \{(x_1, x_2) : x_1 \geq 0, x_2 \geq 0\}$, and $g(x_1, x_2) = 1 - x_1 - x_2$, the resource-payoff set is a subset of $\mathbf{R}^2$ defined by $\mathrm{RP} = \{(g(x_1, x_2), f(x_1, x_2)) : x_1 \geq 0, x_2 \geq$



Lagrangian Duality: BASICS, Figure 2
RP set for (P6)

0}. It can be verified that RP consists of all points in $\mathbf{R}^2$ between the curves $(g-1)^2$ and $(g-1)^2/2$ for $g \leq 1$ as shown in Fig. 2.

The two linear supports of RP shown have slopes of -2 and -1, corresponding to u values of 2 and 1. With $u = 2$, $(x_1(u), x_2(u))$ is the singleton (1, 1) and $(g(x(u)), f(x(u))) = (-1, 2)$. The line with slope -2 passing through the point $(g, f) = (-1, 2)$ intersects the f-axis at the origin. Thus $\theta(2) = 0 < f(x^*)$, illustrating the weak duality theorem.

For $u = 1$, $(x_1(u), x_2(u)) = (1/2, 1/2)$ and $(g(x(u)), f(x(u))) = (0, 1/2)$. Since this point lies on the f-axis it follows that $\theta(1) = 1/2 = f(x^*)$. This illustrates the strong duality theorem.

As an alternative consider Example 5, the *binary linear programming* problem. Since S is discrete, RP consists of the eight points in $\mathbf{R}^2$ listed in the last two columns of Table 2. The optimal solution to the problem is $x^* = (1, 0, 1), f(x^*) = 12$. The resource-payoff set for this example is shown in Fig. 3. The lines in the figure trace out the lower envelope of the resource-payoff set and are found by minimizing the Lagrangian function over S using $u^1 = 7/4$, $u^2 = 5/3$ and $u^3 = 1$. The lines with slope $-7/4$, $-5/3$, and -1 intersect the f-axis at

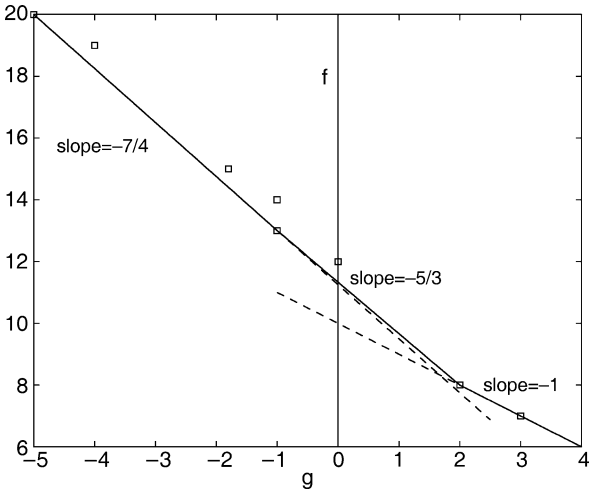
$$11\frac{1}{4}, 11\frac{1}{3}, 10,$$

respectively. Thus

$$\theta\left(\frac{7}{4}\right) = 11\frac{1}{4},\ \theta(1) = 10,\ \theta\left(\frac{5}{3}\right) = 11\frac{1}{3}.$$

Lagrangian Duality: BASICS, Table 2
Values of *g* and *f* for Example 5

x_1	x_2	x_3	$g(x_1, x_2, x_3)$ $x_1 + 3x_2 + 4x_3 - 5$	$f(x_1, x_2, x_3)$ $20 - x_1 - 5x_2 - 7x_3$
0	0	0	−5	20
0	0	1	−1	13
0	1	0	−2	15
0	1	1	2	8
1	0	0	−4	19
1	0	1	0	12
1	1	0	−1	14
1	1	1	3	7



Lagrangian Duality: BASICS, Figure 3
The set RP for Example 5

The duality gap for this problem, as noted earlier, is

$$f(x^*) - \theta(u^*) = 12 - 11\frac{1}{3} = \frac{2}{3}.$$

These two examples illustrate a sufficient condition for there to be no duality gap. There is no gap if the point $(g(x^*), f(x^*))$ lies on the *lower envelope* of the resource-payoff set, and there is a *linear support* of slope $-u \leq 0$ at that point with intercept $f(x^*)$.

This condition is satisfied for (P6), but not for (P5). However, if the constraint in (P5) is replaced by $g(x) = x_1 + 2x_2 + 4x_3 - 4 \leq 0$, the condition is satisfied. The effect of this constraint change can be seen in Table 2 and Fig. 3. In the table, the g column entries would be

increased by 1, and thus $f(x^*) = 13$, with $x^* = (x_1^*, x_2^*, x_3^*) = (0, 0, 1)$. In Fig. 3, the f-axis would be shifted one unit to be left. In this case, note that $(g(x^*), f(x^*))$ now lies on the lower envelope of the set RP. Furthermore, an optimal dual variable is any value $u^* \in [5/3, 7/4]$.

Gap Function

Another geometrical interpretation can be given for the primal problem

$$(\text{P7}) \quad \begin{cases} \min & f(x) \\ \text{s.t.} & Ax = b, \\ & x \geq 0, \end{cases}$$

where f is assumed convex and differentiable. In what follows, let $S = \{x \in \mathbf{R}^n : Ax = b, x \geq 0\}$ which is assumed to be a compact subset of $\mathbf{R}^n$.

For any feasible x, define the *gap function* by

$$\begin{aligned} G(x) &= \max_{y \in S} \nabla f(x)^\top (x - y) \\ &= -\min_{y \in S} \nabla f(x)^\top (y - x). \end{aligned}$$

The gap function has several interesting properties. Letting $y(x)$ be the solution of the linear program defining $G(x)$, note first that the gap function at x is the negative of the *directional derivative* of f at x in the direction $(y(x) - x)$. Second, it can be used to construct a lower bound on the optimal solution $f(x^*)$ of (P7). To see this, consider the convexity inequality

$$f(y) \geq f(x) + \nabla f(x)^\top (y - x), \quad \forall y \in S.$$

Minimizing both sides over $y \in S$ implies

$$f(x^*) \geq f(x) - G(x), \quad \forall x \in S.$$

By the weak duality result, a lower bound for $f(x^*)$ can also be obtained by evaluating the dual objective at any dual feasible solution. The next theorem employs the Wolfe dual of (P7) to show that the bound given above is equivalent to obtaining the maximum dual objective value for a given x.

Let v and u be the dual variables associated with $Ax = b$ and $x \geq 0$, respectively. The *Lagrangian function* of (P7) is $L(x, v, u) = f(x) + v^\intercal(b - Ax) - u^\intercal x$. Then, for the given x, the maximum dual objective value is

$$d(x) = \max_{(v,u) \in D(x)} L(x, v, u),$$

where $D(x)$ is the set of all multipliers such that (x, v, u) is feasible to the Wolfe dual:

$$D(x) = \{(v, u) : \nabla_x L(x, v, u) = 0 \text{ and } u \geq 0\}.$$

Theorem 10 *For any $x \in S$, $G(x) = f(x) - d(x)$.*

Proof First it is verified that $D(x)$ is nonempty so that $d(x)$ is well defined. This will be true if there exists a v such that $A^\intercal v \leq \nabla f(x)$. By adaptation of Farkas' lemma (cf. also ► Farkas lemma; ► Farkas lemma: Generalizations) such a v exists if and only if the alternative system

$$\nabla f(x)^\top z < 0, \quad Az = 0, \quad z \geq 0$$

has no solution. However $Az = 0$, $z \geq 0$ imply that $x + \lambda z \in S$ for all $\lambda \geq 0$. Since S is assumed to be compact, the only possibility is $z = 0$ and the alternative system has no solution. Thus $D(x)$ is nonempty.

The dual constraints imply that $u^\intercal x = \nabla f(x)^\intercal x - v^\intercal Ax$, so

$$d(x) = \begin{cases} \max\limits_{v} & f(x) - \nabla f(x)x + v^\top b \\ \text{s.t.} & A^\top v \leq \nabla f(x). \end{cases}$$

By linear programming duality

$$\begin{cases} \max & b^\top v \\ \text{s.t.} & A^\top v \leq \nabla f(x) \end{cases} = \begin{cases} \min & \nabla f(x)^\top y \\ \text{s.t.} & Ay = b \\ & y \geq 0, \end{cases}$$

and it follows that

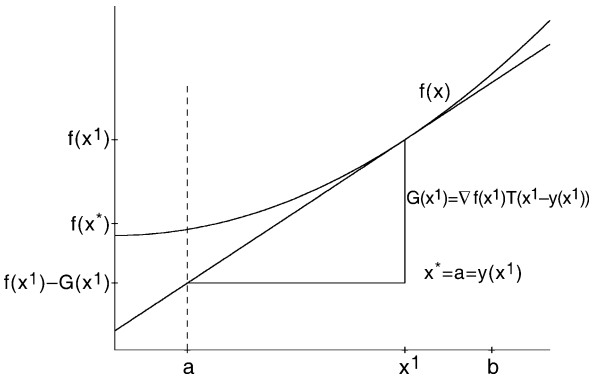
$$\begin{aligned} d(x) &= f(x) + \min_{y \in S} \nabla f(x)^\top (y - x) \\ &= f(x) - G(x). \end{aligned}$$

□

Expressing the duality gap in terms of x allows a simple interpretation of weak and strong duality in the convex case. Figure 4 illustrates the gap function in one variable with S being the interval $[a, b]$. Let $x = x^1$. The linear function

$$f(x^1) + \nabla f(x^1)^\top (y - x^1)$$

is the tangent line shown. It has a minimum in S at $y(x^1) = a$ which, by convexity, must lie below $f(x^*)$. Hence the weak duality result holds: $f(x^*) \geq f(x^1) - G(x^1) =$



Lagrangian Duality: BASICS, Figure 4
A one variable interpretation of weak and strong duality

$d(x^1)$. Strong duality occurs when $x^1 = x^*$ and the minimum of the linear function (i. e., the tangent at x^*) has the value $f(x^*)$. In this case $G(x^*) = 0$. If x^* were at an interior point of S, and/or if x^1 is infeasible to S, this same interpretation holds provided only that $f(x^1)$ and $\nabla f(x^1)$ are defined.

Summary

This section has illustrated basic results and geometrical interpretations of Lagrangian duality. The reference list below is a selection of texts and journal articles on this topic for further reading.

See also

- ► Equality-constrained Nonlinear Programming: KKT Necessary Optimality Conditions
- ► First Order Constraint Qualifications
- ► Inequality-constrained Nonlinear Optimization
- ► Kuhn–Tucker Optimality Conditions
- ► Rosen's Method, Global Convergence, and Powell's Conjecture
- ► Saddle Point Theory and Optimality Conditions
- ► Second Order Constraint Qualifications
- ► Second Order Optimality Conditions for Nonlinear Optimization

References

1. Balinski ML, Baumol WJ (1968) The dual in nonlinear programming and its economic interpretation. Rev Economic Stud 35:237–256
2. Bazaraa MS, Goode JJ (1979) A survey of various tactics for generating Lagrangian multipliers in the context of Lagrangian duality. Europ J Oper Res 3:322–338
3. Bertsekas DP (1975) Nondifferentiable optimization. North-Holland, Amsterdam
4. Bertsekas DP (1982) Constrained optimization and Lagrange multiplier methods. Acad. Press, New York
5. Bertsekas DP (1995) Nonlinear programming. Athena Sci., Belmont, MA
6. Brooks R, Geoffrion A (1966) Finding Everett's Lagrange multipliers by linear programming. Oper Res 16:1149–1152
7. Everett H (1973) Generalized Lagrange multiplier method for solving problems of optimum allocation of resources. Oper Res 4:72–97
8. Falk JE (1967) Lagrange multipliers and nonconvex programming. J Math Anal Appl 19:141–159
9. Fiacco AV, McCormick GP (1968) Nonlinear programming: Sequential unconstrained minimization techniques. Wiley, New York
10. Fisher ML, Northup WD, Shapiro JF (1975) Using duality to solve discrete optimization problems: theory and computational experience. In: Balinski ML, Wolfe P (eds) Nondifferentiable Optimization. North-Holland, Amsterdam
11. Fletcher R (ed) (1969) Optimization. Acad. Press, New York
12. Geoffrion AM (1970) Elements of large- scale mathematical programming I-II. Managem Sci 16:652–675; 676–691
13. Geoffrion AM (1971) Duality in nonlinear programming: A simplified application-oriented development. SIAM Rev 13:1–7
14. Hearn DW (1982) The gap function of a convex program. Oper Res Lett 1:67–71
15. Hearn DW, Lawphongpanich S (1989) Lagrangian dual ascent by generalized linear programming. Oper Res Lett 8:189–196
16. Hearn DW, Lawphongpanich S (1990) A dual ascent algorithm for traffic assignment problems. Transport Res B 248(6):423–430
17. Kiwiel KC (1985) Methods of descent for nondifferentiable optimization. Springer, Berlin
18. Lasdon LS (1970) Optimization theory for large systems. MacMillan, New York
19. Luenberger DG (1969) Optimization by vector space methods. Wiley, New York
20. Luenberger DG (1973) Introduction to linear and nonlinear programming. Addison-Wesley, Reading, MA
21. Mangasarian OL (1969) Nonlinear programming. McGraw-Hill, New York
22. Nemhauser GL, Wolsey LA (1988) Integer and combinatorial optimization. Wiley, New York
23. Powell MJD (1978) Algorithms for nonlinear constraints that use Lagrangian functions. Math Program 14:224–248
24. Rockafellar RT (1970) Convex analysis. Princeton Univ. Press, Princeton

25. Rockafellar RT (1975) Lagrange multipliers in optimization. In: Cottle RW, Lemke CE (eds) Nonlinear Programming, SIAM-AMS Proc. vol IX, pp 23–24
26. Whittle P (1971) Optimization under constraints. Wiley, New York
27. Wolfe P (1961) A duality theorem for nonlinear programming. Quart Appl Math 19:239–244
28. Zangwill WI (1969) Nonlinear programming: A unified approach. Prentice-Hall, Englewood Cliffs, NJ

Lagrangian Multipliers Methods for Convex Programming

LMM

MARC TEBOULLE
School Math. Sci., Tel-Aviv University,
Ramat-Aviv, Tel-Aviv, Israel

MSC2000: 90C25, 90C30

Article Outline

Keywords

Augmented Lagrangians; Convex optimization; Lagrangian multipliers; Primal-dual methods; Proximal algorithms

Optimization problems concern the minimization or maximization of functions over some set of conditions called constraints. The original treatment of constrained optimization problems was to deal only with equality constraints via the introduction of *Lagrange multipliers* which found their origin in basic mechanics. Modeling real world situations often requires using inequality constraints leading to more challenging optimization problems. Lagrange multipliers are used in optimality conditions and play a key role to devise algorithms for constrained problems. What will be summarized here are the basic elements of various algorithms based on Lagrangian multipliers to solve constrained optimization problems, and particularly convex optimization problems. A standard formulation of an optimization problem is:

$$(\mathrm{O}) \quad \min\{f(x):\ x \in X \cap C\},$$

where X is a certain subset of $\mathbf{R}^n$ and C is the set of constraints described by equality and inequality constraints

$$C = \left\{ x \in \mathbb{R}^n : \begin{array}{c} g_i(x) \le 0,\ i = 1, \ldots, m, \\ h_i(x) = 0,\ i = 1, \ldots, p \end{array} \right\}.$$

All the functions in problem (O) are real valued functions on $\mathbf{R}^n$, and the set X can described more abstract constraints of the problem. A point $x \in X \cap C$ is called a *feasible solution* of the problem, and an *optimal solution* is any feasible point where the local or global minimum of f relative to $X \cap C$ is actually attained. By a *convex problem* we mean the case where X is a convex set, the functions $f, g_1, \ldots, g_m$ are convex and $h_1, \ldots, h_p$ are affine. Recall that a set $S \subset \mathbf{R}^n$ is *convex* if the line segment joining any two different points of S is contained in it.

Let S be a convex subset of $\mathbf{R}^n$. A real valued function $f\colon S \to \mathbf{R}$ is convex if for any $x, y \in S$ and any $\lambda \in [0, 1]$,

$$f(\lambda x + (1-\lambda)y) \le \lambda f(x) + (1-\lambda) f(y).$$

Convexity plays a fundamental role in optimization (even in nonconvex problems). One of the key fact is that when a convex function is minimized over a convex set, every local optimal solution is global. Another, fundamental point is that a powerful duality theory can be developed for convex problems, which as we shall see, is also at the root of the development and analysis of Lagrangian multiplier methods.

Augmented Lagrangians

The basic idea of augmented Lagrangian methods for solving constrained optimization problems, also called *multiplier methods*, is to transform a constrained problem into a sequence of unconstrained problems. The approach differs from the penalty-barrier methods, [13] from the fact that in the functional defining the unconstrained problem to be solved, in addition to a penalty parameter, there are also multipliers associated with the

constraints. Multiplier methods can be seen as a combination of penalty and dual methods. The motivation for these methods came from the desire of avoiding ill-conditioning associated with the usual penalty-barrier methods. Indeed, in contrast to penalty methods, the penalty parameter need not to go to infinity to achieve convergence of the multiplier methods. As a consequence, the augmented Lagrangian has a 'good' conditioning, and the methods are robust for solving nonlinear programs. Augmented Lagrangians methods were proposed independently by M.R. Hestenes [16] and M.J.D. Powell [26] for the case of equality constraints, and extended for the case of inequality constraints by R.T. Rockafellar [27]. Many other researchers have contributed to the development of augmented Lagrangian methods, and for an excellent treatment and comprehensive study of multiplier methods, see [7] and references therein.

Quadratic Lagrangian

We start by briefly describing the basic steps involved in generating a multiplier method for the equality constrained problem

$$\text{(E)} \quad \min\{f(x):\ h_i(x) = 0,\ i = 1, \ldots, p\}.$$

Here f and h_i are real valued functions on $\mathbf{R}^n$ and no convexity is assumed (which will not help anyway because of the nonlinear equality constraints). Also for simplicity we let $X = \mathbf{R}^n$. The ordinary Lagrangian associated with (E) is

$$l(x, y) = f(x) + \sum_{i=1}^{p} y_i h_i(x).$$

One of the oldest and simplest way to solve (E) is by sequential minimization of the Lagrangian ([2]). Namely, we start with an initial multiplier y_k and minimize $l(x, y^k)$ over $x \in \mathbf{R}^n$ to produce x^k. We then update the multiplier sequence via the formula:

$$y_i^{k+1} = y_i^k + s^k h_i(x^k), \quad i = 1, \ldots, p,$$

where s_k is a stepsize parameter. The rational behind the above method is that it can be simply interpreted as a gradient-type algorithm to solve an associated dual problem. Unfortunately, such a method while simple requires too many assumptions on the problem's data to generate points converging rapidly toward an optimal solution. Thus this *primal-dual framework* is not in general particularly attractive. However, combining the primal-dual idea to the one of penalty leads to another class of algorithms called *multiplier methods*. In these methods one uses instead of the classical Lagrangian $l(x, y)$ a 'penalized' Lagrangian of the form:

$$P_c(x, y) = f(x) + \sum_{i=1}^{p} y_i h_i(x) + \frac{c}{2} \sum_{i=1}^{p} h_i^2(x),$$

where $c > 0$ is a penalty parameter. Then, starting with an initial multiplier y^k and penalty parameter c^k, the augmented Lagrangian P_c is minimized with respect to x and at the end of each minimization, the multipliers (and sometimes also the penalty parameter) are updated according to some scheme and we continue the process until convergence. More precisely, the method of multipliers generates the sequences $\{y^k\} \subset \mathbf{R}^m$, $\{x^k\} \subset \mathbf{R}^n$ as follows. Given a sequence of nondecreasing scalars $c_k > 0$, compute

$$x^{k+1} \in \arg\min\left\{L_{c_k}(x, y^k):\ x \in \mathbb{R}^n\right\},$$
$$y_i^{k+1} = y_i^k + c_k h_i(x^{k+1}), \quad i = 1, \ldots, p.$$

The rational behind the updating of the multipliers y^k is that if the generated sequence x^k converges to a local minimum then the sequence y^k will converge to the corresponding Lagrange multiplier y^*. Under reasonable assumptions, this happens without increasing the parameter c^k to infinity and thus avoids the difficulty with ill-conditioning. The above scheme provides with the key steps in devising a multiplier method for equality constrained optimization problems. We now turn to the case of problems with inequality constraints:

$$\text{(I)} \quad \min\{f(x):\ g_i(x) \le 0,\ i = 1, \ldots, m\}.$$

One simple way to treat this case is to transform the inequality constraints to equality using squared variables and then apply the multiplier framework previously outlined. Thus, we convert problem (I) to the equality constrained problem in the variables (x, z):

$$\begin{cases} \min & f(x) \\ \text{s.t.} & g_i(x) + z_i^2 = 0, \quad i = 1, \ldots, m, \end{cases}$$

where $z \in \mathbf{R}^m$ are additional variables. The quadratic augmented Lagrangian to be minimized with respect to

(x, z) thus takes the form:

$$Q_c(x, z, y) = f(x) + \sum_{i=1}^{m} y_i(g_i(x) + z_i^2) + \frac{c}{2}\sum_{i=1}^{m}(g_i(x) + z_i^2)^2.$$

The key observation here is that the minimization with respect to z can be carried out analytically. One can verify via simple calculus that for fixed (x, y), $\min_{z\in\mathbb{R}^m} Q_c(x, z, y) = L_c(x, y)$, with

$$L_c(x, y) = f(x) + \frac{1}{2c}\sum_{i=1}^{m}\left[\max^2\{0, y_i + cg_i(x)\} - y_i^2\right].$$

Summarizing, the multiplier method for the inequality constrained problem (I) consists of the following two steps:

$$x^{k+1} \in \arg\min\left\{L_{c_k}(x, y^k):\ x \in \mathbb{R}^n\right\},$$
$$y^{k+1} = \max\{0, y^k + c_k g(x^{k+1})\}.$$

For the general optimization problem (O), namely the case of mixed equality and inequality constraints, Lagrangian multiplier methods can be developed in a similar fashion. Convergence results to a local minimum for the above schemes can be established under second order sufficiency assumptions, ([7,28]). In the case of convex programs, namely when in problem (I) the functions f, $g_1, \ldots, g_m$ are assumed convex functions, (or more generally in problem (O), if we also assume h_i affine and X convex), much stronger convergence results can be established under mild assumptions ([29]). A typical result is as follows.

Assumption 1 The set of optimal solutions of the convex problem (I) is nonempty and compact and the set of multiplier is nonempty and compact.

The assumption on the optimal set of multipliers is guaranteed under the standard *Slater constraint qualification*:

$$\exists\hat{x}:\ g_i(\hat{x}) \le 0, \quad i = 1, \ldots, m.$$

Under assumption 1, one can prove that the sequence y^k converges to some Lagrange multiplier y^* and any limit point of the sequence x^k is an optimal solution of the convex program. Note that we do not require that c_k is sufficiently large and convergence is obtained from any starting point $y^0 \in \mathbf{R}^m$.

The multiplier method for inequality constrained problems was derived by using slack variables in the inequality constraints and then by applying the multiplier method which was originally devised for problems having only equality constraints. An alternative way of constructing an augmented Lagrangian method is via the *proximal framework*.

Proximal Minimization

Consider the convex optimization problem

$$\text{(C)} \quad \min\{F(x):\ x \in \mathbb{R}^n\},$$

where F: $\mathbf{R}^n \to (-\infty, +\infty]$ is a proper, lower semicontinuous convex function. One method to solve (C) is to 'regularize' the objective function using the *proximal map* of J.-J. Moreau [22]. Given a real positive number c, a *proximal approximation* of f is defined by:

$$F_c(x) = \inf_u\left\{F(u) + (2c)^{-1}\|x - u\|^2\right\}. \tag{1}$$

The resulting function F_c enjoys several important properties: it is convex and differentiable with gradient which is Lipschitz with constant (c^{-1}) and when minimized possesses the same set of minimizers and the same optimal value than problem (C). The quadratic regularization process of the function f leads to an iterative procedure for solving problem (C), called the *proximal point algorithm* [21,30]. The method is as follows: given an initial point $x_0 \in \mathbf{R}^n$ a sequence $\{x_k\}$ is generated by solving:

$$x^{k+1} = \arg\min\left\{F(x) + \frac{1}{2c_k}\left\|x - x^k\right\|^2\right\}, \tag{2}$$

where $\{c_k\}_{k=1}^{\infty}$ is a sequence of positive numbers.

One of the most powerful application of the proximal algorithm is when applied to the dual of an optimization problem. Indeed, as shown by Rockafellar [27,29], a direct calculation shows that L_c can be written as

$$L_c(x, y) = \max_{\lambda\in\mathbb{R}_+^m}\left\{l(x, \lambda) - \frac{1}{2c}\|\lambda - y\|^2\right\}, \tag{3}$$

where the maximum is attained uniquely at $\lambda_i = \max\{0, y_i + c\, g_i(x)\}$, $i = 1, \ldots, m$. Here l: $\mathbf{R}^n \times \mathbf{R}^m_+ \to \mathbf{R}$ denotes the usual Lagrangian associated with the inequality constrained problem (I) and $\mathbf{R}^m_+$ stands for the nonnegative orthant. This shows that the quadratic augmented Lagrangian is nothing else but the Moreau proximal regularization of the ordinary Lagrangian, and the quadratic multiplier method can be interpreted as applying the proximal minimization algorithm on the dual problem associated with (I):

$$\text{(D)} \quad \sup\{d(y):\ y \geq 0\},$$

where $d(y) := \inf_x l(x, y)$ is the dual functional. This interplay between the proximal algorithm and multiplier methods is particularly interesting since it offers the possibility of designing and analyzing the convergence properties of the later from the former, and also leads to consider useful potential extensions of multiplier methods which are discussed next.

Modified Lagrangians

One of the main disadvantages of the quadratic multiplier methods for inequality constrained problems is that even when the original problem is given twice continuously differentiable, the corresponding functional L_c is not. Indeed, note that with twice continuously differentiable data $\{f, g_i\}$, the augmented Lagrangian L_c is continuously differentiable in x. However, the Hessian matrix of L_c is discontinuous for all x such that $g_i(x) = -c^{-1} y_i$. This may cause difficulties in designing an efficient unconstrained minimization algorithm for L_c and motivates the search for alternative augmented Lagrangian to handle inequality constrained problems, which we call here *modified Lagrangians*. These Lagrangians possess better differentiability properties to allow the use of efficient Newton-like methods in the minimization step. Modified Lagrangians can be found in several works, [1,15,19,20]. An approach originally developed in [19] proposed a class of methods which uses instead of L_c a modified Lagrangian of the form:

$$B_c(x, y) := f(x) + c^{-1} \sum_{i=1}^{m} y_i \psi(c g_i(x)),$$

where ψ is a scalar penalty function which is at least C^2 and satisfies some other technical conditions. For each choice of ψ we then have a multiplier method which consists of the sequence of unconstrained minimization problems

$$x^{k+1} \in \arg\min_{x \in \mathbb{R}^n} B_{c^k}(x, y^k),$$

followed by the multiplier updates

$$y_i^{k+1} = y_i^k \psi'(c_k g_i(x^{k+1})), \quad i = 1, \ldots, m.$$

The multiplier updating formula can be simply explained as follows. Suppose the functions in problem (I) are given differentiable, then x^{k+1} minimizes $B_{c_k}(x, y^k)$ means that $\nabla_x B_{c_k}(x^{k+1}, y^k) = 0$, i. e.,

$$\nabla f(x^{k+1}) + \sum_{i=1}^{m} y_i^k \psi'(c_k g_i(x^{k+1})) \nabla g_i(x^{k+1}) = 0,$$

and using the multiplier updates defined above the equation reduces to:

$$\nabla f(x^{k+1}) + \sum_{i=1}^{m} y_i^{k+1} \nabla g_i(x^{k+1}) = 0,$$

showing that (x^{k+1}, y^{k+1}) also satisfies the optimality conditions for minimizing the classical Lagrangian, namely $\nabla_x l(x^{k+1}, y^{k+1}) = 0$. Interesting special cases of the generic method described above includes the exponential method ([23,35]) with the choice $\psi(t) = e^t - 1$ and the modified barrier method [24] which is based on the choice $\psi(t) = -\ln(1 - t)$. More examples and further analysis of these methods can be found in [25].

Another way of constructing modified Lagrangians is in view of the results from the previous section, to try alternative proximal regularization terms which could lead to better differentiability properties of the corresponding augmented Lagrangian functional. This approach was considered in [32], who suggested new classes of proximal approximation of a function given by

$$F_\lambda(x) := \inf_u \{f(u) + \lambda^{-1} D(u, x)\}. \tag{4}$$

Here, $D(\cdot, \cdot)$, which replaces the quadratic proximal term in (1), is a measure of 'closeness' between x, y satisfying $D(x, y) \geq 0$ with equality if and only if $x = y$. One generic form for D is the use of a 'proximal-like'

term defined by

$$D(x, y) := d_\varphi(x, y) := \sum_{i=1}^{n} y_i \varphi(y_i^{-1} x_i),$$

where φ is a given convex function defined on the nonnegative real line and which satisfies some technical conditions ([33]). The motivation of using such functional emerges from the desire of eliminating nonnegativity constraints such as the ones present in the dual problem. Thus, by mimicking (2) and (3) with the proximal term d_φ, one can design a wide variety of modified Lagrangians methods with an appropriate choice of φ. The basic steps of the modified multipliers method then emerging can be described as follows: Given a sequence of positive numbers $\{c_k\}$, and initial points $x^k \in \mathbf{R}^n$, $y^k \in \mathbf{R}^m_+$ (the positive orthant) generate iteratively the next points by solving

$$x^{k+1} \in \arg\min \left\{ M_{c_k}(x, y^k) : \; x \in \mathbb{R}^n \right\}, \tag{5}$$

followed by the multiplier updates

$$y^{k+1} \in \arg\max_{y \geq 0} \{ y' g(x^{k+1}) - c_k^{-1} d_\varphi(y, y^k) \}, \tag{6}$$

where M_c is the modified Lagrangian defined by

$$M_c(x, y) = \sup_{\mu \in \mathbb{R}^m_+} \{ l(x, \mu) - c^{-1} d_\varphi(\mu, y) \} \tag{7}$$

i. e., the proximal-like regularization of the usual Lagrangian $l(x, \mu)$ associated with problem (I). In the equation (6), $g(x)$ denotes the column vector $(g_1(x), \ldots, g_m(x))' \in \mathbf{R}^m$ and the prime denotes transposition. The method is viable since both (6) and (7) can be solved analytically, and the computational analysis and effort should concentrate on (5). This method of multipliers is nothing else but a proximal-like algorithm applied to the dual problem (D) ([17]) i. e., starting with $y^0 \in \mathbf{R}^m_+$, generate a sequence $\{y^k\}$ by solving

$$y^{k+1} = \arg\max_{y \geq 0} \{ d(y) - c_k^{-1} d_\varphi(y, y^k) \}.$$

The above scheme gives rise to a rich family of numerical methods, which includes (with an appropriate choice of φ) several classes of nonquadratic multiplier methods ([7,24,35]). One of the main advantage of using these modified multiplier methods is that in contrast with the usual quadratic augmented Lagrangian function, the modified Lagrangian for various choices of d_φ is twice continuously differentiable if the problem's data f, g are. Thus, this opens the possibility of using Newton methods for solving efficiently (5).

Under assumption 1 and appropriate condition on the kernel φ one can prove convergence results for these modified multiplier methods similar to the one obtains in the quadratic case ([17]). There has been considerable recent research on modified Lagrangian methods and for further results see [3,4,5,11,18,25].

The Lagrangian functional plays a central role in the analysis and algorithmic development of constrained optimization problems. Lagrangian based methods and the related proximal framework have been used in other optimization contexts, such as convexification of nonconvex optimization problems [6,28], decomposition algorithms [9,12,31,34], semidefinite programming [10] and in many other applications, see e. g., [8,14] where more references can be found.

See also

- Convex Max-functions
- Decomposition Techniques for MILP: Lagrangian Relaxation
- Integer Programming: Lagrangian Relaxation
- Lagrange, Joseph-Louis
- Multi-objective Optimization: Lagrange Duality

References

1. Arrow KJ, Gould FJ, Howe SM (1973) A general saddle point result for constrained optimization. Math Program 5:225–234
2. Arrow KJ, Hurwicz L, Uzawa H (1958) Studies in linear and nonlinear programming. Stanford Univ. Press, Palo Alto, CA
3. Auslender AA, Cominetti R, Haddou M (1997) Asymptotic analysis of penalty and barrier methods in convex and linear programming. Math Oper Res 22:43–62
4. Auslender AA, Teboulle M, Ben-Tiba S (1999) Interior proximal and multiplier methods based on second order homogeneous kernels. Math Oper Res 24:645–668
5. Ben-Tal A, Zibulevsky M (1997) Penalty-barrier methods for convex programming problems. SIAM J Optim 7:347–366
6. Bertsekas D (1979) Convexification procedures and decomposition methods for nonconvex optimization problems. J Optim Th Appl 29:169–197
7. Bertsekas D (1982) Constrained optimization and Lagrangian multipliers. Acad. Press, New York

8. Bertsekas D, Tsitsiklis JN (1989) Parallel and distributed computation: Numerical methods. Prentice-Hall, Englewood Cliffs, NJ
9. Chen G, Teboulle M (1994) A proximal-based decomposition method for convex minimization problems. Math Program 64:81–101
10. Doljanski M, Teboulle M (1998) An interior proximal algorithm and the exponential multiplier method for semidefinite programming. SIAM J Optim 9:1–13
11. Eckstein J (1993) Nonlinear proximal point algorithms using Bregman functions with applications to convex programming. Math Oper Res 18:202–226
12. Eckstein J, Bertsekas. DP (1992) On the Douglas–Rachford splitting method and the proximal point algorithm for maximal monotone operators. Math Program 55:293–318
13. Fiacco AV, McCormick GP (1990) Nonlinear programming: Sequential unconstrained minimization techniques. Classics Appl Math. SIAM, Philadelphia
14. Glowinski R, Le Tallec P (1989) Augmented Lagrangians and operator-splitting methods in nonlinear mechanics. Stud Appl Math. SIAM, Philadelphia
15. Golshtein EG, Tretyakov NV (1996) Modified Lagrangians and monotone maps in optimization. Discrete Math and Optim. Wiley, New York
16. Hestenes MR (1969) Multiplier and gradient methods. J Optim Th Appl 4:303–320
17. Iusem A, Teboulle M (1995) Convergence analysis of nonquadratic proximal methods for convex and linear programming. Math Oper Res 20:657–677
18. Kiwiel KC (1997) Proximal minimization methods with generalized Bregman functions. SIAM J Control Optim 35:1142–1168
19. Kort KBW, Bertsekas DP (1972) A new penalty function method for constrained minimization. In: Proc. IEEE Conf. Decison Control, 162–166
20. Mangasarian OL (1975) Unconstrained Lagrangians in nonlinear programming. SIAM J Control 13:772–791
21. Martinet B (1978) Perturbation des méthodes D, optimisation application. RAIRO Anal Numer/Numer Anal 93(12):152–171
22. Moreau JJ (1965) Proximité and dualité dans un espace Hilbertien. Bull Soc Math France 93:273–299
23. Nguyen VH, Strodiot JJ (1979) On the convergence rate of a penalty function method of the exponential type. J Optim Th Appl 27:495–508
24. Polyak RA (1992) Modified barrier functions: Theory and methods. Math Program 54:177–222
25. Polyak RA, Teboulle M (1997) Nonlinear rescaling and proximal-like methods in convex optimization. Math Program 76:265–284
26. Powell MJD (1969) A method for nonlinear constraints in minimization problems. In: Fletcher R (ed) Optimization. Acad. Press, New York, 283–298
27. Rockafellar RT (1973) A dual approach to solving nonlinear programming problems by unconstrained optimization. Math Program 5:354–373
28. Rockafellar RT (1974) Augmented Lagrange multiplier functions and duality in nonconvex programming. SIAM J Control 12:268–285
29. Rockafellar RT (1976) Augmented Lagrangians and applications of the proximal point algorithm in convex programming. Math Oper Res 1:97–116
30. Rockafellar RT (1976) Monotone operators and the proximal point algorithm. SIAM J Control Optim 14:877–898
31. Spingarn JE (1985) Applications of the method of partial inverses to convex programming: Decomposition. Math Program 32:199–223
32. Teboulle M (1992) Entropic proximal mappings in nonlinear programming and applications. Math Oper Res 17:670–690
33. Teboulle M (1997) Convergence of proximal-like algorithms. SIAM J Optim 7:1069–1083
34. Tseng P (1991) Applications of a splitting algorithm to decomposition in convex programming and variational inequalities. SIAM J Control Optim 29:119–138
35. Tseng P, Bertsekas DP (1993) On the convergence of the exponential multiplier method for convex programming. Math Program 60:1–19

Laplace Method and Applications to Optimization Problems

PANOS PARPAS, BERÇ RUSTEM
Department of Computing, Imperial College, London, GB

Article Outline

Abstract

The Laplace method has found many applications in the theoretical and applied study of optimization problems. It has been used to study: the asymptotic behavior of stochastic algorithms, 'phase transitions' in combinatorial optimization, and as a smoothing technique

for non–differentiable minimax problems. This article describes the theoretical foundation and practical applications of this useful technique.

Background

Laplace's method is based on an ingenious trick used by Laplace in one his papers [19]. The technique is most frequently used to perform asymptotic evaluations to integrals that depend on a scalar parameter t, as t tends to infinity. Its use can be theoretically justified for integrals in the following form:

$$I(t) = \int_{\mathcal{A}} \exp\left\{\frac{-f(x)}{T(t)}\right\} \mathrm{d}\Lambda(x) .$$

Where $f : \mathbb{R}^n \to \mathbb{R}$, $T : \mathbb{R} \to \mathbb{R}$, are assumed to be smooth, and $T(t) \to 0$ as t tends to ∞. $\mathcal{A}$ is some compact set, and Λ is some measure on $\mathcal{B}$ (the $\sigma-$field generated by $\mathcal{A}$). We know that since $\mathcal{A}$ is compact, the continuous function f will have a global minimum in $\mathcal{A}$. For simplicity, assume that the global minimum x^* is unique, and that it occurs in the interior $\mathcal{A}$. Under these conditions, and as t tends to infinity, only points that are in the immediate neighborhood of x^* contribute to the asymptotic expansion of $I(t)$ for large t. The heuristic argument presented above can be made precise. The complete argument can be found in [2], and in [4]. Instead we give a heuristic but didactic argument that is usually used when introducing the method.

Heuristic Foundations of the Method

For the purpose of this subsection only, assume that f is a function of one variable, and that $\mathcal{A}$ is given by some interval $[a, b]$. It will be instructive to give a justification of the method based on the one dimensional integral:

$$K(t) = \int_a^b \exp\left\{-\frac{f(x)}{t}\right\} \mathrm{d}x .$$

Suppose that f has a unique global minimum, say c, such that $c \in (a, b)$. As t is assumed to be large, we only need to take into account points near c when evaluating $K(t)$. We therefore approximate $K(t)$ by $K(t;\epsilon)$. The latter quantity is given by:

$$K(t;\epsilon) = \int_{c-\epsilon}^{c+\epsilon} \exp\left\{-\frac{f(x)}{t}\right\} \mathrm{d}x .$$

Expanding f to second order, and by noting that $f'(c) = 0$, we obtain the following approximation:

$$\begin{aligned} K(t;\epsilon) &\approx \int_{c-\epsilon}^{c+\epsilon} \exp\left\{-\frac{f(c) + \frac{1}{2}f''(c)(x-c)^2}{t}\right\} \mathrm{d}x \\ &= \exp\left\{-\frac{f(c)}{t}\right\} \int_{c-\epsilon}^{c+\epsilon} \exp\left\{-\frac{f''(c)(x-c)^2}{2t}\right\} \mathrm{d}x . \end{aligned}$$

The limits of the integral above can be extended to infinity. This extension can be justified by the fact only points around c contribute to the asymptotic evaluation of the integral.

$$\begin{aligned} &K(t;\epsilon) \\ &\approx \exp\left\{-\frac{f(c)}{t}\right\} \int_{-\infty}^{+\infty} \exp\left\{-\frac{f''(c)(x-c)^2}{2t}\right\} \mathrm{d}x \\ &= \exp\left\{-\frac{f(c)}{t}\right\} \sqrt{\frac{2\pi t}{f''(c)}} . \end{aligned}$$

In conclusion we have that:

$$\lim_{t\to\infty} K(t) = \exp\left\{-\frac{f(c)}{t}\right\} \sqrt{\frac{2\pi t}{f''(c)}} .$$

Rigorous justifications of the above arguments can be found in [4]. These types of results are standard in the field of asymptotic analysis. The same ideas can be applied to optimization problems.

Applications

Consider the following problem:

$$\begin{aligned} F^* &= \min f(x) \\ &\quad \text{s.t } g_i(x) \le 0 \qquad i = 1, \ldots, l . \end{aligned} \tag{1}$$

Let S denote the feasible region of the problem above, and assume that it is nonempty, and compact, then:

$$\lim_{t\downarrow 0} -\epsilon \ln c(t) = F^* . \tag{2}$$

Where,

$$\begin{aligned} c(t) &\triangleq \int_S \exp\left\{\frac{-f(x)}{t}\right\} \mathrm{d}\Lambda \\ &= \int_{\mathbb{R}^n} \exp\left\{\frac{-f(x)}{t}\right\} I_x(S)\mathrm{d}\Lambda . \end{aligned} \tag{3}$$

Λ is any measure on $(\mathbb{R}^n, \mathcal{B})$. A proof of Eq. (2) can be found in [16].

The relationship in Eq. (3) can be evaluated using the Laplace method. The link between the Laplace method and optimization has been explored in:

- Stochastic methods for global optimization.
- Phase transitions in combinatorial optimization.
- Algorithms for worst case analysis.

These application areas will be explored next.

Stochastic Methods for Global Optimization

Global optimization is concerned with the computation of global solutions of Eq. (1). In other words, one seeks to compute F^*, and if possible obtaining points from the following set:

$$S^* = \{x \in S \mid f(x) = F^*\} \ .$$

Often the only way to solve such problems is by using a stochastic method. Deterministic methods are also available but are usually applicable to low dimensional problems. When designing stochastic methods for global optimization, it is often the case that the algorithm can be analyzed as a stochastic process. Then in order to analyze the behavior of the algorithm we can examine the asymptotic behavior of the stochastic process. In order to perform this analysis we need to define a probability measure that has its support in S^*. This strategy has been implemented in [3,6,7,8,9,10,16].

A well known method for obtaining a solution to an unconstrained optimization problem is to consider the following Ordinary Differential Equation (ODE):

$$\mathrm{d}X(t) = -\nabla f(X(t))\mathrm{d}t \ . \tag{4}$$

By studying the behavior of $X(t)$ for large t, it can be shown that $X(t)$ will eventually converge to a stationary point of the unconstrained problem. A review of, so called, continuous-path methods can be found in [22]. More recently, application of this method to large scale problems was considered by Li-Zhi et al. [13]. A deficiency of using Eq. (4) to solve optimization problems is that it will get trapped in local minima. In order to allow the trajectory to escape from local minima, it has been proposed by various authors (e. g. [1,3,7,8,12,16]) to add a stochastic term that would allow the trajectory to "climb" hills. One possible augmentation to Eq. (4) that would enable us to escape from local minima is to add noise. One then considers the *diffusion process*:

$$\mathrm{d}X(t) = -\nabla f(X(t))\mathrm{d}t + \sqrt{2T(t)}\mathrm{d}B(t) \ . \tag{5}$$

Where $B(t)$ is the standard Brownian motion in $\mathbb{R}^n$. It has been shown in [3,7,8], under appropriate conditions on f, that if the *annealing schedule* is chosen as follows:

$$T(t) \triangleq \frac{c}{\log(2+t)} \ , \quad \text{for some} \quad c \geq c_0 \ , \tag{6}$$

where c_0 is a constant positive scalar (the exact value of c_0 is problem dependent). Under these conditions, as $t \to \infty$, the transition probability of $X(t)$ converges (weakly) to a probability measure Π. The latter, has its support on the set of global minimizers. A characterization of Π was given by Hwang in [11]. It was shown that Π is the weak limit of the following, so called, *Boltzmann density*:

$$p(t,x) = \left[\exp\left\{-\frac{f(x)}{T(t)}\right\}\right]\left[\int_{\mathbb{R}^n} \exp\left\{-\frac{f(x)}{T(t)}\right\} \mathrm{d}x\right]^{-1} . \tag{7}$$

Discussion of the conditions for the existence of Π, can be found in [11]. A description of Π in terms of the Hessian of f can also be found in [11]. Extensions of these results to constrained optimization problems appear in [16].

Phase Transitions in Combinatorial Optimization

The aim in combinatorial optimization is to select from a finite set of configurations of the system, the one that minimizes an objective function. The most famous combinatorial problem is the Travelling Salesman Problem (TSP). A large part of theoretical computer science is concerned with estimating the complexity of combinatorial problems. Loosely speaking, the aim of computational complexity theory is to classify problems in terms of their degree of difficulty. One measure of complexity is time complexity, and worst case time complexity has been the aspect that received most attention. We refer the interested reader to [15] for results in this direction. We will briefly summarize results that have to do with average time complexity, the Laplace method, and phase transitions.

Most of complexity theory is concerned with worst case complexity. However, many useful methods (e. g. the simplex method) will require an exponential amount of time to converge only in pathological cases. It is therefore of great interest to estimate average case complexity. The physics community has recently proposed the use of tools from statistical mechanics as one way of estimating average case complexity. A review in the form of a tutorial can be found in [14]. Here we just briefly adumbrate the main ideas.

The first step in the statistical mechanics approach is to define a probability measure on the configuration of the system. This definition is done with the Boltzmann density:

$$p_t(C) = \frac{\exp\left\{-\frac{1}{t}f(C)\right\}}{\sum\limits_C \exp\left\{-\frac{1}{t}f(C)\right\}} .$$

The preceding equation is of course the discrete version of Eq. (7). Using the above definition, the average value of the objective function is given by:

$$\langle f_t \rangle = \sum_C p_t(C) f(C) .$$

Tools and techniques of statistical mechanics can be used to calculate 'computational phase transitions'. A computational phase transition is an abrupt change in the computational effort required to solve a combinatorial optimization problem. It is beyond the scope of this article to elaborate on this interesting area of optimization. We refer the interested reader to the review in [14]. The book of Talagrand [20] presents some rigorous results on this subject.

Worst Case Optimization

In many areas where optimization methods can be fruitfully applied, worst case analysis can provide considerable insight into the decision process. The fundamental tool for worst case analysis is the continuous minimax problem:

$$\min_{x \in X} \Phi(x) ,$$

where $\Phi(x) = \max_{y \in Y} f(x, y)$. The continuous minimax problem arises in numerous disciplines, including n–person games, finance, economics and policy optimization (see [18] for a review). In general, they are used by the decision maker to assess the worst-case strategy of the opponent and compute the optimal response. The opponent can also be interpreted as nature choosing the worst-case value of the uncertainty, and the solution would be the strategy which ensures the optimal response to the worst–case. Neither the robust decision maker nor the opponent would benefit by deviating unilaterally from this strategy. The solution can be characterized as a saddle point when $f(x, \cdot)$ is convex in x and $f(\cdot, y)$ is concave in y. A survey of algorithms for computing saddle points can be found in [5,18].

Evaluating $\Phi(x)$ is extremely difficult due to the fact that global optimization is required over Y. Moreover, this function will in general be non-differentiable. For this reason, it has been suggested by many researchers (e. g. [17,21]) to approximate $\Phi(x)$ with $\Phi(x; t)$ given by:

$$\Phi(x; t) = \int_Y \exp\left\{-\frac{f(x, y)}{t}\right\} \mathrm{d}y .$$

This is of course another application of the Laplace method, and it can easily be seen that:

$$\lim_{t \downarrow 0} -t \ln \Phi(x; t) = \Phi(x) .$$

This idea has been implemented in [17,21] with considerable success.

References

1. Aluffi-Pentini F, Parisi V, Zirilli F (1985) Global optimization and stochastic differential equations. J Optim Theory Appl 47(1):1–16
2. Bender CM, Orszag SA (1999) Advanced mathematical methods for scientists and engineers I. Asymptotic methods and perturbation theory, Reprint of the 1978 original. Springer, New York
3. Chiang TS, Hwang CR, Sheu SJ (1987) Diffusion for global optimization in $\mathbf{R}^n$. SIAM J Control Optim 25(3):737–753
4. de Bruijn NG (1981) Asymptotic methods in analysis, 3rd edn. Dover Publications Inc., New York
5. Dem'yanov VF, Malozëmov VN (1990) Introduction to minimax. Translated from the Russian by Louvish D, Reprint of the 1974 edn. Dover Publications Inc., New York
6. Gelfand SB, Mitter SK (1991) Recursive stochastic algorithms for global optimization in $\mathbf{R}^d$. SIAM J Control Optim 29(5):999–1018
7. Geman S, Hwang CR (1986) Diffusions for global optimization. SIAM J Control Optim 24(5):1031–1043
8. Gidas B (1986) The Langevin equation as a global minimization algorithm. In: Disordered systems and biological organization (Les Houches 1985). NATO Adv Sci Inst Ser F Comput Systems Sci, vol 20. Springer, Berlin, pp 321–326

9. Gidas B (1987) Simulations and global optimization. In: Random media (Minneapolis, MN, 1985), IMA Vol Math Appl, vol 7. Springer, New York, pp 129–145
10. Gidas B (1985) Metropolis-type Monte Carlo simulation algorithms and simulated annealing. In: Topics in contemporary probability and its applications. Probab Stochastics Ser. CRC, Boca Raton, FL, pp 159–232
11. Hwang CR (1980) Laplace's method revisited: weak convergence of probability measures. Ann Probab 8(6):1177–1182
12. Kushner HJ (1987) Asymptotic global behavior for stochastic approximation and diffusions with slowly decreasing noise effects: global minimization via Monte Carlo. SIAM J Appl Math 47(1):169–185
13. Li-Zhi L, Liqun Q, Hon WT (2005) A gradient-based continuous method for large-scale optimization problems. J Glob Optim 31(2):271
14. Martin OC, Monasson R, Zecchina R (2001) Statistical mechanics methods and phase transitions in optimization problems. Theoret Comput Sci 265(1–2):3–67
15. Papadimitriou CH (1994) Computational complexity. Addison-Wesley, Reading, MA
16. Parpas P, Rustem B, Pistikopoulos E (2006) Linearly constrained global optimization and stochastic differential equations. J Glob Optim 36(2):191–217
17. Polak E, Royset JO, Womersley RS (2003) Algorithms with adaptive smoothing for finite minimax problems. J Optim Theory Appl 119(3):459–484
18. Rustem B, Howe M (2002) Algorithms for worst-case design and applications to risk management. Princeton University Press, Princeton, NJ
19. Stigler SM (1986) Laplace's 1774 memoir on inverse probability. Statist Sci 1(3):359–378
20. Talagrand M (2003) Spin glasses: a challenge for mathematicians. Ergebnisse der Mathematik und ihrer Grenzgebiete. 3. Folge (Results in Mathematics and Related Areas. 3rd Series). A Series of Modern. Surveys in Mathematics, vol 46. Springer, Berlin
21. Xu S (2001) Smoothing method for minimax problems. Comput Optim Appl 20(3):267–279
22. Zirilli F (1982) The use of ordinary differential equations in the solution of nonlinear systems of equations. In: Nonlinear optimization (Cambridge 1981). NATO Conf Ser II: Systems Sci. Academic Press, London, pp 39–46

Large Scale Trust Region Problems

LSTR

LAURA PALAGI
DIS, Universitá Roma 'La Sapienza', Rome, Italy

MSC2000: 90C30

Article Outline

Keywords

Large scale trust region problem; Exact penalty function; D.C. programming; Eigenvalue problem; Semidefinite programming

The *trust region* (TR) problem consists in minimizing a general quadratic function q: $\mathbf{R}^n \to \mathbf{R}$ of the type

$$q(x) = \frac{1}{2} x^\top Q x + c^\top x$$

subject to an ellipsoidal constraint $x^\intercal Hx \le r^2$ with the symmetric matrix H positive definite and r a positive scalar. By rescaling and without loss of generality, it can be assumed for sake of simplicity $H = I$, hence the TR problem is

$$\begin{cases} \min & q(x) \\ \text{s.t.} & \|x\|^2 \le r^2, \end{cases} \tag{1}$$

where $\|\cdot\|$ denotes the ℓ_2 norm.

The interest in this problem initially arose in the context of unconstrained optimization when $q(x)$ is a local quadratic model of the objective function which is 'trusted' to be valid over a restricted ellipsoidal region centered around the current iterate. However, it has been shown later that problems with the same structure of (1) are at the basis of algorithms for solving general constrained nonlinear programming problems (e. g. [2,14,19,21,27,28] and references therein), and for obtaining bounds for integer programming problems

(e. g. [10,11,12,17,18,26]; cf. also ► Integer programming).

Many papers have been devoted to study the specific features of Problem (1). It is well known [7,22] that a feasible point x^* is a global solution for (1) if and only if there exists a scalar $\lambda^* \geq 0$ such that the following KKT conditions are satisfied:

$$(Q + \lambda^* I)x^* = -c,$$
$$\lambda^*(\|x^*\|^2 - r^2) = 0,$$

and furthermore $Q + \lambda^* I \succcurlyeq 0$, where $\succcurlyeq$ denotes positive semidefinitness of the matrix.

Note that a complete characterization of global minimizers is given without requiring any convexity assumption on the matrix Q. Moreover, it has been proved that an approximation to the global solution can be computed in polynomial time (see, for example, [1,24,25]). Hence Problem (1) can be considered an 'easy' problem from a theoretical point of view. These peculiarities led to the development of 'ad hoc' algorithms for finding a global solution of Problem (1). The first ones proposed in [7,16,22] were essentially based on the solution of a sequence of linear system of the type $(Q + \lambda_k I)\, x = -\, c$ for a sequence $\{\lambda_k\}$. These algorithms produce an approximate global minimizer of Problem (1), but rely on the ability to compute a Cholesky factorization of the matrix $(Q + \lambda_k I)$ at each iteration k, and hence these methods are appropriate when forming a factorization for different values of λ_k is realistic in terms of both memory and time requirements. Indeed, they are appropriate for large scale problems with special structure, but in the general case, when no sparsity pattern is known, one cannot rely on factorizations of the matrices involved.

Thus one concentrates on iterative methods of conjugate gradient type (cf. ► Conjugate-gradient methods) that require only matrix-vector products. Among the methods that have been proposed to solve *large scale trust region* problems, the following two main categories can be identified:

- methods that produce a sequence of KKT points of (1) with progressive improvement of the objective function;
- methods that solve (1) via a sequence of parametric eigenvalue problems.

Algorithms Based on Successive Improvement of KKT Points

Methods in this class are based on special properties of KKT points of Problem (1). Indeed one can prove the following properties:

1) given a KKT point that is not a global minimizer, it is possible to find a new feasible point with a lower value of the objective function [5,13];
2) the number of distinct values of the objective function $q(x)$ at KKT points is bounded from above by $2m + 2$ where m is the number of negative eigenvalues of Q [13].

Exploiting these properties, a global minimizer of Problem (1) can be found, by applying a finite number of times an algorithm that, starting from a feasible point, locates a KKT point with a lower value of the objective function.

An algorithmic scheme of methods in this framework is summarized in the pseudocode of Table 1. The procedure described above is well-posed in the sense that it enters the 'DO cycle' a finite number of steps, since by Property 2, the function can assume at most a finite number of values at a KKT point.

To complete the scheme of Table 1 and obtain an efficient algorithm for the solution of Problem (1), it remains to specify how to move from a non global KKT

Large Scale Trust Region Problems, Table 1
A pseudocode for TR problem based on successive improvement of KKT points

```
procedure TR-IMPROVE-KKT()
   input instance (Q, c, r, x^0);
   Set k = 0; x = x^k; (starting point)
   find a KKT point x̂^k s.t. q(x̂^k) ≤ q(x^k);
   DO (until a global minimizer is found)
     (escape from a nonglobal KKT point)
       find x s.t. ‖ x ‖≤ r, q(x) < q(x̂^k);
     (update starting point)
       set k = k + 1, x^k = x;
     (find a 'better' KKT point)
        find a KKT point x̂^k s.t.
        q(x̂^k) ≤ q(x^k);
   OD;
   RETURN (solution)
END TR-IMPROVE-KKT;
```

point to a feasible point while improving the objective function, and how to define a globally and 'fast' convergent algorithm to locate a KKT point.

To check global optimality of a KKT point (i. e. to check if $Q + \lambda I \succcurlyeq 0$), one needs an estimate of the KKT multiplier λ corresponding to the point x, and has to verify whether $\lambda \geq -\lambda_{\min}(Q)$. To obtain λ the following multiplier function can be used

$$\lambda(x) = -\frac{1}{2r^2} x^\top (Qx + c), \tag{2}$$

which is consistent, namely at a KKT point $\lambda(x) = \lambda$. If $\lambda < -\lambda_{\min}(Q)$, then (x, λ) is a nonglobal KKT point and a negative curvature direction for the matrix $Q + \lambda I$ exists, namely a vector z such that $z^\intercal (Q + \lambda I)\, z < 0$. To perform the step 'escape from a non global KKT point', one can use such a direction. Roughly speaking and without discussing the details (see [5,13]), a new feasible point can be obtained by moving from x along z itself or along a direction easily obtainable from z of a computable quantity α. The efficiency of this step depends on the ability of finding efficiently such a vector z. Hence a procedure that finds an approximation of the minimum eigenvalue of $(Q + \lambda I)$ and of the corresponding eigenvector is needed. In the large scale setting, this can be done efficiently by using a Lanczos method [3,23] which meets the requirement of limited storage and needs only matrix-vector products.

In the algorithmic scheme of Table 1, it remains to define how to find efficiently a KKT point for Problem (1). Two different approaches have been recently (1998) proposed to perform this step; one is based on a continuously differentiable exact penalty function approach, the other is based on a difference of convex function approach. In both cases, the basic idea is to reformulate the constrained Problem (1) in a different form that allows one to use ideas typical of other fields of mathematical programming. Both approaches, which are described briefly in the sequel, treat indifferently the so called 'easy and hard' cases of Problem (1) and require only matrix vector products.

Exact Penalty Function Based Algorithm (EPA)

The main idea at the basis of a *continuously differentiable exact penalty function approach* is the reformulation of the constrained Problem (1) as an unconstrained one. In particular, a continuously differentiable function $P(x)$ can be defined [13] such that Problem (1) is 'equivalent' to the unconstrained problem

$$\min_{x \in \mathbb{R}^n} P(x).$$

The merit function takes full advantage of the structure of Problem (1) and it is a piecewise quartic function, whose definition relies on the particular multiplier function (2). The analytic expression of P is

$$P(x) = q(x) - \frac{\varepsilon}{4}\lambda(x)^2 + \frac{\varepsilon}{4} \max\left(0, \frac{2}{\varepsilon}(\|x\|^2 - r^2) + \lambda(x)\right)^2,$$

where $0 < \varepsilon < 2r^4/[r^2(\| Q \| + 1) + \|c\|^2]$. The function $P(x)$ has the following features:

- it has compact level sets;
- stationary (global minimum) points of $P(x)$ are KKT (global minimum) points of Problem (1) and vice versa; moreover $P(x) = q(x)$ at these points;
- the penalty parameter ε need not be updated;
- for points such that $\|x\|^2 \leq r^2$ it results $P(x) \leq q(x)$;
- $P(x)$ is twice continuously differentiable in a neighborhood of a KKT point that satisfies strict complementarity.

The unconstrained reformulation of Problem (1) can be exploited to define an algorithm for finding a KKT point while improving the value of objective function with respect to the initial one. Indeed any unconstrained method for the minimization of $P(x)$ can be used. Starting from a point x_0, any of these algorithms produce a sequence of the type

$$x^{k+1} = x^k + \alpha^k d^k, \tag{3}$$

where d^k is a suitable direction, α^k is a stepsize along d^k. The sequence $\{x^k\}$ need not to be feasible for Problem (1). The boundedness of the level sets of $P(x)$ guarantees the boundedness of the iterates and that any convergent unconstrained method obtains a stationary point $\overline{x}$ for P such that $P(\overline{x}) < P(x_0)$. Furthermore a stationary point of $P(x)$ is a KKT point of Problem (1) and $P(\overline{x}) = q(\overline{x})$. If, in addition, x_0 is a feasible point, the following relation holds:

$$q(\overline{x}) = P(\overline{x}) < P(x_0) \leq q(x_0),$$

which means that $\overline{x}$ is a KKT point of Problem (1) with a value of the objective function lower than the value at the starting point.

As regard the efficiency of the algorithms, in terms of rate of convergence and computational requirement, a 'good' direction d^k can be defined, by further exploiting the features of the unconstrained reformulation. Indeed, in a neighborhood of points satisfying the strict complementarity assumption, $P(x) \in C^2$ and therefore any unconstrained truncated Newton algorithm [4] can be easily adapted in order to define globally convergent methods which show a superlinear rate of convergence. Methods in this class include conjugate gradient based iterative method that requires only matrix-vector products and hence are suitable for large scale instances.

The resulting algorithmic scheme is reported in Table 2.

In the nonconvex case ($Q \not\geq 0$) strict complementarity holds in a neighborhood of every global minimizer of Problem (1) [13]. However, this may not be true in a neighborhood of a KKT point and the function $P(x)$ may be not twice differentiable there. Nevertheless algorithms which exhibit superlinear rate of convergence can be defined. In fact, drawing inspiration from the results in [6], the direction d^k is defined as the approximate solution of one of the following linear systems:

$$\begin{cases} \text{if} \quad \|x^k\|^2 - r^2 < -\varepsilon\frac{\lambda^k}{2}, \text{then} \\ \quad (Q + \lambda^k I)d^k = -(Qx^k + c), \\ \text{if} \quad \|x^k\|^2 - r^2 \geq -\varepsilon\frac{\lambda^k}{2}, \text{then} \\ \quad \begin{pmatrix} Q + \lambda^k I & x^k \\ (x^k)^\top & 0 \end{pmatrix} \begin{pmatrix} d^k \\ z^k \end{pmatrix} = \begin{pmatrix} -Qx^k - c \\ r^2 - \|x^k\|^2 \end{pmatrix}. \end{cases} \tag{4}$$

The solution of the linear systems (4), can be determined approximately by using the truncated Newton method proposed in [8]. The direction d^k satisfies suitable descent conditions with respect to the penalty function P, which can be used to measure the progressive improvement of the iterate. The stepsize α^k can be determined by any Armijo-type line search [9] that uses P as merit function.

It is possible to prove that the sequence $\{x^k\}$ produced by (3) with d^k obtained by (4) and $\{\lambda(x^k)\}$ by (2) converges to a KKT point $(\widehat{x}, \widehat{\lambda})$. Moreover if the KKT point $(\widehat{x}, \widehat{\lambda})$ satisfies $z^\top(Q + \widehat{\lambda}I)z > 0$ for all z: $z^\top\widehat{x} = 0$ whenever $\|\widehat{x}\|^2 = r^2$ and $\widehat{\lambda} > 0$, then there exists a neighborhood of $\widehat{x}$ where the rate of convergence of the algorithm is superlinear.

Large Scale Trust Region Problems, Table 2
A pseudocode for finding a KKT point by EPA

```
procedure KKT point by EPA()
   Given x^0 :|| x^0 ||^2 ≤ r^2 and ε > 0;
   set λ^0 = λ(x^0) and k = 0;
   DO (until a KKT point (x^k, λ^k) is found)
      set x^{k+1} = x^k + α^k d^k
      and λ^{k+1} = λ(x^{k+1});
      k = k + 1;
   OD;
   RETURN(KKT point);
END KKT point by EPA;
```

D.C. Decomposition Based Algorithm (DCA)

This algorithm is based on an appropriate reformulation of Problem (1) as the minimization of the difference of convex functions [5]. DCA has been proposed for solving large scale d.c. programming problems. The key aspect in d.c. optimization (cf. ► D.C. programming) relies on the particular structure of the objective function to be minimized on $\mathbf{R}^n$ that is expressed as $f(x) = g(x) - h(x)$, with g and h being convex. One uses the tools of convex analysis applied to the two components g and h of the d.c. function. In particular d.c. duality plays a fundamental role to understand how DCA works. Indeed for a generic d.c. problem, DCA constructs two sequences $\{x^k\}$ and $\{y^k\}$ and it can be viewed as a sort of decomposition approach of the primal and dual d.c. problems. It must be pointed out that a d.c. function has infinitely many d.c. decompositions that give rise to different primal dual pairs of d.c. problems and so to different DCA relative to these d.c. decompositions. Thus, choosing a d.c. decomposition may have an important influence on the qualities (such as robustness, stability, rate of convergence) of the DCA. This aspect is related to regularization techniques in d.c. programming.

In the special case of Problem (1), a quite appropriate d.c. decomposition has been proposed, so that DCA becomes very simple and it requires only matrix-vector products. To apply DCA to Problem (1), a *d.c. decomposition* of the objective function $f(x) = q(x) + \chi_F(x)$

must be defined, where $\chi_F(x)$ is the indicator function for the feasible set, namely

$$\chi_F(x) = \begin{cases} 0 & \text{if } \|x\|^2 \le r^2, \\ \infty & \text{otherwise.} \end{cases}$$

From the computational point of view, the most efficient decomposition that has been proposed is

$$g(x) = \frac{1}{2}\rho\|x\|^2 + c^\top x + \chi_F(x),$$
$$h(x) = \frac{1}{2}x^\top(\rho I - Q)x,$$

with $\rho > 0$ and such that $(\rho\, I - Q) \succcurlyeq 0$. In this case the sequence $\{y^k\}$ is obtained by the following rule $y^k = (\rho I - Q)\, x^k$ and x^{k+1} is obtained as the solution of the problem

$$\min_{x\in\mathbb{R}^n} \frac{1}{2}\rho\|x\|^2 + x^\top(c - y^k) + \chi_F(x).$$

Thus x^{k+1} is the projection of $(y^k - c)/\rho$ onto the feasible region $\|x\|^2 \le r^2$. The scheme for obtaining KKT points by DCA is reported in Table 3.

It has been proved [5] that algorithm DCA generates a sequence of feasible points $\{x^k\}$ with strictly decreasing value of the objective function and such that $\{x^k\}$ converges to a KKT point.

In practice the convergence rate depends on the choice of the parameter ρ. A possible choice (the best one according to some numerical experimentations performed in [5]) consists in taking ρ as close as possible to the largest eigenvalue of the matrix Q, namely $\rho = \max\{\lambda_{\max}(Q) + \varepsilon, 10^{-3}\}$ with $\varepsilon > 0$ and sufficiently small. Actually only a low accuracy estimate of $\lambda_{\max}(Q)$, which can be found by using a Lanczos method, is needed.

Large Scale Trust Region Problems, Table 3
A pseudocode for finding a KKT point by DCA

```
procedure KKT POINT by DCA()
  Given x^0, ρ > 0 such that (ρI − Q) ⪰ 0;
  DO (until a KKT point is found)
    IF ‖ (ρI − Q)x^k − c ‖≤ ρr THEN
      x^{k+1} = (1/ρ)[(ρI − Q)x^k − c]
    ELSE x^{k+1} = r ((ρI − Q)x^k − c) / ‖ (ρI − Q)x^k − c ‖
    END IF;
    IF ‖ x^{k+1} − x^k ‖≤ tol exit;
    set k = k + 1;
  OD;
  RETURN (KKT point);
END KKT POINT by DCA;
```

Parametric Eigenvalue Reformulation Based Algorithms

The algorithms in this framework are based on the reformulation of the TR problem into a parametric eigenvalue problem of a bordered matrix. It must be noted that, if the linear term is not present in the function $q(x)$, i. e. $c = 0$, Problem (1) is a pure quadratic problem that corresponds to finding the smallest eigenvalue of the matrix Q. Indeed the intuitive observation behind this idea is that given a real number t, one can write

$$\frac{1}{2}t + q(x) = \frac{1}{2}\begin{pmatrix}1\\ x\end{pmatrix}^\top \begin{pmatrix} t & c^\top \\ c & Q\end{pmatrix}\begin{pmatrix}1\\ x\end{pmatrix}$$

and for a fixed t the goal is to minimize the function $q(x)$ over the set $\{x\colon \|x\|^2 + 1 = r^2 + 1\}$, that is to minimize a pure quadratic form $z^\top D(t)\, z/2$ over a spherical region where

$$D(t) = \begin{pmatrix} t & c^\top \\ c & Q\end{pmatrix}.$$

This suggests that a solution of (1) may be found using eigenpairs of the matrix $D(t)$ where t is a parameter to be adjusted. Indeed, in both the algorithms proposed in this framework a key role is played by eigenpairs of the matrix $D(t)$. At each iteration the main computational step is the calculation of the smallest eigenvalue and a corresponding normalized eigenvector of the parametric matrix $D(t)$. The evaluation of the eigenvalue-eigenvector pair can be done by using Lanczos method as a black box. Therefore methods can exploit sparsity in the matrices and requires only matrix-vector multiplications. Moreover, only one element of the matrix $D(t)$ is changed at each iteration of both the algorithms and so consecutive steps of Lanczos algorithm become cheaper.

Both algorithms have to distinguish between the easy and hard case of Problem (1). The hard case is said to occur when the vector c is orthogonal to the eigenspace associated to the smallest eigenvalue of Q,

i. e. $c^{\mathsf{T}}y = 0$, for all $y \in S_{\min}$ with

$$S_{\min} = \{x \in \mathbb{R}^n : \; Qx = \lambda_{\min}(Q)x\}\,.$$

Depending on whether the easy or the hard case occurs, eigenpairs of the perturbed matrix $D(t)$ satisfies different properties. In the easy case, the smallest eigenvalue $\mu_{\min}\,(D(t))$ is simple and such that $\mu_{\min}\,(D(t)) < \lambda_{\min}\,(Q)$ for all values t. Moreover in this case the corresponding eigenvector has the first component not equal to zero and this plays a fundamental role in defining the iteration of both the algorithms. In the hard case caution should be used, due to the fact that the first component of the eigenvector corresponding to the smallest eigenvalue of $D(t)$ may be zero. Actually, any vector of the form $(0, y^{\mathsf{T}})^{\mathsf{T}}$ with $y \in S_{\min}$ is an eigenvector of the matrix $D(t)$ if and only if $c \perp S_{\min}$.

The two algorithms in this framework are briefly described below. Although the basic idea behind both the algorithms is the same, namely inverse interpolation for a parametric eigenvalue problem, the second one is embedded in a semidefinite programming framework. So the first one is referred to as 'inverse interpolation parametric eigenvalue' (IPE) approach and the second one as 'semidefinite programming approach' (SDP).

Inverse Interpolation Parametric Eigenvalue Formulation (IPE)

In [23] it is observed that if an eigenvector z of $D(t)$ corresponding to a given eigenvalue μ can be normalized so that its first component is one, that is $z = (1, x^{\mathsf{T}})^{\mathsf{T}}$, then a solution of the TR problem can be found in terms of eigenpairs of $D(t)$. This corresponds to the easy case and indeed the pair (x, μ) satisfies

$$\begin{pmatrix} t & c^{\top} \\ c & Q \end{pmatrix} \begin{pmatrix} 1 \\ x \end{pmatrix} = \mu \begin{pmatrix} 1 \\ x \end{pmatrix},$$

from which we get:

$$\begin{pmatrix} t - \mu = -c^{\top}x, \\ (Q - \mu I)x = -c. \end{pmatrix}$$

For $\mu < \lambda_{\min}(Q)$, that holds in the easy case with $\mu = \mu_{\min}(D(t))$, the matrix $(Q - \mu\, I)$ is positive definite and hence one can define the function

$$\phi(\mu) = -c^{\top}x = c^{\top}(Q - \mu I)^{-1}c,$$

whose derivative is

$$\phi'(\mu) = c^{\top}(Q - \mu I)^{-2}c = \|x\|^2\,.$$

For a given value of t, finding the smallest eigenvalue $\mu(t) := \mu_{\min}\,(D(t)) < \lambda_{\min}(Q)$ and the corresponding eigenvector of $D(t)$ and then normalizing the eigenvector to have its first component equal to one $(1, x_{\mu(t)}^{\mathsf{T}})^{\mathsf{T}}$ will provide a mean to evaluate the function $\phi(\mu)$ and its derivative. If t can be adjusted so that the corresponding $x_{\mu(t)}$ satisfies $\phi'(\mu(t)) = \| x_{\mu(t)} \|^2 = r^2$ with $t - \mu(t) = -\, c^{\mathsf{T}}\, x_{\mu(t)}$, and $\mu(t) \le 0$ then $(x, -\mu(t))$ satisfies the optimality conditions for Problem (1). Whereas if, during the course of adjusting t, it happens that $\mu(t) > 0$ with $\| x_{\mu(t)} \|^2 < r^2$ then the optimal solution of Problem (1) is actually unconstrained and can be found by solving the system $Qx = -\, c$ with any iterative method.

Hence using the parametric eigenvalue formulation, the optimal value of (x^*, λ^*) of Problem (1) can be found by solving a sequence of eigenvalue problems adjusting iteratively the parameter t. In order to make this observation useful, a modified Lanczos methods, the *implicit restarted Lanczos method* [23], is used for computing the smallest eigenvalue and the corresponding eigenvector of $D(t)$. Moreover a rapidly convergent iteration to adjust t has been developed, based on a two-point interpolant method. Recalling that the goal is to adjust t so that $\phi(\mu) = t - \mu$ and $\phi'(\mu) = r^2$, an interpolation based iteration that exploits the structure of the problem is proposed. The method is based upon an interpolant $\widehat{\phi}(\mu)$ of $\phi(\mu)$ of the form

$$\widehat{\phi}(\mu) = \frac{\gamma^2}{\alpha - \mu} + \beta(\alpha - \mu) + \delta.$$

The values of the parameters $\alpha, \beta, \gamma, \delta$ appearing in the interpolant function $\widehat{\phi}(\mu)$ are determined using the values of two iterations (x^k, μ^k), (x^{k-1}, μ^{k-1}) according to the following rules. The value δ is chosen so as to provide the current estimate $\delta_{\min}$ of $\lambda_{\min}(Q)$. In particular, if $\| x^k \| < r$ or $\| x^{k-1} \| < r$

$$\delta = \min\left(\delta_{\min}, \frac{(x^k)^{\top}Qx^k}{\left\|x^k\right\|^2}\right);$$

if $\| x^k \| > r$ and $\| x^{k-1} \| > r$ then

$$\delta = \min\left(\frac{(x^k)^{\top}Qx^k}{\left\|x^k\right\|^2}, \frac{(x^{k-1})^{\top}Qx^{k-1}}{\left\|x^{k-1}\right\|^2}\right)$$

Large Scale Trust Region Problems, Table 4
A pseudocode for TR based on (IPE)

```
procedure TR INTERPOL-PARAM-EIG()
  input instace (Q, c, r, x^0);
  (initialization)
  Find λ_min(Q) and its eigenvector x;
  set k = 0, t^k = 0, x^k = x, μ^k = λ_min(Q).
  DO ( until | (|| x^k ||^2 − r^2) / r^2 | ≤ tol )
    construct the interpolar φ̂(μ);
    find μ̂ : μ̂'(μ̂) = r^2, that is:
      μ̂ = α − ( γ^2 / (r^2 + β) )^{1/2};
    set t^{k+1} = μ̂ + φ̂(μ̂), that is:
      t^{k+1} = λ̂ + δ + β(α − μ̂) + γ^2/(α − μ̂);
    compute μ^{k+1} = μ_min(D(t^{k+1}))
      and the corresponding normalized eigenvec-
      tor ( 1, (x^{k+1})^⊤ )^⊤ ;
    set k = k + 1;
  OD;
  RETURN(solution)
END TR INTERPOLATION-PARAM-EIG;
```

and $\delta_{\min} = \min(\delta_{\min}, \delta) \geq \lambda_{\min}(Q)$. The other coefficient are chosen to satisfy $\widehat{\phi}(\mu^k) = -c^\top x^k$, $\widehat{\phi}'(\mu^k) = \|x^k\|^2$, $\widehat{\phi}'(\mu^{k-1}) = \|x^{k-1}\|^2$.

An algorithmic scheme for finding the global minimizer of Problem (1) in the easy case, is reported in Table 4.

It has been proved in [23] that there exists a neighborhood of $-\lambda^*$ such that if μ^0, μ^1 are in this neighborhood, all the sequence $\{\mu^k\}$ is well defined, remains in the neighborhood and converge superlinearly to $-\lambda^*$ with the corresponding iterates x^k converging superlinearly to x^*.

Unfortunately, the iteration described above can break down in the hard case. Indeed the iteration is based on the ability to normalize the eigenvector of the bordered matrix $D(t)$. This is not possible when the first component is equal to zero, that is in the hard case. From the computational point of view, also a near-hard case can be difficult and it is important to detect these cases and to define alternative rules so as to obtain a convergent iteration. This can be done, by using again eigenpairs of the bordered matrix and additional information such as the value of an upper bound λ_U on the optimal value λ^*. When the hard case is detected the new iteration should be used. The convergence of this new iteration can be established but unfortunately the rate of convergence is no longer superlinear.

Semidefinite Programming Approach (SDP)

In [20] a primal-dual simplex type method for Problem (1) has been proposed, which is essentially based on a primal dual pair of semidefinite programming problems. Primal-dual pairs of SDP provide a general framework for TR problem. The idea arises from the fact that Problem (1) enjoys strict *duality*, that is there is no duality gap and

$$q(x^*) = \min_x \max_\lambda L(x, \lambda) = \max_\lambda \min_x L(x, \lambda),$$

where $L(x, \lambda) = q(x) + \lambda(\|x\|^2 - r^2)$ denotes the Lagrangian function. By exploiting this feature it is possible to define a primal-dual pair of linear SDP problems that are strictly connected with the TR problem. In particular, a dual for Problem (1) is

$$\begin{cases} \max & (r^2+1)\mu_{\min}(D(t)) - t, \\ \text{s.t.} & \mu_{\min}(D(t)) \leq 0. \end{cases} \tag{5}$$

The objective function in (5) is a real valued concave function. When the constraint in Problem (1) is an equality one, its dual problem (5) is an unconstrained problem, and as an immediate consequence, the non convex constrained TR problem is transformed into a convex problem and hence it can be solved in polynomial time by the results for general convex programs.

Problem (5) can be easily reformulated as a SDP problem, by introducing an additional variable $\mu \in \mathbf{R}$:

$$\begin{cases} \max & (r^2+1)\mu - t, \\ \text{s.t.} & D(t) - \mu I \succeq 0, \\ & \mu \leq 0. \end{cases} \tag{6}$$

Slater's condition holds for Problem (6), and it is possible to write its Lagrangian dual that is:

$$\begin{cases} \min & \text{trace}(D(0)X), \\ \text{s.t.} & \text{trace}(X) \leq r^2 + 1, \\ & X_{11} = 1 \\ & X \succeq 0. \end{cases} \tag{7}$$

The algorithm parallels the dual simplex method for linear programming. At each iteration it maintains dual feasibility for Problem (6) and complementary slackness, while iterating to get primal feasibility of Problem (7) ($X_{11} = 1$) and reduce the duality gap.

Essentially these steps can be summarized as follows:

1) find a basic solution $(t, \mu_{\min}(D(t)))$ of Problem (6);
2) find an approximate solution of Problem (7), by using the complementary slackness relation

$$\text{trace}((D(t) - \mu I)X) = 0;$$

an eigenvector $z(t) = (z_0(t), v(t)^\mathsf{T})^\mathsf{T}$ corresponding to $\mu_{\min}(D(t))$ is used and $X = (r^2 + 1)\, zz^\mathsf{T}$ so that the constraint on the trace of X in Problem (7) is satisfied;
3) use inverse interpolation to predict a value of the parameter t such that $X_{11} = 1$ and/or the duality gap

$$\text{trace}(D(0)X) - ((r^2 + 1)\mu - t)$$

is decreasing.

Some differences occur depending on whether the easy or the hard case happens. Let us denote by $z(t) = (z_0(t), v(t)^\mathsf{T})^\mathsf{T}$ the eigenvector of $D(t)$ corresponding to $\mu_{\min}(D(t))$.

In the easy case, the first component $z_0(t) \neq 0$ and the vector $v(t)/z_0(t)$ is the unique optimal solution of

$$\begin{cases} \min & q(x) \\ \text{s.t.} & \|x\|^2 = \frac{1-z_0(t)^2}{z_0(t)^2}. \end{cases}$$

Hence, a value t^* such that $(1-z_0(t^*)^2)/z_0(t^*)^2 = r^2$ must be found and then the point $x^* = v(t^*)/z_0(t^*)$ with multiplier $\lambda^* = -\mu_{\min}(D(t^*))$ is the unique solution of Problem (1). The correct value of t can be found by standard search procedures and the algorithm produces an interval containing t^* that is iteratively updated.

In the hard case, $z_0(t)$ may be zero. However there is still a value t_0 such that $\mu_{\min}(D(t_0)) = \lambda_{\min}(Q)$ and a corresponding eigenvector $z(t_0)$ exists with first component not equal to zero. In order to obtain the value t_0, consider, without loss of generality, a diagonal Q with elements λ_i in increasing order, so that $\lambda_1 = \lambda_{\min}(Q)$. Assume that p is the multiplicity of $\lambda_{\min}(Q)$, and define

$$t_0 = \lambda_{\min}(Q) + \sum_{k=p+1}^{n} \frac{c_k}{\lambda_k - \lambda_{\min}(Q)}.$$

Then the smallest eigenvalue $\mu_{\min}(D(t_0)) = \lambda_{\min}(Q)$ with multiplicity $p + 1$.

Two cases can occur. If $(1-z_0(t_0)^2)/z_0(t_0)^2 > r^2$ then the value $t^* < t_0$. This case can be treated as the preceding easy case since there exists $t < t_0$ such that the eigenvalue $\mu_{\min}(D(t))$ is simple, it results $\mu_{\min}(D(t)) < \lambda_{\min}(Q)$, and the corresponding eigenvector satisfies $(1-z_0(t)^2)/z_0(t)^2 = r^2$. On the other hand, if $z_0^2(t_0) \geq 1/(r^2 + 1)$, then a primal step to the boundary of the feasible region of Problem (7) is taken while improving the objective function. In particular, let $w \in S_{\min}$ with $\|w\| = 1$, then the vector

$$x^* = \frac{v}{z_0(t_0)} + \left[\left(r^2 - \frac{1 - z_0^2(t_0)}{z_0^2(t_0)}\right) w\right]^{\frac{1}{2}}$$

together with $\lambda^* = -\lambda_{\min}(Q))$ satisfy the optimality conditions for Problem (1) and $t^* = t_0$. Hence in the hard case, a vector is found that allows to move to the correct radius while improving the objective function.

Inverse interpolation on the value of the first component z_0 of the eigenvector corresponding to $\mu_{\min}(D(t^k))$ is used to predict a new value for t^{k+1}.

A brief scheme of the algorithm is in Table 5.

Large Scale Trust Region Problems, Table 5
A pseudocode for TR based on SDP

```
procedure TR PRIMAL-DUAL-SDP()
    input instance (Q, c, r, x^0);
    (initialization)
    Find λ_min(Q); set k = 0.
    Set the interval of uncertainty
        [t_l^k, t_u^k] for t*, and [μ_l^k, μ_u^k] for q(x*);
    DO (until a solution is found)
        improve the parameter t^{k+1}
            using inverse interpolation
        update the iterate
            using μ_min(D(t^k)) and its corresponding
            eigenvector;
        update the intervals
            [t_l^{k+1}, t_u^{k+1}] and [μ_l^{k+1}, μ_u^{k+1}];
        set k = k + 1;    OD;
    RETURN(solution)
END TR PRIMAL-DUAL-SDP;
```

Conclusion

All the algorithms described above appear to be potentially equivalent from the computational point of view. They have been implemented in MATLAB [15] codes and the results of the numerical testing are reported in the corresponding papers.

See also

- ABS Algorithms for Linear Equations and Linear Least Squares
- Best Approximation by Bounded or Continuous Functions
- Cholesky Factorization
- Conjugate-gradient Methods
- Interval Linear Systems
- Large Scale Unconstrained Optimization
- Linear Programming
- Local Attractors for Gradient-related Descent Iterations
- Nonlinear Least Squares: Newton-type Methods
- Nonlinear Least Squares: Trust Region Methods
- Orthogonal Triangularization
- Overdetermined Systems of Linear Equations
- QR Factorization
- Solving Large Scale and Sparse Semidefinite Programs
- Symmetric Systems of Linear Equations

References

1. Ben-tal A, Teboulle M (1996) Hidden convexity in some nonconvex quadratically constrained quadratic programming. Math Program 72(1):51–63
2. Coleman TF, Li Y (1996) An interior trust region approach for nonlinear minimization subject to bounds. SIAM J Optim 6(2):418–445
3. Cullum JK, Willoughby RA (1985) Lanczos algorithms for large symmetric eigenvalue computation. Birkhäuser, Basel
4. Dembo RS, Steihaug T (1983) Truncated-Newton methods algorithms for large-scale unconstrained optimization. Math Program 26(2):190–212
5. DinhTao Pham, HoaiAn LeThi (1998) D.C. optimization algorithm for solving the trust region subproblem. SIAM J Optim 8(2):476–505
6. Facchinei F, Lucidi S (1995) Quadratically and superlinear convergent algorithms for the solution of inequality constrained optimization problems. J Optim Th Appl 85(2):265–289
7. Gay DM (1981) Computing optimal locally constrained steps. SIAM J Sci Statist Comput 2(2):186–197
8. Grippo L, Lampariello F, Lucidi S (1989) A truncated Newton method with nonmonotone linesearch for unconstrained optimization. J Optim Th Appl 60(3):401–419
9. Grippo L, Lampariello F, Lucidi S (1991) A class of nonmonotone stabilization methods in unconstrained optimization. Numerische Math 59:779–805
10. Kamath A, Karmarkar N (1991) A continuous approach to compute upper bounds in quadratic maximization problems with integer constraints. In: Floudas CA, Pardalos PM (eds) Recent Advances in Global Optimization. Princeton Univ. Press, Princeton, pp 125–140
11. Karmarkar N (1990) An interior-point approach to {NP}-complete problems. In: Proc. Math. Program. Soc. Conf. Integer Programming and Combinatorial Optimization, pp 351–366
12. Karmarkar N, Resende MGC, Ramakrishnan KG (1991) An interior point algorithm to solve computationally difficult set covering problems. Math Program 52(3):597–618
13. Lucidi S, Palagi L, Roma M (1998) On some properties of quadratic programs with a convex quadratic constraint. SIAM J Optim 8(1):105–122
14. Martínez JM, Santos SA (1995) Trust region algorithms on arbitrary domains. Math Program 68(3):267–302
15. Matlab (1995) Reference guide. MathWorks
16. Moré JJ, Sorensen DC (1983) Computing a trust region step. SIAM J Sci Statist Comput 4(3):553–572
17. Pardalos PM (1996) Continuous approches to discrete optimization problems. In: Di Pillo G, Giannessi F (eds) Nonlinear Optimization and Applications. Plenum, New York, pp 313–328
18. Pardalos PM, Ye Y, Han C-G (1991) Algorithms for the solution of quadratic knapsack problems. LAA 25:69–91
19. Powell MJD, Yuan Y (1991) A trust region algorithm for equality constrained optimization. Math Program 49:189–211
20. Rendl F, Wolkowicz H (1997) A semidefinite framework to trust region subproblems with applications to large scale minimization. Math Program 77(2):273–299
21. Sartenaer A (1995) A class of trust region methods for nonlinear network optimization problems. SIAM J Optim 5(2):379–407
22. Sorensen DC (1982) Newton's method with a model trust region modification. SIAM J Sci Statist Comput 19(2):409–427
23. Sorensen DC (1997) Minimization of a large-scale quadratic function subject to an ellipsoidal constraint. SIAM J Optim 7(1):141–161
24. Vavasis SA (1991) Nonlinear optimization. Oxford Univ. Press, Oxford
25. Ye Y (1991) A new complexity result on minimization of a quadratic function with a sphere constraint. In: Floudas CA, Pardalos PM (eds) Recent Advances in Global Optimization. Princeton Univ. Press, Princeton, pp 19–31

26. Ye Y (1992) On affine scaling algorithms for nonconvex quadratic programming. Math Program 56:285–300
27. Ye Y, Tse E (1989) An extension of Karmarkar's projective algorithm for convex quadratic programming. Math Program 44:157–179
28. Yuan Y (1990) On a subproblem of trust region algorithms for constrained optimization. Math Program 47:33–63

Large Scale Unconstrained Optimization

LSUO

MASSIMO ROMA
Dip. Inform. e Sistemistica,
Universitá Roma 'La Sapienza', Roma, Italy

MSC2000: 90C06

Article Outline

Keywords

Large scale problem; Unconstrained optimization

A large scale unconstrained optimization problem can be formulated as the problem of finding a local minimizer of a real valued function $f: \mathbf{R}^n \to \mathbf{R}$ over the space $\mathbf{R}^n$, namely to solve the problem

$$\min_{x \in \mathbb{R}^n} f(x), \tag{1}$$

where the dimension n is large. The notion of 'large scale' is machine dependent and hence it could be difficult to state a priori when a problem is of large size. However, today an unconstrained problem with more than one thousand variables is usually considered a *large scale problem*.

Besides its own theoretical importance, the growing interest in the last years in solving problems of large size derives from the fact that problems with a larger and larger number of variables are arising very frequently from real world as a result of modeling systems with a very complex structure.

The main difficulty in dealing with large scale problems is the fact that effective algorithms for small scale problems do not necessarily translate into efficient algorithms when applied to solve large problems. Therefore in most cases it is improper to tackle a problem with a large number of variables by using one of the many existing algorithms for the small scale case relying on the growing powerful of the modern computers (see, e. g., [11,13,34] for a review on the existing methods for small scale unconstrained optimization).

A basic feature of an algorithm for large scale problems is a low storage overhead needed to make practicable its implementation. Moreover, whenever a large scale problem has some structure it should be exploited to define reliable algorithms; in fact, often the structure of a problem reflects in the sparsity of the Hessian matrix of the function f which can be efficiently exploited.

Methods for unconstrained optimization differ according to how much information on the function f is available. In the framework of large scale unconstrained optimization it is usually required that the user provides at least subroutines which evaluate the objective function and its gradient for any point x. More effective methods can be obtained if second order derivatives are known. When the derivatives are not available they can be obtained by finite difference or by using automatic differentiation. Throughout we assume that the function f is twice continuously differentiable, i. e. that the gradient $g(x) = \nabla f(x)$ and the Hessian matrix $H(x) = \nabla^2 f(x)$ of the function f exist and are continuous. Moreover, we denote by $\|v\|$ the Euclidean norm of a vector $v \in \mathbf{R}^n$.

As in the small scale case, most of the large scale unconstrained algorithms are iterative methods which generate a sequence of points according to the scheme

$$x_{k+1} = x_k + \alpha_k d_k \tag{2}$$

where $d_k \in \mathbf{R}^n$ is a search direction and $\alpha_k \in \mathbf{R}$ is a steplength obtained by means of a one-dimensional search. Obviously, also in large scale optimization it is important that an algorithm presents both the *global convergence* (i. e. convergence of the sequence $\{x_k\}$ towards a stationary point from any starting point) and a good *convergence rate*.

A basic method for solving large scale unconstrained optimization problems can be considered the *steepest descent method* obtained by setting $d_k = -g(x_k)$ in (2). This method is based on the linear approximation of the objective function f and hence only first order information are need. Due to its very limited storage required by a standard implementation, steepest descent method could be considered very attractive in the large scale setting; moreover the global convergence can also be ensured. However, its convergence rate is only linear and therefore it is too slow to be used. A particular rule for computing the stepsize α_k has been proposed [39] and this led to a significant improvement of the efficiency of the steepest descent method.

One of the most effective methods for solving unconstrained problems is the Newton method (cf. ► Unconstrained nonlinear optimization: Newton–Cauchy framework). It is based on the quadratic approximation of $f(x_k + w)$ given by

$$\phi_k(w) = f(x_k) + g(x_k)^\top w + \tfrac{1}{2} w^\top H(x_k) w \qquad (3)$$

and it is defined by iterations of the form

$$x_{k+1} = x_k + s_k \qquad (4)$$

where the search direction s_k is obtained by minimizing the quadratic model of the objective function (3) over $\mathbf{R}^n$. On the one hand, Newton method presents quadratic convergence rate and it is scale invariant, but, on the other hand, in its pure form it is not globally convergent. Globally convergent modifications of the Newton method has been defined following the line search approach and the trust region approach (see, e. g. [11,12,27]; cf. also ► Large scale trust region problems), but the main difficulty, in dealing with large scale problems, is represented by the possibility to efficiently solve, at each iteration, linear systems which arise in computing the search direction s_k. In fact, the problem dimension could be too large for any explicit use of the Hessian matrix and iterative methods must be used to solve systems of linear equations instead of factorizations of the matrices involved. Indeed, whereas in the small scale setting the Newton direction s_k is usually determined by using direct methods for solving the linear system

$$H(x_k)s = -g(x_k), \qquad (5)$$

when n is large, it is impossible to store or factor the full $n \times n$ Hessian matrix unless it is a sparse matrix. Moreover the exact solution, at each iteration, of the system (5) could be too burdensome and not justified when x_k is far from a solution. In fact, since the benefits of using the Newton direction are mainly local (i. e. in the neighborhood of a solution), it should not be necessary a great computational effort to get an accurate solution of system (5) when $g(x_k)$ is large.

On the basis of these remarks, in [8] the *inexact Newton methods* were proposed. They represent the basic approach underlying most of the Newton-type large scale unconstrained algorithms. The main idea is to approximately solve the system (5) still ensuring a good convergence rate of the method by using a particular trade-off rule between the computational burden required to solve the system (5) and the accuracy with which it is solved. The measure of this accuracy is the relative residual

$$\frac{\|r_k\|}{\|g(x_k)\|}, \quad \text{where } r_k = H(x_k)s_k + g(x_k) \qquad (6)$$

and s_k is an approximate solution of (5). The analysis given in [8] shows that if the sequence $\{x_k\}$ generated by (4) converges to a point $x_\star$ and if

$$\lim_{k\to\infty} \frac{\|r_k\|}{\|g(x_k)\|} = 0, \qquad (7)$$

then $\{x_k\}$ converges superlinearly to $x_\star$. This result is at the basis of the *truncated Newton methods* which represent one of the most effective approach for solving large scale problems. This class of methods was introduced in [9] within the line search based Newton-type methods. They are based on the fact that whenever the Hessian matrix $H(x_k)$ is positive definite, to solve the Newton equation (5) is equivalent to determine the minimizer of the quadratic model (3). Therefore, in these methods, a Newton-type direction, i. e. an approximate solution of (5), is computed by applying the (linear) conjugate gradient (CG) method (cf. ► Conjugate-gradient methods) [23] to approximately minimize the quadratic function (3). A scheme of a line search based truncated Newton algorithm is the following:

Line search based truncated Newton algorithm

OUTER iterations
  For $k = 0, 1, \ldots$
  Compute $g(x_k)$
  Test for convergence
INNER iterations
  (Computation of the direction s_k)
  Iterate CG algorithm until
  a termination criterion is satisfied
  Compute a stepsize α_k by a line search procedure
  Set $x_{k+1} = x_k + \alpha_k s_k$

A scheme for a truncated Newton algorithm

Given a starting point x_0, at each iteration k, a Newton-type direction s_k is computed by truncating the CG iterates – the inner iterations – whenever a required accuracy is obtained. The definition of an effective truncation criterion represents a key aspect of any truncated Newton method and a natural choice is represented by monitoring when the relative residual (6) is sufficiently small. Moreover, by requiring that $\|r_k\| / \|g(x_k)\| \leq \eta_k$ with $\lim_{k\to\infty} \eta_k \to 0$, the condition given by (7) is satisfied and hence the superlinear convergence is guaranteed [9]. In particular η_k can be chosen to ensure that, as a critical point is approached, more accuracy is required. Other truncation criteria based on the reduction of the quadratic model can be defined [31]. Numerical experiences showed that a relatively small number of CG iterations is needed, in most cases, for obtaining a good approximation of the Newton direction and this is one the main advantage of the truncated Newton methods since a considerable computational savings can be obtained still ensuring a good convergence rate. The performance of the CG algorithm used in the inner iterations can be improved by using a preconditioning strategy based either on the information gained during the outer iterations or on some scaling of the variables. Several different preconditioning schemes have been proposed and tested [29,40]. Truncated Newton methods can be modified to enable their use whenever the Hessian matrix is not available; in fact, the CG method only needs the product of the Hessian matrix with a displacement vector, and this product can be approximated by finite difference [35]. The resulting method is called *discrete truncated Newton method*. In [41] a Fortran package (TNPACK) implementing a line search based (discrete) truncated Newton algorithm which uses a preconditioned conjugate gradient is proposed. However, additional safeguard is needed within truncated Newton algorithms since the Hessian matrix could be not positive definite. In fact, the CG inner iterations may break down before satisfying the termination criterion when the Hessian matrix is indefinite. To handle this case, whenever a *direction of negative curvature* (i. e. a direction d_k such that $d_k^\top H(x_k)\, d_k < 0$) is encountered, the inner iterations are usually terminated and a *descent direction* (i. e. a direction d_k such that $g(x_k)^\intercal\, d_k < 0$) is computed [9]. More sophisticated strategies can be applied for iteratively solving the system (5) when it is indefinite [6,15,36,43]. In particular, the equivalent characterization of the linear conjugate gradient algorithm via the Lanczos method can be exploited to define a truncated Newton algorithm which can be used to solve problems with indefinite Hessian matrices [28]. In fact, the Lanczos algorithm does not requires the Hessian matrix to be positive definite and hence it enables to obtain an effective Newton-type direction.

A truncated Newton method which uses a *nonmonotone line search* (i. e. which does not enforce the monotone decrease of the objective function values) was proposed in [20] and the effectiveness of this approach was shown especially in the solution of ill-conditioned problems. Moreover in the CG-truncated scheme proposed in [20] an efficient strategy to handle the indefinite case is also proposed.

A new class of truncated Newton algorithms for solving large scale unconstrained problems has been defined in [25]. In particular, a nonmonotone stabilization framework is proposed based on a *curvilinear line search*, i. e. a line search along the curvilinear path

$$x(\alpha) = x_k + \alpha^2 s_k + \alpha d_k,$$

where s_k is a Newton-type direction and d_k is a particular negative curvature direction which has some resemblance to an eigenvector of the Hessian matrix corresponding to the minimum eigenvalue. The use of the combination of these two directions enables, also in the large scale case, to define a class of line search based algorithms which are globally convergent towards points which satisfy second order necessary optimality conditions, i. e. stationary points where the Hessian matrix is

positive semidefinite. Besides satisfying this important theoretical property, this class of algorithms was also shown to be very efficient in solving large scale unconstrained problems [25,26]. This is also due to the fact that a Lanczos based iterative scheme is used to compute both the directions without terminating the inner iterations when indefiniteness is detected and, as result, more information about the curvature of the objective function are conveyed.

Truncated Newton methods have been also defined within the trust region based methods. These methods are characterized by iterations of the form (4) where, at each iteration k, the search direction s_k is determined by minimizing the quadratic model of the objective function (3) in a neighborhood of the current iterate, namely by solving the problem

$$\min_{\|s\| \le \Delta} \phi_k(s), \tag{8}$$

where Δ is the trust region radius. Also in this framework most of the existing algorithms require the solution of systems of linear equations. Some approaches are the *dogleg methods* [10,38] which aim to solve problem (8) over a one-dimensional arc and the method proposed in [5] which solves problem (8) over a two-dimensional subspace. However, whenever the problem dimension is large, it is impossible to rely on matrix factorizations, and iterative methods must be used. If the quadratic model (3) is positive definite and the trust region radius is sufficiently large that the trust region constraint is inactive at the unconstrained minimizer of the model, problem (8) can be solved by using the preconditioned conjugate gradient method [42,44]. Of course, a suitable strategy is needed whenever the unconstrained minimizer of the quadratic model is no longer lying within the trust region and the desired solution belongs to the trust region boundary. A simple strategy to handle this case was proposed in [42] and [44] and it considers the piecewise linear path connecting the CG iterates, stopping at the point where this path leaves the trust region. If the quadratic model (3) is indefinite, the solution must also lie on the trust region boundary and the piecewise linear path can be again followed until either it leaves the trust region, or a negative curvature direction is found. In this latter case, two possibilities have been considered: in [42] the path is continued along this direction until the boundary is reached; in [44] the minimizer of the quadratic model within the trust region along the steepest descent direction (the *Cauchy point*) is considered. This class of algorithms represents a trust region version of truncated Newton methods and an efficient implementation is carried out within the LANCELOT package [7]. These methods have become very important in large scale optimization, due to both their strong theoretical convergence properties and good efficiency in practice, but they are known to possess some drawbacks. Indeed, they are essentially unconcerned with the trust region until they blunder into its boundary and stop. Moreover, numerical experiences showed that very frequently this untimely stop happens during the first inner iterations when a negative curvature is present and this could deteriorate the efficiency of the method. In order to overcome this drawback an alternative strategy is proposed in [16] where ways of continuing the process once the boundary of the trust region is reached are investigated. The key point of this approach is the use of the Lanczos method and the fact that preconditioned conjugate gradient and Lanczos methods generate different bases for the same Krylov space. Several other large scale trust region methods (cf. ► Large scale trust region problems) have been proposed.

Another class of methods which can be successfully applied to solve large scale unconstrained optimization problems is the wide class of the nonlinear conjugate gradient methods [14,23]. They are extensions to the general (nonquadratic) case of the already mentioned linear conjugate gradient method. They represent a compromise between steepest descent method and Newton method and they are particularly suited for large scale problems since there is never a need to store a full Hessian matrix. They are defined by the iteration scheme (2) where the search direction is of the form

$$d_k = -g(x_k) + \beta_k d_{k-1} \tag{9}$$

with $d_0 = -g(x_0)$ and where β_k is a scalar such that the algorithm reduces to the linear conjugate gradient method if the objective function f is a strictly convex quadratic function and α_k in (2) is obtained by means of an exact line search (i. e., α_k is the one-dimensional minimizer of $f(x_k + \alpha\, d_k)$ with respect to α). The most widely used formulas for β_k are *Fletcher–Reeves* (FR)

and *Polak–Ribière* (PR) formulas given by

$$\beta_k^{\mathrm{FR}} = \frac{\|g(x_k)\|^2}{\|g(x_{k-1})\|^2},$$
$$\beta_k^{\mathrm{PR}} = \frac{g(x_k)^\top \left[g(x_k) - g(x_{k-1})\right]}{\|g(x_{k-1})\|^2}.$$

Many efforts have been devoted to investigate the global convergence for nonlinear conjugate gradient methods. A widespread technique to enforce the global convergence is the use of a regular *restart* along the steepest descent direction every n iterations obtained by setting $\beta_k = 0$. However, computational experiences showed that this restart can have a negative effect on the efficiency of the method; on the other hand, in the large scale setting, restarting does not play a significant role since n is large and very few restarts can be performed. Global convergence results have been obtained for the *Fletcher–Reeves method* without restart both in the case of exact line search [46] and when α_k is computed by means of an inexact line search [1]; then, the global convergence was extended to methods with $|\beta_k| \le \beta_k^{FR}$ [14]. As regards the global convergence of the *Polak–Ribière method*, for many years it was proved with exact line search only under strong convexity assumptions [37]. Global convergence both for exact and inexact line search can also be enforced by modifying the Polak–Ribière method by setting $\beta_k = \max\{\beta_k^{PR}, 0\}$ [14]; this strategy correspond to restart the iterations along the steepest descent direction whenever a negative value of β_k occurs. However, an inexact line search which ensures global convergence of the Polak–Ribière method for nonconvex function has been obtained in [21]. As regards the numerical performance of these two methods, extensive numerical experiences showed that, in general, Polak–Ribière method is usually more efficient than the Fletcher–Reeves method. An efficient implementation of the Polak–Ribière method (with restarts) is available as routine VA14 within the Harwell subroutine library [22]. See, e. g., [34] for a detailed survey on the nonlinear conjugate gradient methods.

Another effective approach to large scale unconstrained optimization is represented by the *limited-memory BFGS method* (L-BFGS) proposed in [32] and then studied in [24,30]. This method resembles the BFGS quasi-Newton method, but it is particularly suited for large scale (unstructured) problems because the storage of matrices is avoided. It is defined by the iterative scheme (2) with the search direction given by

$$d_k = -H_k g(x_k)$$

and where H_k is the approximation to the inverse Hessian matrix of the function f at the kth iteration. In the BFGS method the approximation H_k is updated by means of the BFGS correction given by

$$H_{k+1} = V_k^\top H_k V_k + \rho_k s_k s_k^\top$$

where $V_k = I - \rho_k\, y_k\, s_k^\top$, $s_k = x_{k+1} - x_k$, $y_k = g(x_{k+1}) - g(x_k)$, and $\rho_k = 1/y_k^\top\, s_k$. In the L-BFGS method, instead of storing the matrices H_k, a prefixed number (say m) of vectors pairs $\{s_k, y_k\}$ that define them implicitly are stored. Therefore, during the first m iterations the L-BFGS and the BFGS methods are identical, but when $k > m$ only information from the m previous iterations are used to obtain H_k. The number m of BFGS corrections that must be kept can be specified by the user. Moreover, in the L-BFGS the product $H_k\, g(x_k)$ which represents the search direction is obtained by means of a recursive formula involving $g(x_k)$ and the most recent vectors pairs $\{s_k, y_k\}$. An implementation of L-BFGS method is available as VA15 routine within the Harwell subroutine library [22]. An interesting numerical study of L-BFGS method and a comparison of its numerical performance with the discrete truncated Newton method and the Polak–Ribière conjugate gradient method are reported in [30]. The results of a numerical experience with limited-memory quasi-Newton and truncated Newton methods on standard library test problems and on two real life large scale unconstrained optimization applications can be found in [45]. A method which combines the discrete Newton method and the L-BFGS method is proposed in [4] to produce an efficient algorithm able to handle also ill-conditioned problems.

Limited memory quasi-Newton methods represent an adaptation of the quasi-Newton methods to large scale *unstructured optimization*. However, the quasi-Newton approach can be successfully applied to large scale problems with a particular structure. In fact, fre-

quently, an optimization problem has some structure which may be reflected in the sparsity of the Hessian matrix. In this framework, the most effective method is the *partitioned quasi-Newton method* proposed in [18,19]. It is based on the fact that a function f with a sparse Hessian is a *partially separable function*, i. e. it can be written in the form

$$f(x) = \sum_{i=1}^{n_e} f_i(x)$$

where the element functions f_i depends only on a few variables. Many practical problems can be formulated (or recasted) in this form showing a wide range of applicability of this approach. The basic idea of the partitioned quasi-Newton method is to decompose the Hessian matrix into a sum of Hessians of the element functions f_i. Each approximation to the Hessian of f_i is then updated by using dense updating techniques. These small matrices are assembled to define an approximation to the Hessian matrix of f used to compute the search direction. However, the element Hessian matrices may not be positive definite and hence BFGS formula cannot be used, and in this case a symmetric rank one formula is used. Global convergence results have been obtained under convexity assumption of the function f_i [17]. An implementation of the partitioned quasi-Newton method is available as VE08 routine of the Harwell subroutine library [22]. A comparison of the performance of partitioned quasi-Newton, L-BFGS, CG Polak–Ribière and truncated discrete Newton methods is reported in [33].

Another class of methods which has been extended to large sparse unconstrained optimization are *tensor methods* [3]. Tensor methods are based on fourth order model of the objective function and are particularly suited for problems where the Hessian matrix has a small rank deficiency.

To conclude, it is worthy to outline that in dealing with large scale unconstrained problems with a very large number of variables (more than 10^4) high performance computer architectures must be considered. See e. g. [2] for the solution of large scale optimization problems on vector and parallel architectures.

The reader can find the details of the methods mentioned in this brief survey in the specific cited references.

See also

- ABS Algorithms for Linear Equations and Linear Least Squares
- Broyden Family of Methods and the BFGS Update
- Cholesky Factorization
- Conjugate-gradient Methods
- Continuous Global Optimization: Models, Algorithms and Software
- Interval Linear Systems
- Large Scale Trust Region Problems
- Linear Programming
- Modeling Languages in Optimization: A New Paradigm
- Nonlinear Least Squares: Trust Region Methods
- Optimization Software
- Orthogonal Triangularization
- Overdetermined Systems of Linear Equations
- QR Factorization
- Solving Large Scale and Sparse Semidefinite Programs
- Symmetric Systems of Linear Equations
- Unconstrained Nonlinear Optimization: Newton–Cauchy Framework
- Unconstrained Optimization in Neural Network Training

References

1. Al-Baali M (1985) Descent property and global convergence of the Fletcher–Reeves method with inexact line search. IMA J Numer Anal 5:121–124
2. Averick BM, Moré JJ (1994) Evaluation of large-scale optimization problems on vector and parallel architectures. SIAM J Optim 4:708–721
3. Bouaricha A (1997) Tensor methods for large, sparse unconstrained optimization. SIAM J Optim 7:732–756
4. Byrd RH, Nocedal J, Zhu C (1995) Towards a discrete Newton method with memory for large-scale optimization. In: Di Pillo G, Giannessi F (eds) Nonlinear Optimization and Applications. Plenum, New York, pp 1–12
5. Byrd RH, Schnabel RB, Shultz GA (1988) Approximate solution of the trust region problem by minimization over two-dimensional subspaces. Math Program 40:247–263
6. Chandra R (1978) Conjugate gradient methods for partial differential equations. PhD Thesis Yale Univ.
7. Conn AR, Gould NIM, Toint PhL (1992) LANCELOT: A Fortran package for large-scale nonlinear optimization (release A). Springer, Berlin
8. Dembo RS, Eisenstat SC, Steihaug T (1982) Inexact Newton methods. SIAM J Numer Anal 19:400–408

9. Dembo RS, Steihaug T (1983) Truncated-Newton algorithms for large-scale unconstrained optimization. Math Program 26:190–212
10. Dennis JE, Mei HHW (1979) Two new unconstrained optimization algorithms which use function and gradient values. J Optim Th Appl 28:453–482
11. Dennis JE, Schnabel RB (1989) A view of unconstrained optimization. In: Nemhauser GL, Rinnooy Kan AHG, Tood MJ (eds) Handbook Oper. Res. and Management Sci., vol 1. North-Holland, Amsterdam, pp 1–72
12. Fletcher R (1987) Practical methods of optimization. Wiley, New York
13. Fletcher R (1994) An overview of unconstrained optimization. In: Spedicato E (ed) Algorithms for continuous optimization. The state of the art. Kluwer, Dordrecht, pp 109–143
14. Gilbert JC, Nocedal J (1992) Global convergence properties of conjugate gradient methods for optimization. SIAM J Optim 2:21–42
15. Gill PE, Murray W, Ponceleon DB, Saunders MA (1992) Preconditioners for indefinite systems arising in optimization. SIAM J Matrix Anal Appl 13:292–311
16. Gould NIM, Lucidi S, Roma M, Toint PhL (1999) Solving the trust-region subproblem using the Lanczos method. SIAM J Optim 9:504–525
17. Griewank A (1991) The global convergence of partitioned BFGS on problems with convex decomposition and Lipschitzian gradients. Math Program 50:141–175
18. Griewank A, Toint PhL (1982) Local convergence analysis of partitioned quasi-Newton updates. Numerische Math 39:429–448
19. Griewank A, Toint PhL (1982) Partitioned variable metric updates for large structured optimization problems. Numerische Math 39:119–137
20. Grippo L, Lampariello F, Lucidi S (1989) A truncated Newton method with nonmonotone linesearch for unconstrained optimization. J Optim Th Appl 60:401–419
21. Grippo L, Lucidi S (1997) A globally convergent version of the Polak–Ribière conjugate gradient method. Math Program 78:375–391
22. Harwell Subroutine Library (1998) A catalogue of subroutines. AEA Techn.
23. Hestenes MR (1980) Conjugate direction methods in optimization. Springer, Berlin
24. Liu DC, Nocedal J (1989) On the limited memory BFGS method for large scale optimization. Math Program 45:503–528
25. Lucidi S, Rochetich F, Roma M (1998) Curvilinear stabilization techniques for truncated Newton methods in large scale unconstrained optimization. SIAM J Optim 8:916–939
26. Lucidi S, Roma M (1997) Numerical experiences with new truncated Newton methods in large scale unconstrained optimization. Comput Optim Appl 7:71–87
27. Moré JJ, Sorensen DC (1984) Newton's method. In: Golub GH (ed) Studies in Numerical Analysis. Math. Assoc. Amer., Washington, DC, pp 29–82
28. Nash SG (1984) Newton-type minimization via the Lanczos method. SIAM J Numer Anal 21:770–788
29. Nash SG (1985) Preconditioning of truncated-Newton methods. SIAM J Sci Statist Comput 6:599–616
30. Nash SG, Nocedal J (1991) A numerical study of the limited memory BFGS method and the truncated-Newton method for large scale optimization. SIAM J Optim 1:358–372
31. Nash SG, Sofer A (1990) Assessing a search direction within a truncated-Newton method. Oper Res Lett 9:219–221
32. Nocedal J (1980) Updating quasi-Newton matrices with limited storage. Math Comput 35:773–782
33. Nocedal J (1990) The performance of several algorithms for large-scale unconstrained optimization. In: Coleman TF, Li Y (eds) Large-scale Numerical Optimization. SIAM, Philadelphia, pp 138–151
34. Nocedal J (1992) Theory and algorithms for unconstrained optimization. Acta Numer 1:199–242
35. O'Leary DP (1982) A discrete Newton algorithm for minimizing a function of many variables. Math Program 23:20–33
36. Paige CC, Saunders MA (1975) Solution of sparse indefinite systems of linear equations. SIAM J Numer Anal 12:617–629
37. Polak E, Ribière G (1969) Note sur la convergence de methodes de directions conjugées. Revue Franc Inform et Rech Oper 16:35–43
38. Powell MJD (1970) A new algorithm for unconstrained optimization. In: Mangasarian OL, Ritter K (eds) Nonlinear programming. Acad. Press, New York, pp 31–65
39. Raydan M (1997) The Barzilai and Borwein gradient method for large scale unconstrained minimization problems. SIAM J Optim 7:26–33
40. Schlick T (1993) Modified Cholesky factorization for sparse preconditioners. SIAM J Sci Comput 14:424–445
41. Schlick T, Fogelson A (1992) TNPACK – A truncated Newton package for large-scale problems: I. Algorithm and usage. ACM Trans Math Softw 18:46–70
42. Steihaug T (1983) The conjugate gradient method and trust regions in large-scale optimization. SIAM J Numer Anal 20:626–637
43. Stoer J (1983) Solution of large linear systems of equations by conjugate gradient type methods. In: Bachem A, Grötschel M, Korte B (eds) Mathematical Programming. The State of the Art. Springer, Berlin, pp 540–565
44. Toint PhL (1981) Towards an efficient sparsity exploiting Newton method for minimization. In: Duff IS (ed) Sparse Matrices and Their Uses. Acad. Press, New York, pp 57–88
45. Zou X, Navon IM, Berger M, Phua KH, Schlick T, Dimet FX (1993) Numerical experience with limited-memory quasi-Newton and truncated Newton methods. SIAM J Optim 3:582–608

46. Zoutendijk G (1970) Nonlinear programming computational methods. In: Abadie J (ed) Integer and Nonlinear Programming. North-Holland, Amsterdam, pp 37–86

L-convex Functions and M-convex Functions

KAZUO MUROTA
Res. Institute Math. Sci. Kyoto University, Kyoto, Japan

MSC2000: 90C27, 90C25, 90C10, 90C35

Article Outline

Keywords

L-convexity; M-convexity; Discrete convex analysis; Submodular function; Matroid

In the field of nonlinear programming (in continuous variables), convex analysis [20,21] plays a pivotal role both in theory and in practice. An analogous theory for discrete optimization (nonlinear integer programming), called 'discrete convex analysis' [15,16], is developed for L-convex and M-convex functions by adapting the ideas in convex analysis and generalizing the results in matroid theory. The L- and M-convex functions are introduced in [15] and [12,18], respectively.

Definitions of L- and M-Convexity

Let V be a nonempty finite set and $\mathbf{Z}$ be the set of integers. For any function $g: \mathbf{Z}^V \to Z \cup \{+\infty\}$ define dom $g = \{p \in \mathbf{Z}^V : g(p) < +\infty\}$, called the *effective domain* of g.

A function $g: \mathbf{Z}^V \to \mathbf{Z} \cup \{+\infty\}$ with dom $g \neq \emptyset$ is called *L-convex* if

$$g(p) + g(q) \geq g(p \vee q) + g(p \wedge q) \quad (p, q \in \mathbb{Z}^V),$$
$$\exists r \in \mathbb{Z} : \; g(p + \mathbf{1}) = g(p) + r \quad (p \in \mathbb{Z}^V),$$

where $p \vee q = (\max(p(v), q(v)) \,|v \in V) \in \mathbf{Z}^V$, $p \wedge q = (\min p(v), q(v))|v \in V) \in \mathbf{Z}^V$, and $\mathbf{1}$ is the vector in $\mathbf{Z}^V$ with all components being equal to 1.

A set $D \subseteq \mathbf{Z}^V$ is said to be an *L-convex set* if its indicator function δ_D (defined by $\delta_D(p) = 0$ if $p \in D$, and $= +\infty$ otherwise) is an L-convex function, i. e., if

i) $D \neq \emptyset$;
ii) $p, q \in D \Rightarrow p \vee q, p \wedge q \in D$; and
iii) $p \in D \Rightarrow p \pm \mathbf{1} \in D$.

A function $f: \mathbf{Z}^V \to \mathbf{Z} \cup \{+\infty\}$ with dom $f \neq \emptyset$ is called *M-convex* if it satisfies

- M-EXC) For x, $y \in$ dom f and $u \in \text{supp}^+(x - y)$, there exists $v \in \text{supp}^-(x - y)$ such that

$$f(x) + f(y) \geq f(x - \chi_u + \chi_v) + f(y + \chi_u - \chi_v),$$

where, for any $u \in V$, χ_u is the *characteristic vector* of u (defined by $\chi_u(v) = 1$ if $v = u$, and $= 0$ otherwise), and

$$\text{supp}^+(z) = \{v \in V : \; z(v) > 0\} \quad (z \in \mathbb{Z}^V),$$
$$\text{supp}^-(z) = \{v \in V : \; z(v) < 0\} \quad (z \in \mathbb{Z}^V).$$

A set $B \subseteq \mathbf{Z}^V$ is said to be an *M-convex set* if its indicator function is an M-convex function, i. e., if B satisfies

- B-EXC) For $x, y \in B$ and for $u \in \text{supp}^+(x - y)$, there exists $v \in \text{supp}^-(x - y)$ such that $x - \chi_u + \chi_v \in B$ and $y + \chi_u - \chi_v \in B$.

This means that an M-convex set is the same as the set of integer points of the base polyhedron of an integral submodular system (see [8] for submodular systems).

L-convexity and M-convexity are conjugate to each other under the *integral Fenchel–Legendre transformation* $f \longmapsto f^\bullet$ defined by

$$f^\bullet(p) = \sup\{\langle p, x\rangle - f(x) : \; x \in \mathbb{Z}^V\}, \quad p \in \mathbb{Z}^V,$$

where $\langle p, x\rangle = \sum_{v \in V} p(v)\, x(v)$. That is, for L-convex function g and M-convex function f, it holds [15] that $g^\bullet$ is M-convex, $f^\bullet$ is L-convex, $g^{\bullet\bullet} = g$, and $f^{\bullet\bullet} = f$.

Example 1 (Minimum cost flow problem) L-convexity and M-convexity are inherent in the integer minimum-cost flow problem, as pointed out in [12,15]. Let $G = (V, A)$ be a graph with vertex set V and arc set A, and let $T \subseteq V$ be given. For $\xi: A \to Z$ its *boundary* $\partial\xi: V \to \mathbf{Z}$ is defined by

$$\partial\xi(v) = \sum \{\xi(a): a \in \delta^+ v\} - \sum \{\xi(a): a \in \delta^- v\} \quad (v \in V),$$

where $\delta^+ v$ and $\delta^- v$ denote the sets of out-going and incoming arcs incident to v, respectively. For $\widetilde{p}: V \to \mathbb{Z}$ its *coboundary* $\delta\widetilde{p}: A \to \mathbb{Z}$ is defined by

$$\delta\widetilde{p}(a) = \widetilde{p}(\partial^+ a) - \widetilde{p}(\partial^- a) \quad (a \in A),$$

where $\partial^+ a$ and $\partial^- a$ mean the initial and terminal vertices of a, respectively. Denote the class of one-dimensional discrete convex functions by

$$C_1 = \{\varphi: \mathbb{Z} \to \mathbb{Z} \cup \{+\infty\} \mid \operatorname{dom}\varphi \neq \emptyset, \varphi(t-1) + \varphi(t+1) \geq 2\varphi(t) \quad (t \in \mathbb{Z})\}.$$

For $\varphi_a \in \mathcal{C}_1$ $(a \in A)$, representing the arc-cost in terms of flow, the *total cost function* $f: \mathbf{Z}^T \to \mathbf{Z} \cup \{+\infty\}$ defined by

$$f(x) = \inf_{\xi} \left\{ \sum_{a \in A} \varphi_a(\xi(a)) : \begin{array}{c} \partial\xi(v) = -x(v) \\ (v \in T), \\ \partial\xi(v) = 0 \\ (v \in V \setminus T) \end{array} \right\} \quad (x \in \mathbb{Z}^T)$$

is M-convex, provided that $f > -\infty$ (i.e., f does not take the value of $-\infty$). For $\psi_a \in \mathcal{C}_1$ $(a \in A)$, representing the arc-cost in terms of tension, the total cost function $g: \mathbf{Z}^T \to \mathbf{Z} \cup \{+\infty\}$ defined by

$$g(p) = \inf_{\widetilde{p}} \left\{ \sum_{a \in A} \psi_a(\eta(a)) : \begin{array}{c} \eta = -\delta\widetilde{p}, \\ \widetilde{p}(v) = p(v) \\ (v \in T) \end{array} \right\} \quad (p \in \mathbb{Z}^T)$$

is L-convex, provided that $g > -\infty$. The two cost functions $f(x)$ and $g(p)$ are conjugate to each other in the sense that, if $\psi_a = \varphi_a^\bullet$ $(a \in A)$, then $g = f^\bullet$.

Example 2 (Polynomial matrix) Let $A(s)$ be an $m \times n$ matrix of rank m with each entry being a polynomial in a variable s, and let $\mathcal{B} \subseteq 2^V$ be the family of bases of $A(s)$ with respect to linear independence of the column vectors; namely, $J \subseteq V$ belongs to $\mathcal{B}$ if and only if $|J| = m$ and the column vectors with indices in J are linearly independent. Then $f: \mathbf{Z}^V \to \mathbf{Z} \cup \{+\infty\}$ defined by

$$f(x) = \begin{cases} -\deg_s \det A[J] & (x = \chi_J,\ J \in \mathcal{B}) \\ +\infty & (\text{otherwise}) \end{cases}$$

is M-convex, where $\chi_J \in \{0, 1\}^V$ is the characteristic vector of J (defined by $\chi_J(v) = 1$ if $v \in J$, and $= 0$ otherwise), $A[J]$ denotes the $m \times m$ submatrix with column indices in $J \in \mathcal{B}$, and $\deg_s(\cdot)$ means the degree as a polynomial in s. The Grassmann–Plücker identity implies the exchange property of f. This example was the motivation of valuated matroids in [2,3], which in turn can be identified with the negative of M-convex functions f with $\operatorname{dom} f \subseteq \{0, 1\}^V$.

For $p = (p(v))_{v \in V} \in \mathbf{Z}^V$ denote by $D(p)$ the diagonal matrix of order $n = |V|$ with diagonal elements $s^{p(v)}$ $(v \in V)$. Then the function $g: \mathbf{Z}^V \to \mathbf{Z}$ defined by

$$g(p) = \max \{\deg_s \det(A \cdot D(p))[J]: J \in \mathcal{B}\}$$

is L-convex [16], where $(A \cdot D(p))[J]$ means the $m \times m$ submatrix of $A \cdot D(p)$ with column indices in J. We have $g = f^\bullet$.

L-Convex Sets

An L-convex set $D \subseteq \mathbf{Z}^V$ has 'no holes' in the sense that $D = \overline{D} \cap \mathbb{Z}^V$, where $\overline{D}$ denotes the convex hull of D in $\mathbf{R}^V$. Hence it is natural to consider the polyhedral description of $\overline{D}$, 'L-convex polyhedron' (see [15,16]). For any function $\gamma: V \times V \to \mathbf{Z} \cup \{+\infty\}$ with $\gamma(v, v) = 0$ $(v \in V)$, define

$$\mathbf{D}(\gamma) = \left\{ p \in \mathbb{R}^V : \begin{array}{c} p(v) - p(u) \leq \gamma(u, v) \\ (\forall u, v \in V) \end{array} \right\}.$$

If $\mathbf{D}(\gamma) \neq \emptyset$, $\mathbf{D}(\gamma)$ is an integral polyhedron and $D = \mathbf{D}(\gamma) \cap \mathbf{Z}^V$ is an L-convex set. If γ satisfies triangle inequality:

$$\gamma(u, v) + \gamma(v, w) \geq \gamma(u, w) \quad (u, v, w \in V),$$

then $\mathbf{D}(\gamma) \neq \emptyset$ and

$$\gamma(u, v) = \sup \{p(v) - p(u): p \in \mathbf{D}(\gamma)\} \quad (u, v \in V).$$

Conversely, for any nonempty $D \subseteq \mathbf{Z}^V$,

$$\gamma(u, v) = \sup\{p(v) - p(u) : p \in D\} \quad (u, v \in V),$$

satisfies triangle inequality as well as $\gamma(v, v) = 0$ ($v \in V$), and if D is L-convex, then $\overline{D} = \mathbf{D}(\gamma)$. Thus there is a one-to-one correspondence between L-convex set D and function γ satisfying triangle inequality. In particular, $D \subseteq \mathbf{Z}^V$ is L-convex if and only if $D = \mathbf{D}(\gamma) \cap \mathbf{Z}^V$ for some γ satisfying triangle inequality. For L-convex sets $D_1, D_2 \subseteq \mathbf{Z}^V$, it holds that $D_1 + D_2 = \overline{D_1 + D_2} \cap \mathbb{Z}^V$ and $\overline{D_1} \cap \overline{D_2} = \overline{D_1 \cap D_2}$.

It is also true that a function γ satisfying triangle inequality corresponds one-to-one to a positively homogeneous M-convex function f (i. e., $f(\lambda\, x) = \lambda\, f(x)$ for $x \in \mathbf{Z}^V$ and $0 \leq \lambda \in \mathbf{Z}$). The correspondence $f \longmapsto \gamma$ is given by

$$\gamma(u, v) = f(\chi_v - \chi_u) \quad (u, v \in V),$$

whereas $\gamma \longmapsto f$ by

$$f(x) = \inf_{\lambda} \left\{ \sum_{u,v \in V} \lambda_{uv} \gamma(u, v) : \begin{array}{c} \sum_{u,v \in V} \lambda_{uv}(\chi_v - \chi_u) = x, \\ 0 \leq \lambda_{uv} \in \mathbb{Z} \\ (u, v \in V) \end{array} \right\} \quad (x \in \mathbb{Z}^V).$$

The correspondence between L-convex sets and positively homogeneous M-convex functions via functions with triangle inequality is a special case of the conjugacy relationship between L- and M-convex functions.

M-Convex Sets

An M-convex set $B \subseteq \mathbf{Z}^V$ has 'no holes' in the sense that $B = \overline{B} \cap \mathbb{Z}_V$. Hence it is natural to consider the polyhedral description of $\overline{B}$, 'M-convex polyhedron'. A set function $\rho: 2^V \to \mathbf{Z} \cup \{+\infty\}$ is said to be *submodular* if

$$\rho(X) + \rho(Y) \geq \rho(X \cup Y) + \rho(X \cap Y) \quad (X, Y \subseteq V),$$

where the inequality is satisfied if $\rho(X)$ or $\rho(Y)$ is equal to $+\infty$. It is assumed throughout that $\rho(\emptyset) = 0$ and $\rho(V) < +\infty$ for any set function $\rho: 2^V \to \mathbf{Z} \cup \{+\infty\}$. For a set function ρ, define

$$\mathbf{P}(\rho) = \left\{ x \in \mathbb{R}^V : \begin{array}{c} x(X) \leq \rho(X) \\ (\forall X \subset V), \\ x(V) = \rho(V) \end{array} \right\},$$

where $x(X) = \sum_{v \in X} x(v)$. If ρ is submodular, $\mathbf{P}(\rho)$ is a nonempty integral polyhedron, $B = \mathbf{P}(\rho) \cap \mathbf{Z}^V$ is an M-convex set, and

$$\rho(X) = \sup\{x(X) : x \in \mathbf{P}(\rho)\} \quad (X \subseteq V).$$

Conversely, for any nonempty $B \subseteq \mathbf{Z}^V$, define a set function ρ by

$$\rho(X) = \sup\{x(X) : x \in B\} \quad (X \subseteq V).$$

If B is M-convex, then ρ is submodular and $\overline{B} = \mathbf{P}(\rho)$. Thus there is a one-to-one correspondence between M-convex set B and submodular set function ρ. In particular, $B \subseteq \mathbf{Z}^V$ is M-convex if and only if $B = \mathbf{P}(\rho) \cap \mathbf{Z}^V$ for some submodular ρ. The correspondence $B \leftrightarrow \rho$ is a restatement of a well-known fact [4,8]. For M-convex sets $B_1, B_2 \subseteq \mathbf{Z}^V$, it holds that $B_1 + B_2 = \overline{B_1 + B_2} \cap \mathbb{Z}^V$ and $\overline{B_1} \cap \overline{B_2} = \overline{B_1 \cap B_2}$.

It is also true that a submodular set function ρ corresponds one-to-one to a positively homogeneous L-convex function g. The correspondence $g \longmapsto \rho$ is given by the restriction

$$\rho(X) = g(\chi_X) \quad (X \subseteq V)$$

(χ_X is the characteristic vector of X), whereas $\rho \longmapsto g$ by the Lovász extension (explained below). The correspondence between M-convex sets and positively homogeneous L-convex functions via submodular set functions is a special case of the conjugacy relationship between M- and L-convex functions.

For a set function $\rho: 2^V \to \mathbf{Z} \cup \{+\infty\}$, the *Lovász extension* [11] of ρ is a function $\widehat{\rho}: \mathbb{R}^V \to \mathbb{R} \cup \{+\infty\}$ defined by

$$\widehat{\rho}(p) = \sum_{j=1}^{n} (p_j - p_{j+1}) \rho(V_j) \quad (p \in \mathbb{R}^V),$$

where, for each $p \in \mathbf{R}^V$, the elements of V are indexed as $\{v_1, \ldots, v_n\}$ (with $n = |V|$) in such a way that $p(v_1) \geq \cdots \geq p(v_n)$; $p_j = p(v_j)$, $V_j = \{v_1, \ldots, v_j\}$ for $j = 1, \ldots, n$, and

$p_{n+1} = 0$. The right-hand side of the above expression is equal to $+\infty$ if and only if $p_j - p_{j+1} > 0$ and $\rho(V_j) = +\infty$ for some j with $1 \le j \le n-1$. The Lovász extension $\widehat{\rho}$ is indeed an extension of ρ, since $\widehat{\rho}(\chi_X) = \rho(X)$ for $X \subseteq V$.

The relationship between submodularity and convexity is revealed by the statement [11] that a set function ρ is submodular if and only if its Lovász extension $\widehat{\rho}$ is convex.

The restriction to $\mathbf{Z}^V$ of the Lovász extension of a submodular set function is a positively homogeneous L-convex function, and any positively homogeneous L-convex function can be obtained in this way [15].

Properties of L-Convex Functions

For any $g: \mathbf{Z}^V \to \mathbf{Z} \cup \{+\infty\}$ and $x \in \mathbf{R}^V$, define $g[-x]: \mathbf{Z}^V \to \mathbf{R} \cup \{+\infty\}$ by

$$g[-x](p) = g(p) - \langle p, x \rangle \quad (p \in \mathbb{Z}^V).$$

The set of the minimizers of $g[-x]$ is denoted as $\operatorname{argmin}(g[-x])$.

Let $g: \mathbf{Z}^V \to \mathbf{Z} \cup \{+\infty\}$ be L-convex. Then $\operatorname{dom} g$ is an L-convex set. For each $p \in \operatorname{dom} g$,

$$\rho_p(X) = g(p + \chi_X) - g(p) \quad (X \subseteq V)$$

is a submodular set function with $\rho_p(\emptyset) = 0$ and $\rho_p(V) < +\infty$.

An L-convex function g can be extended to a convex function $\overline{g}: \mathbb{R}^V \to \mathbb{R} \cup \{+\infty\}$ through the Lovász extension of the submodular set functions ρ_p for $p \in \operatorname{dom} g$. Namely, for $p \in \operatorname{dom} g$ and $q \in [0, 1]^V$, it holds [15] that

$$\overline{g}(p + q) = g(p) + \sum_{j=1}^{n} (q_j - q_{j+1})(g(p + \chi_{V_j}) - g(p)),$$

where, for each q, the elements of V are indexed as $\{v_1, \ldots, v_n\}$ (with $n = |V|$) in such a way that $q(v_1) \ge \cdots \ge q(v_n)$; $q_j = q(v_j)$, $V_j = \{v_1, \ldots, v_j\}$ for $j = 1, \ldots, n$, and $q_{n+1} = 0$. The expression of $\overline{g}$ shows that an L-convex function is an integrally convex function in the sense of [5].

An L-convex function g enjoys *discrete midpoint convexity*:

$$g(p) + g(q) \ge g\left(\left\lceil \frac{p+q}{2} \right\rceil\right) + g\left(\left\lfloor \frac{p+q}{2} \right\rfloor\right)$$

for $p, q \in \mathbf{Z}^V$, where $\lceil p \rceil$ (or $\lfloor p \rfloor$) for any $p \in \mathbf{R}^V$ denotes the vector obtained by rounding up (or down) the components of p to the nearest integers.

The minimum of an L-convex function g is characterized by the local minimality in the sense that, for $p \in \operatorname{dom} g$, $g(p) \le g(q)$ for all $q \in Z^V$ if and only if $g(p + \mathbf{1}) = g(p) \le g(p + \chi_X)$ for all $X \subseteq V$.

The minimizers of an L-convex function, if nonempty, form an L-convex set. For any $x \in \mathbf{R}^V$, $\operatorname{argmin}(g[-x])$, if nonempty, is an L-convex set. Conversely, this property characterizes L-convex functions under an auxiliary assumption.

A number of operations can be defined for L-convex functions [15,16]. For $x \in \mathbf{Z}^V$, $g[-x]$ is an L-convex function. For $a \in \mathbf{Z}^V$ and $\beta \in \mathbf{Z}$, $g(a + \beta p)$ is L-convex in p. For $U \subseteq V$, the projection of g to U:

$$g^U(p') = \inf\left\{g(p', p'') : p'' \in \mathbb{Z}^{V \setminus U}\right\} \quad (p' \in \mathbb{Z}^U)$$

is L-convex in p', provided that $g^U > -\infty$. For $\psi_v \in \mathcal{C}_1$ $(v \in V)$,

$$\widetilde{g}(p) = \inf_{q \in \mathbb{Z}^V}\left[g(q) + \sum_{v \in V} \psi_v(p(v) - q(v))\right]$$

is L-convex in $p \in \mathbf{Z}^V$, provided that $\widetilde{g} > -\infty$. The sum of two (or more) L-convex functions is L-convex, provided that its effective domain is nonempty.

Properties of M-Convex Functions

Let $f: \mathbf{Z}^V \to \mathbf{Z} \cup \{+\infty\}$ be M-convex. Then $\operatorname{dom} f$ is an M-convex set. For each $x \in \operatorname{dom} f$,

$$\gamma_x(u, v) = f(x - \chi_u + \chi_v) - f(x) \quad (u, v \in V)$$

satisfies [16] triangle inequality.

An M-convex function f can be extended to a convex function $\overline{f}: \mathbb{R}^V \to \mathbb{R} \cup \{+\infty\}$, and the value of $\overline{f}(x)$ for $x \in \mathbf{R}^V$ is determined by $\{f(y): y \in \mathbf{Z}^V, \lfloor x \rfloor \le y \le \lceil x \rceil\}$. That is, an M-convex function is an integrally convex function in the sense of [5].

The minimum of an M-convex function f is characterized by the local minimality in the sense that for $x \in \operatorname{dom} f$, $f(x) \le f(y)$ for all $y \in \mathbf{Z}^V$ if and only if $f(x) \le f(x - \chi_u + \chi_v)$ for all $u, v \in V$ [12,15,18].

The minimizers of an M-convex function, if nonempty, form an M-convex set. Moreover, for any

$p \in \mathbf{R}^V$, $\operatorname{argmin}(f[-p])$, if nonempty, is an M-convex set. Conversely, this property characterizes M-convex functions, under an auxiliary assumption that the effective domain is bounded or the function can be extended to a convex function over $\mathbf{R}^V$ (see [12,15]).

The level set of an M-convex function is not necessarily an M-convex set, but enjoys a weaker exchange property. Namely, for any $p \in \mathbf{R}^V$ and $\alpha \in \mathbf{R}$, $S = \{x \in \mathbf{Z}^V : f[-p](x) \leq \alpha\}$ (the level set of $f[-p]$) satisfies: For x, $y \in S$ and for $u \in \operatorname{supp}^+(x-y)$, there exists $v \in \operatorname{supp}^-(x-y)$ such that either $x - \chi_u + \chi_v \in S$ or $y + \chi_u - \chi_v \in S$. Conversely, this property characterizes M-convex functions [25].

A number of operations can be defined for M-convex functions [15,16]. For $p \in \mathbf{Z}^V$, $f[-p]$ is an M-convex function. For $a \in \mathbf{Z}^V$, $f(a-x)$ and $f(a+x)$ are M-convex in x. For $U \subseteq V$, the restriction of f to U:

$$f_U(x') = f(x', \mathbf{0}_{V \setminus U}) \quad (x' \in \mathbf{Z}^U)$$

(where $\mathbf{0}_{V \setminus U}$ is the zero vector in $\mathbf{Z}^{V \setminus U}$) is M-convex in x', provided that $\operatorname{dom} f_U \neq \emptyset$. For $\varphi_v \in \mathcal{C}_1$ $(v \in V)$,

$$\widetilde{f}(x) = f(x) + \sum_{v \in V} \varphi_v(x(v)) \quad (x \in \mathbf{Z}^V)$$

is M-convex, provided that $\operatorname{dom} \widetilde{f} \neq \emptyset$. In particular, a separable convex function $\widetilde{f}(x) = \sum_{v \in V} \varphi_v(x(v))$ with $\operatorname{dom} \widetilde{f}$ being an M-convex set is an M-convex function. For two M-convex functions f_1 and f_2, the integral convolution

$$(f_1 \square f_2)(x) = \inf \left\{ f_1(x_1) + f_2(x_2) : \begin{array}{c} x = x_1 + x_2 \\ x_1, x_2 \in \mathbf{Z}^V \end{array} \right\} \quad (x \in \mathbf{Z}^V)$$

is either M-convex or else $(f_1 \square f_2)(x) = \pm\infty$ for all $x \in \mathbf{Z}^V$.

Sum of two M-convex functions is not necessarily M-convex; such function with nonempty effective domain is called *M_2-convex*. Convolution of two L-convex functions is not necessarily L-convex; such function with nonempty effective domain is called *L_2-convex*. M_2- and L_2-convex functions are in one-to-one correspondence through the integral Fenchel–Legendre transformation.

$L^\natural$- and $M^\natural$-Convexity

$L^\natural$- and $M^\natural$-convexity are variants of, and essentially equivalent to, L- and M-convexity, respectively. $L^\natural$- and $M^\natural$-convex functions are introduced in [9] and [19], respectively.

Let v_0 be a new element not in V and define $\widetilde{V} = \{v_0\} \cup V$. A function $g: \mathbf{Z}^V \to \mathbf{Z} \cup \{+\infty\}$ with $\operatorname{dom} g \neq \emptyset$ is called *$L^\natural$-convex* if it is expressed in terms of an L-convex function $\widetilde{g}: \mathbb{Z}^{\widetilde{V}} \to \mathbb{Z} \cup \{+\infty\}$ as $g(p) = \widetilde{g}(0, p)$. Namely, an $L^\natural$-convex function is a function obtained as the restriction of an L-convex function. Conversely, an $L^\natural$-convex function determines the corresponding L-convex function up to the constant r in the definition of L-convex function.

An $L^\natural$-convex function is essentially the same as a submodular integrally convex function of [5], and hence is characterized by discrete midpoint convexity [9]. An L-convex function, enjoying discrete midpoint convexity, is an $L^\natural$-convex function.

Quadratic function

$$g(p) = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} p_i p_j \quad (p \in \mathbb{Z}^n)$$

with $a_{ij} = a_{ji} \in \mathbf{Z}$ is $L^\natural$-convex if and only if $a_{ij} \leq 0$ $(i \neq j)$ and $\sum_{j=1}^{n} a_{ij} \geq 0$ $(i = 1, \cdots, n)$. For $\{\psi_i \in \mathcal{C}_1 : i = 1, \ldots, n\}$, a separable convex function

$$g(p) = \sum_{i=1}^{n} \psi_i(p_i) \quad (p \in \mathbb{Z}^n)$$

is $L^\natural$-convex.

The properties of L-convex functions mentioned above are carried over, mutatis mutandis, to $L^\natural$-convex functions. In addition, the restriction of an $L^\natural$-convex function g to $U \subseteq V$, denoted g_U, is $L^\natural$-convex.

A subset of $\mathbf{Z}^V$ is called an *$L^\natural$-convex set* if its indicator function is an $L^\natural$-convex function. A set $E \subseteq \mathbf{Z}^V$ is an $L^\natural$-convex set if and only if

$$p, q \in E \quad \Rightarrow \quad \left\lceil \frac{p+q}{2} \right\rceil, \left\lfloor \frac{p+q}{2} \right\rfloor \in E.$$

A function $f: \mathbf{Z}^V \to \mathbf{Z} \cup \{+\infty\}$ with $\operatorname{dom} f \neq \emptyset$ is called *$M^\natural$-convex* if it is expressed in terms of an M-convex function $\widetilde{f}: \mathbb{Z}^{\widetilde{V}} \to \mathbb{Z} \cup \{+\infty\}$ as

$$\widetilde{f}(x_0, x) = \begin{cases} f(x) & \text{if } x_0 + \sum_{u \in V} x(u) = 0 \\ +\infty & \text{otherwise.} \end{cases}$$

Namely, an $M^\natural$-convex function is a function obtained as the projection of an M-convex function. Conversely, an $M^\natural$-convex function determines the corresponding M-convex function up to a translation of dom f in the direction of v_0. A function $f: \mathbf{Z}^V \to \mathbf{Z} \cup \{+\infty\}$ with dom $f \neq \emptyset$ is $M^\natural$-convex if and only if (see [19]) it satisfies

- $M^\natural$-EXC) For $x, y \in \mathrm{dom}\, f$ and $u \in \mathrm{supp}^+(x-y)$,

$$f(x)+f(y) \geq \min\Big[f(x-\chi_u)+f(y+\chi_u), \min_{v\in \mathrm{supp}^-(x-y)}\{f(x-\chi_u+\chi_v)+f(y+\chi_u-\chi_v)\}\Big].$$

Since M-EXC) implies $M^\natural$-EXC), an M-convex function is an $M^\natural$-convex function.

Quadratic function

$$f(x)=\sum_{i=1}^{n} a_i {x_i}^2 + b\sum_{i<j} x_i x_j \qquad (x \in \mathbb{Z}^n)$$

with $a_i \in \mathbf{Z}$ $(1 \leq i \leq n)$, $b \in \mathbf{Z}$ is $M^\natural$-convex if $0 \leq b \leq 2 \min_{1\leq i\leq n} a_i$ (cf. [19]). For $\{\varphi_i \in \mathcal{C}_1 : i = 0, \ldots, n\}$, a function of the form

$$f(x)=\varphi_0\left(\sum_{i=1}^{n} x_i\right)+\sum_{i=1}^{n}\varphi_i(x_i) \quad (x \in \mathbb{Z}^n)$$

is $M^\natural$-convex [19]; a separable convex function is a special case of this (with $\varphi_0 = 0$). More generally, for $\{\varphi_X \in \mathcal{C}_1 : X \in \mathcal{T}\}$ indexed by a laminar family $\mathcal{T} \subseteq 2^V$, the function

$$f(x)=\sum_{X\in\mathcal{T}} \varphi_X(x(X)) \quad (x \in \mathbb{Z}^V)$$

is $M^\natural$-convex [1], where $\mathcal{T}$ is called *laminar* if for any X, $Y \in \mathcal{T}$, at least one of $X \cap Y$, $X \setminus Y$, $Y \setminus X$ is empty.

The properties of M-convex functions mentioned above are carried over, mutatis mutandis, to $M^\natural$-convex functions. In addition, the projection of an $M^\natural$-convex function f to $U \subseteq V$, denoted f^U, is $M^\natural$-convex.

A subset of $\mathbf{Z}^V$ is called an *$M^\natural$-convex set* if its indicator function is an $M^\natural$-convex function. A set $Q \subseteq \mathbf{Z}^V$ is an $M^\natural$-convex set if and only if Q is the set of integer points of an integral generalized polymatroid (cf. [7] for generalized polymatroids).

As a consequence of the conjugacy between L- and M-convexity, $L^\natural$-convex functions and $M^\natural$-convex functions are conjugate to each other under the integral Fenchel–Legendre transformation.

Duality

Discrete duality theorems hold true for L-convex/concave and M-convex/concave functions. A function $g: \mathbf{Z}^V \to \mathbf{Z} \cup \{-\infty\}$ is called L-concave (respectively, $L^\natural$-, M-, or $M^\natural$-concave) if $-g$ is L-convex (respectively, $L^\natural$-, M-, or $M^\natural$-convex); dom g means the effective domain of $-g$. The concave counterpart of the discrete Fenchel–Legendre transform is defined as

$$g^\circ(p) = \inf\{\langle p, x\rangle - g(x) : x \in \mathbb{Z}^V\} \quad (p \in \mathbb{Z}^V).$$

A discrete separation theorem for L-convex/concave functions, named *L-separation theorem* [15] (see also [9]), reads as follows. Let $f: \mathbf{Z}^V \to \mathbf{Z} \cup \{+\infty\}$ be an $L^\natural$-convex function and $g: \mathbf{Z}^V \to \mathbf{Z} \cup \{-\infty\}$ be an $L^\natural$-concave function such that dom $f \cap$ dom $g \neq \emptyset$ or dom $f^\bullet \cap$ dom $g^\circ \neq \emptyset$. If $f(p) \geq g(p)$ $(p \in \mathbf{Z}^V)$, there exist $\beta^* \in \mathbf{Z}$ and $x^* \in \mathbf{Z}^V$ such that

$$f(p) \geq \beta^* + \langle p, x^*\rangle \geq g(p) \quad (p \in \mathbb{Z}^V).$$

Since a submodular set function can be identified with a positively homogeneous L-convex function, the L-separation theorem implies *Frank's discrete separation theorem* for a pair of sub/supermodular functions [6], which reads as follows. Let $\rho: 2^V \to \mathbf{Z} \cup \{+\infty\}$ and $\mu: 2^V \to \mathbf{Z} \cup \{-\infty\}$ be submodular and supermodular functions, respectively, with $\rho(\emptyset) = \mu(\emptyset) = 0$, $\rho(V) < +\infty$, $\mu(V) > -\infty$, where μ is called *supermodular* if $-\mu$ is submodular. If $\rho(X) \geq \mu(X)$ $(X \subseteq V)$, there exists $x^* \in \mathbf{Z}^V$ such that

$$\rho(X) \geq x^*(X) \geq \mu(X) \quad (X \subseteq V).$$

Another discrete separation theorem, *M-separation theorem* [12,15] (see also [9]), holds true for M-convex/concave functions. Namely, let $f: \mathbf{Z}^V \to \mathbf{Z} \cup \{+\infty\}$ be an $M^\natural$-convex function and $g: \mathbf{Z}^V \to \mathbf{Z} \cup \{-\infty\}$ be an $M^\natural$-concave function such that dom $f \cap$ dom $g \neq \emptyset$ or dom $f^\bullet \cap$ dom $g^\circ \neq \emptyset$. If $f(x) \geq g(x)$ $(x \in \mathbf{Z}^V)$, there exist $\alpha^* \in \mathbf{Z}$ and $p^* \in \mathbf{Z}^V$ such that

$$f(x) \geq \alpha^* + \langle p^*, x\rangle \geq g(x) \quad (x \in \mathbb{Z}^V).$$

The L- and M-separation theorems are conjugate to each other, while a self-conjugate statement can be made in the form of the *Fenchel-type duality* [12,15], as follows. Let $f: \mathbf{Z}^V \to \mathbf{Z} \cup \{+\infty\}$ be an $L^\natural$-convex function and $g: \mathbf{Z}^V \to \mathbf{Z} \cup \{-\infty\}$ be an $L^\natural$-concave function

such that $\mathrm{dom}\, f \cap \mathrm{dom}\, g \neq \emptyset$ or $\mathrm{dom}\, f^\bullet \cap \mathrm{dom}\, g^\circ \neq \emptyset$. Then it holds that

$$\inf\{f(p) - g(p) \colon\ p \in \mathbb{Z}^V\} = \sup\{g^\circ(x) - f^\bullet(x) \colon\ x \in \mathbb{Z}^V\}.$$

Moreover, if this common value is finite, the infimum is attained by some $p \in \mathrm{dom}\, f \cap \mathrm{dom}\, g$ and the supremum is attained by some $x \in \mathrm{dom}\, f^\bullet \cap \mathrm{dom}\, g^\circ$.

Example 3 Here is a simple example to illustrate the subtlety of discrete separation for discrete functions. Functions $f\colon \mathbf{Z}^2 \to \mathbf{Z}$ and $g\colon \mathbf{Z}^2 \to \mathbf{Z}$ defined by $f(x_1, x_2) = \max(0, x_1 + x_2)$ and $g(x_1, x_2) = \min(x_1, x_2)$ can be extended respectively to a convex function $\overline{f}\colon \mathbb{R}^2 \to \mathbb{R}$ and a concave function $\overline{g}\colon \mathbb{R}^2 \to \mathbb{R}$ according to the defining expressions. With $\overline{p} = (\frac{1}{2}, \frac{1}{2})$, we have $\overline{f}(x) \geq \langle \overline{p}, x\rangle \geq \overline{g}(x)$ for all $x \in \mathbf{R}^2$, and a fortiori, $f(x) \geq \langle \overline{p}, x\rangle \geq g(x)$ for all $x \in \mathbf{Z}^2$. However, there exists no integral vector $p \in \mathbf{Z}^2$ such that $f(x) \geq \langle p, x\rangle \geq g(x)$ for all $x \in \mathbf{Z}^2$. Note also that f is $\mathrm{M}^\natural$-convex and g is L-concave.

Network Duality

A conjugate pair of M- and L-convex functions can be transformed through a network ([12,16]; see also [23]). Let $G = (V, A)$ be a directed graph with arc set A and vertex set V partitioned into three disjoint parts as $V = V^+ \cup V^0 \cup V^-$. For $\varphi_a \in \mathcal{C}_1$ $(a \in A)$ and M-convex $f\colon \mathbf{Z}^{V^+} \to \mathbf{Z} \cup \{+\infty\}$, define $\widetilde{f}\colon \mathbb{Z}^{V^-} \to \mathbb{Z} \cup \{\pm\infty\}$ by

$$\widetilde{f}(y) = \inf_{\xi, x} \left\{ f(x) + \sum_{a \in A} \varphi_a(\xi(a)) \colon \begin{array}{c} \partial\xi = (x, 0, -y) \\ \in \mathbb{Z}^{V^+ \cup V^0 \cup V^-} \\ \xi \in \mathbb{Z}^A \end{array} \right\}.$$

For $\psi_a \in \mathcal{C}_1$ $(a \in A)$ and L-convex $g\colon \mathbf{Z}^{V^+} \to \mathbf{Z} \cup \{+\infty\}$, define $\widetilde{g}\colon \mathbb{Z}^{V^-} \to \mathbb{Z} \cup \{\pm\infty\}$ by

$$\widetilde{g}(q) = \inf_{\eta, p, r} \left\{ g(p) + \sum_{a \in A} \psi_a(\eta(a)) \colon \begin{array}{c} \eta = -\delta(p, r, q) \\ \eta \in \mathbb{Z}^A \\ (p, r, q) \in \mathbb{Z}^{V^+ \cup V^0 \cup V^-} \end{array} \right\}.$$

Then $\widetilde{f}$ is M-convex, provided that $\widetilde{f} > -\infty$, and $\widetilde{g}$ is L-convex, provided that $\widetilde{g} > -\infty$. If $g = f^\bullet$ and $\psi_a = \varphi_a^\bullet$ $(a \in A)$, then $\widetilde{g} = \widetilde{f}^\bullet$. A special case $(V^+ = V)$ of the last statement yields the network duality:

$$\inf \left\{ \Phi(x, \xi) \colon \begin{array}{c} \partial\xi = x, \\ x \in \mathbb{Z}^V, \\ \xi \in \mathbb{Z}^A \end{array} \right\} = \sup \left\{ \Psi(p, \eta) \colon \begin{array}{c} \eta = -\delta p, \\ p \in \mathbb{Z}^V, \\ \eta \in \mathbb{Z}^A \end{array} \right\},$$

where $\Phi(x, \xi) = f(x) + \sum_{a \in A} \varphi_a(\xi(a))$, $\Psi(p, \eta) = -g(p) - \sum_{a \in A} \psi_a(\eta(a))$ and the finiteness of $\inf \Phi$ or $\sup \Psi$ is assumed. The network duality is equivalent to the Fenchel-type duality.

Subdifferentials

The subdifferential of $f\colon \mathbf{Z}^V \to \mathbf{Z} \cup \{+\infty\}$ at $x \in \mathrm{dom}\, f$ is defined by $\{p \in \mathbf{R}^V \colon f(y) - f(x) \geq \langle p, y - x\rangle\ (\forall y \in \mathbf{Z}^V)\}$. The subdifferential of an L_2- or M_2-convex function forms an integral polyhedron. More specifically:

- The subdifferential of an L-convex function is an integral base polyhedron (an M-convex polyhedron).
- The subdifferential of an L_2-convex function is the intersection of two integral base polyhedra (M-convex polyhedra).
- The subdifferential of an M-convex function is an L-convex polyhedron.
- The subdifferential of an M_2-convex function is the Minkowski sum of two L-convex polyhedra.

Similar statements hold true with L and M replaced respectively by $\mathrm{L}^\natural$ and $\mathrm{M}^\natural$.

Algorithms

On the basis of the equivalence of $\mathrm{L}^\natural$-convex functions and submodular integrally convex functions, the minimization of an L-convex function can be done by the algorithm of [5], which relies on the ellipsoid method. The minimization of an M-convex function can be done by purely combinatorial algorithms; a greedy-type algorithm [2] for valuated matroids and a domain reduction-type polynomial time algorithm [24] for M-convex functions. Algorithms for duality of M-convex functions (in other words, for M_2-convex functions) are also developed; polynomial algorithms [14,22] for valuated matroids, and a finite primal algorithm [18] and a polynomial time conjugate-scaling algorithm [10] for the submodular flow problem.

Applications

A discrete analog of the conjugate duality framework [21] for nonlinear optimization is developed in [15]. An application of M-convex functions to engineering system analysis and matrix theory is in [13,17]. M-convex functions find applications also in mathematical economics [1].

See also

- ► Generalized Concavity in Multi-objective Optimization
- ► Invexity and its Applications
- ► Isotonic Regression Problems

References

1. Danilov V, Koshevoy G, Murota K (May 1998) Equilibria in economies with indivisible goods and money. RIMS Preprint Kyoto Univ 1204
2. Dress AWM, Wenzel W (1990) Valuated matroid: A new look at the greedy algorithm. Appl Math Lett 3(2):33–35
3. Dress AWM, Wenzel W (1992) Valuated matroids. Adv Math 93:214–250
4. Edmonds J (1970) Submodular functions, matroids and certain polyhedra. In: Guy R, Hanani H, Sauer N, Schönheim J (eds) Combinatorial Structures and Their Applications. Gordon and Breach, New York, pp 69–87
5. Favati P, Tardella F (1990) Convexity in nonlinear integer programming. Ricerca Oper 53:3–44
6. Frank A (1982) An algorithm for submodular functions on graphs. Ann Discret Math 16:97–120
7. Frank A, Tardos É (1988) Generalized polymatroids and submodular flows. Math Program 42:489–563
8. Fujishige S (1991) Submodular functions and optimization, vol 47. North-Holland, Amsterdam
9. Fujishige S, Murota K (2000) Notes on L-/M-convex functions and the separation theorems. Math Program 88:129–146
10. Iwata S, Shigeno M (1998) Conjugate scaling technique for Fenchel-type duality in discrete optimization. IPSJ SIG Notes 98-AL-65
11. Lovász L (1983) Submodular functions and convexity. In: Bachem A, Grötschel M, Korte B (eds) Mathematical Programming – The State of the Art. Springer, Berlin, pp 235–257
12. Murota K (1996) Convexity and Steinitz's exchange property. Adv Math 124:272–311
13. Murota K (1996) Structural approach in systems analysis by mixed matrices – An exposition for index of DAE. In: Kirchgässner K, Mahrenholtz O, Mennicken R (eds) ICIAM 95. Math Res. Akad. Verlag, Berlin, pp 257–279
14. Murota K (1996) Valuated matroid intersection, I: optimality criteria, II: algorithms. SIAM J Discret Math 9:545–561, 562–576
15. Murota K (1998) Discrete convex analysis. Math Program 83:313–371
16. Murota K (1998) Discrete convex analysis. In: Fujishige S (ed) Discrete Structures and Algorithms, vol V. Kindai-Kagaku-sha, Tokyo, pp 51–100 (In Japanese.)
17. Murota K (1999) On the degree of mixed polynomial matrices. SIAM J Matrix Anal Appl 20:196–227
18. Murota K (1999) Submodular flow problem with a nonseparable cost function. Combinatorica 19:87–109
19. Murota K, Shioura A (1999) M-convex function on generalized polymatroid. Math Oper Res 24:95–105
20. Rockafellar RT (1970) Convex analysis. Princeton Univ. Press, Princeton
21. Rockafellar RT (1974) Conjugate duality and optimization. SIAM Regional Conf Appl Math, vol 16. SIAM, Philadelphia
22. Shigeno M (1996) A dual approximation approach to matroid optimization problems. PhD Thesis Tokyo Inst. Techn.
23. Shioura A (1998) A constructive proof for the induction of M-convex functions through networks. Discrete Appl Math 82:271–278
24. Shioura A (1998) Minimization of an M-convex function. Discrete Appl Math 84:215–220
25. Shioura A (2000) Level set characterization of M-convex functions. IEICE Trans Fundam Electronics, Commun and Comput Sci E83-A:586–589

LCP: Pardalos–Rosen Mixed Integer Formulation

PANOS M. PARDALOS
Center for Applied Optim. Department Industrial and Systems Engineering, University Florida, Gainesville, USA

MSC2000: 90C33, 90C11

Article Outline

Keywords

Linear complementarity problem; Mixed integer programming; Bimatrix games; Mixed integer problem; Minimum norm solution

In this article we consider the general *linear complementarity problem* (LCP) of finding a vector $x \in \mathbf{R}^n$ such that

$$Mx + q \geq 0, \quad x \geq 0, \quad x^\top Mx + q^\top x = 0$$

(or proving that such an x does not exist), where M is an $n \times n$ rational matrix and $q \in \mathbf{R}^n$ is a rational vector. For given data M and q, the problem is generally denoted by LCP(M, q). The LCP unifies a number of important problems in operations research. In particular, it generalizes the primal-dual linear programming problem, convex quadratic programming, and *bimatrix games* [1,2].

For the general matrix M, where $S = \{x: Mx + q \geq 0, x \geq 0\}$ can be bounded or unbounded, the LCP can always be solved by solving a specific zero-one, linear, *mixed integer problem* with n zero-one variables. Consider the following mixed zero-one integer problem:

$$\text{(MIP)} \quad \begin{cases} \max\limits_{\alpha,y,z} & \alpha \\ \text{s.t.} & 0 \leq My + \alpha q \leq e - z, \\ & \alpha \geq 0, \quad 0 \leq y \leq z, \\ & z \in \{0,1\}^n. \end{cases}$$

Theorem 1 *Let (α^*, y^*, z^*) be any optimal solution of (MIP). If $\alpha^* > 0$, then $x^* = y^*/\alpha^*$ solves the LCP. If in the optimal solution $\alpha^* = 0$, then the LCP has no solution.*

The equivalent mixed integer programming formulation (MIP) was first given in [3]. Every feasible point (α, y, z) of (MIP), with $\alpha > 0$, corresponds to a solution of LCP. Therefore, solving (MIP), we may generate several solutions of the corresponding LCP. J.B. Rosen [4] proved that the solution obtained by solving (MIP) is the *minimum norm solution* to the linear complementarity problem.

See also

- Branch and Price: Integer Programming with Column Generation
- Convex-simplex Algorithm
- Decomposition Techniques for MILP: Lagrangian Relaxation
- Equivalence Between Nonlinear Complementarity Problem and Fixed Point Problem
- Generalized Nonlinear Complementarity Problem
- Integer Linear Complementary Problem
- Integer Programming
- Integer Programming: Algebraic Methods
- Integer Programming: Branch and Bound Methods
- Integer Programming: Branch and Cut Algorithms
- Integer Programming: Cutting Plane Algorithms
- Integer Programming Duality
- Integer Programming: Lagrangian Relaxation
- Lemke Method
- Linear Complementarity Problem
- Linear Programming
- Mixed Integer Classification Problems
- Multi-objective Integer Linear Programming
- Multi-objective Mixed Integer Programming
- Multiparametric Mixed Integer Linear Programming
- Order Complementarity
- Parametric Linear Programming: Cost Simplex Algorithm
- Parametric Mixed Integer Nonlinear Optimization
- Principal Pivoting Methods for Linear Complementarity Problems
- Sequential Simplex Method
- Set Covering, Packing and Partitioning Problems
- Simplicial Pivoting Algorithms for Integer Programming
- Stochastic Integer Programming: Continuity, Stability, Rates of Convergence
- Stochastic Integer Programs
- Time-dependent Traveling Salesman Problem
- Topological Methods in Complementarity Theory

References

1. Cottle RW, Dantzig GB (1968) Complementarity pivot theory of mathematical programming. In: Dantzig GB, Veinott AF (eds) Mathematics of the Decision Sci., Part 1. Amer. Math. Soc., Providence, RI, pp 115–136
2. Horst R, Pardalos PM, Thoai NV (1995) Introduction to global optimization. Kluwer, Dordrecht
3. Pardalos PM, Rosen JB (1988) Global optimization approach to the linear complementarity problem. SIAM J Sci Statist Comput 9(2):341–353
4. Rosen JB (1990) Minimum norm solution to the linear complementarity problem. In: Leifman LJ (ed) Functional Analysis, Optimization and Mathematical Economics. Oxford Univ. Press, Oxford, pp 208–216

Least-index Anticycling Rules

LindAcR

TAMÁS TERLAKY
Department Comput. & Software,
McMaster University, West Hamilton, Canada

MSC2000: 90C05, 90C33, 90C20, 05B35

Article Outline

Keywords

Pivot rules; Anticycling; Least-index; Recursion; Oriented matroids

From the early days of mathematical optimization people were looking for simple rules that ensure that certain algorithms terminate in a finite number of steps. Specifically, on combinatorial structures the lack of finite termination imply that the algorithm cycles, i. e. periodically visits the same solutions. That is why rules ensuring finite termination of algorithms on finite structures are frequently referred to as *anticycling rules*.

One frequently used anticycling rule in linear optimization (cf. ► Linear programming) is the so-called *lexicographic pivoting rule* [9]. The other large class of anticycling procedures, the 'least-index' rules, is the subject of this paper. least-index rules were designed for network flow problems, linear optimization problems, linear complementarity problems and oriented matroid programming problems. These classes will be considered in the sequel.

Consistent Labeling For the Max-Flow Problem

The *maximal flow problem* (see e. g. [11]; [24]) is one of the basic problems of mathematical programming. The problem is given as follows. A *directed capacitated network* (N, A, u) is given, where N, the set of nodes, is a finite set; $A \subset N \times N$ is the set of directed arcs; finally, $u \in \mathbf{R}^A$ denotes the nonnegative capacity upper bound for flows through the arcs. Let further $s, t \in N$ be specified as the source and the sink in the network. A vector $f \in \mathbf{R}^A$ is a *flow* in the network, if the incoming flow at each node, different from the source and the sink, is equal to the flow going out from the node. The goal is to find a maximal flow, namely a flow for which the total flow flowing out of the source or, equivalently, flowing in to the sink is the largest possible. The *Ford–Fulkerson algorithm* is the best known algorithm to find such a maximal flow. It is based on generating augmenting path's subsequently. A *path* P connecting the source s and the sink t is a finite subset of arcs, where the source is the tail of the first arc; the sink is the head of the last arc; finally, the tail of an arc is always equal to the head of its predecessor. For ease of simplicity let us assume that if $(v^1, v^2) \in A$, then $(v^2, v^1) \in A$ as well. If the opposite arc were not present, we can introduce it with zero capacity.

0	Initialization. Let f be equal to zero. Let a free capacity network $(N, \overline{A}, \overline{u})$ be defined. Initially let $\overline{A} = \{a \in A \ : \ u_a > 0\}$ and $\overline{u} = u$.
1	Augmenting path. Let P be a path from s to t in the free capacity network. IF no such path exists, THEN STOP; A maximal flow is obtained.
2	Augmenting the flow. Let ϑ be the minimum of the arc capacities along the path P. Clearly $\vartheta > 0$. Increase the flow f on each arc of P by ϑ.
3	Update the free-capacity network. Decrease (increase) $\overline{u}_a$ by ϑ if the (opposite) of arc a is on the path P. Let $\overline{A} = \{a \in A \ : \ \overline{u}_a > 0\}$. Go to Step 1.

The Ford–Fulkerson max-flow algorithm

At each iteration cycle the flow value strictly increases. Thus, if the vector u is integral and the max-flow problem is bounded, then the Ford–Fulkerson algorithm provides a maximal flow in a finite number of steps. However, if the vector u contains irrational com-

ponents, then the algorithm does not terminate in a finite number of steps and, even worse, it might converge to a nonoptimal flow. For such an example see [11,24]. An elegant solution for this problem is the *consistent labeling algorithm* of A.W. Tucker [28]. This most simple refinement reads as follows:

> Be consistent at any time during the algorithm, specifically when building the augmenting path by using the *labeling procedure*. Whenever a labeled but unscanned subset of nodes is given during the procedure pick always the same from the same subset to be scanned.
> Particularly, if we assign an index to each node, then we are supposed to choose always the least-indexed node among the possibilities.

Tucker writes [28]: 'Fulkerson (unpublished) conjectured that a consistent labeling procedure would be polynomially bounded; a proof of this conjecture appears to be very difficult.'

Linear Optimization

Before discussing the general LO problem, first the linear *feasibility problem* is considered.

0	Initialization. Let $T(B)$ be an arbitrary basis tableau and fix an arbitrary ordering of the variables.
1	Leaving variable selection. Let K_P be the set of the indices of the infeasible variables in the basis. IF $K_P = \emptyset$, THEN STOP; the feasibility problem is solved. ELSE, let p be the least-index in K_P and then x_p will leave the basis.
2	Entering variable selection. Let K_D be the set of the column indices of the negative elements in row p of $T(B)$. IF $K_D = \emptyset$, THEN STOP; Row p of the tableau $T(B)$ gives an evidence that the feasibility problem is inconsistent and row p of the inverse basis is a solution of the alternative system. ELSE, let q be the least-index in K_D and then x_q will enter the basis.
3	Basis transformation. Pivot on (p, q). Go to Step 1.

Pivot rule

Least-Index Rules for Feasibility Problem

The feasibility problem

$$Ax = b, \quad x \geq 0,$$

and its alternative pair

$$b^\top y > 0, \quad A^\top y \leq 0,$$

can be solved by a very simple least-index pivot algorithm. A fundamental result, the so-called *Farkas lemma* (cf. also ▶ Farkas lemma; ▶ Farkas lemma: Generalizations) [10] says that exactly one of the two alternative systems has a solution. This result is also known as the *theorem of the alternatives*. When a simple finite pivot rule gives a solution to either of the two alternatives, an elementary constructive proof for the Farkas lemma and its relatives is obtained. The above simple finite least-index pivot rule for the feasibility problem is a special case (see below) of Bland's algorithm [5]. It is taken from [19] where the role of pivoting, and specifically the role of finite, least-index pivot rules in linear algebra is explored.

The Linear Optimization Problem

The general linear optimization (LO), linear programming (cf. ▶ Linear programming), problem will be considered in the standard primal form

$$\min \left\{ c^\top x : \; Ax = b, \; x \geq 0 \right\},$$

together with its standard dual

$$\max \left\{ b^\top y : \; A^\top y \leq c \right\}.$$

One of the most efficient, and for a long time the only, practical method to solve LO problems was the simplex method of G.B. Dantzig. The *simplex method* is a *pivot algorithm* that traverses through feasible *basic solutions* while the objective value is improving. The simplex method is in practice one of the most efficient algorithms but it is theoretically a finite algorithm only for *nondegenerate problems*.

A basis is called *primal degenerate* if at least one of the basic variables is zero; it is called *dual degenerate* if the reduced cost of at least one nonbasic variable is zero. In general, the basis is degenerate if it is either primal or dual, or both primal and dual degenerate. The LO problem is degenerate, if it has a degenerate ba-

sis. A pivot is called degenerate when after the pivot the objective remains unchanged. When the problem is *degenerate* the objective might stay the same in subsequent iterations and the simplex algorithm may *cycle*, i. e. starting from a basis, after some iterations the same basis is revisited and this process is repeated endlessly. Because the simplex method produces a sequence with monotonically improving objective values, the objective stays constant in a cycle, thus each pivot in the cycle must be degenerate. The possibility of cycling was recognized shortly after the invention of the simplex algorithm. Cycling examples were given by E.M.L. Beale [2] and by A.J. Hoffman [17]. Recently (1999) a scheme to construct cycling LO examples is presented in [15]. These examples made evident that extra techniques are needed to ensure finite termination of simplex methods. The first and widely used such tool is the class of lexicographic pivoting rules (cf. ► Lexicographic pivoting rules). Other, more recent techniques are the *least-index anticycling rules* and some more general recursive schemes.

Least-Index Pivoting Methods for LO

Cycling of the simplex method is possible only when the LO problem is degenerate. In that case not only many variables might be eligible to enter, but also to leave the basis. The least-index primal simplex rule makes the selection of both the entering and the leaving variable uniquely determined. Least-index rules are based on consistent selection among the possibilities. The first such rule for the simplex method was published by R.G. Bland [4,5].

The least-index simplex method is finite. The finiteness proofs are quite elementary. All are based on the simple fact that there is a finite number of different basis tableaus. Further, orthogonality of the primal and dual spaces on some recursive argumentation is used [4, 5,27]

It is straightforward to derive the least-index dual simplex algorithm. The only restriction relative to the dual simplex algorithm is, that when there are more candidates to leave or to enter the basis, always the least-indexed candidate has to be selected.

An interesting use of least index-resolution is used in [18] by designing finite primal-dual type Hungarian methods for LO. Note that finite criss-cross rules (cf. also ► Criss-cross pivoting rules) [14,26] make maximum possible use of least-index resolution.

0	Initialization Let $T(B)$ be a given primal feasible basis tableau and fix an arbitrary ordering of the variables.
1	Entering variable selection. Let K_D be the set of the indices of the dual infeasible variables, i.e. those with negative reduced cost. IF $K_D = \emptyset$, THEN STOP; The tableau $T(B)$ is optimal and this way a pair of solutions is obtained. ELSE, let q be the least-index in K_D and x_q, will enter the basis.
2	Leaving variable selection. Let K_P be the set of the indices of those candidate pivot elements in column q that satisfy the usual pivot selection conditions of the primal simplex method. IF $K_P = \emptyset$, THEN STOP; the primal problem is unbounded, and so the dual problem is infeasible. ELSE, let p be the least-index in K_P and then x_p will leave the basis.
3	Basis transformation. Pivot on (p, q). Go to Step 1.

The least-index primal simplex rule

Least-index simplex methods are not polynomial, they might require exponential number of steps to solve a LO problem, as it was shown by D. Avis and V. Chvátal [1]. Their example is essentially the Klee–Minty polytope [21]. Another example, again on the Klee–Minty polytope, is Roos's exponential example [25] for the least-index criss-cross method. Here the initial basis is feasible and, although it is not required, feasibility happens to be preserved, thus the criss-cross method reduces to a least index simplex method.

Linear Complementarity Problems

A linear complementarity problem (cf. ► Linear complementarity problem) (LCP) is given as follows:

$$-Mx + s = t, \quad x, s \geq 0, \quad x^\top s = 0.$$

Pivot algorithms are looking for a *complementary basis* solution of the LCP. A basis is called complementary, if

exactly one of the complementary variables x_i and s_i for all i is in the basis.

The solvability of LCP depends on the properties of the matrix M. One of the simplest case is when M is a *P-matrix*. The matrix M is a P-matrix if all of its principal minors are positive. K.G. Murty [22] presented an utmost simple finite pivot algorithm for solving the *P*-matrix LCP. This algorithm is a least-index *principal pivot algorithm*.

Two extremal behaviors, exponential in the worst case and polynomial in average, of this finite pivot rule is studied in [13].

Finite least-index pivot rules are developed for larger classes of LCPs. All are least-index principal pivoting methods, some more classical feasibility preserving simplex type methods [7,8,23], others are least-index criss-cross pivoting rules (cf. ► Criss-cross pivoting rules) [6,16,20]. More details are given in ► Principal pivoting methods for linear complementarity problems.

0	Initialization. Let $T(B)$ be complementary basis tableau and fix an arbitrary ordering of the variables. (We can choose $x = 0$, $s = t$ i.e., x nonbasic, s basic.)
1	Leaving variable selection. Let K be the set of the infeasible variables. IF $K = \emptyset$, THEN STOP; a complementary solution for LCP is obtained. ELSE, let p be the least-index in K.
2	Basis transformation. Pivot on (p, p), i.e. replace the least-indexed infeasible variable in the basis by its complementary pair. Go to Step 1.

Murty's Bard-type schema

Least-Index Rules and Oriented Matroids

The least-index simplex method was originally designed for oriented matroid linear programming (cf. also ► Oriented matroids) [3,4]. It turned soon out, that this is *not* a finite algorithm in the oriented matroid context. The reason is the possibility of *nondegenerate cycling* [3,12], a phenomenon what is impossible in the linear case. An apparent difference between the linear and the oriented matroid context is that for oriented matroids none of the finite-, recursive- or least-index-type rules yield a simplex method, i. e. a pivot method that preserves feasibility of the basis throughout. This discrepancy is also due to the possibility of nondegenerate cycling.

See also

- ► Criss-cross Pivoting Rules
- ► Lexicographic Pivoting Rules
- ► Linear Programming
- ► Pivoting Algorithms for Linear Programming Generating Two Paths
- ► Principal Pivoting Methods for Linear Complementarity Problems
- ► Probabilistic Analysis of Simplex Algorithms

References

1. Avis D, Chvátal V (1978) Notes on Bland's rule. Math Program Stud 8:24–34
2. Beale EML (1955) Cycling in the dual simplex algorithm. Naval Res Logist Quart 2:269–275
3. Bjorner A, Las Vergnas M, Sturmfels B, White N, Ziegler G (1993) Oriented matroids. Cambridge Univ. Press, Cambridge
4. Bland RG (1977) A combinatorial abstraction of linear programming. J Combin Th B 23:33–57
5. Bland RG (1977) New finite pivoting rules for the simplex method. Math Oper Res 2:103–107
6. Chang YY (1979) Least index resolution of degeneracy in linear complementarity problems. Techn Report Dept Oper Res Stanford Univ 79-14
7. Chang YY, Cottle RW (1980) Least index resolution of degeneracy in quadratic programming. Math Program 18:127–137
8. Cottle R, Pang JS, Stone RE (1992) The linear complementarity problem. Acad. Press, New York
9. Dantzig GB (1963) Linear programming and extensions. Princeton Univ. Press, Princeton
10. Farkas J (1902) Theorie der Einfachen Ungleichungen. J Reine Angew Math 124:1–27
11. Ford LR Jr, Fulkerson DR (1962) Network flows. Princeton Univ. Press, Princeton
12. Fukuda K (1982) Oriented matroid programming. PhD Thesis Waterloo Univ.
13. Fukuda K, Namiki M (1994) On extremal behaviors of Murty's least index method. Math Program 64:365–370
14. Fukuda K, Terlaky T (1997) Criss-cross methods: A fresh view on pivot algorithms. In: Mathematical Programming, (B) Lectures on Mathematical Programming, ISMP97, vol 79. Lausanne, pp 369–396

15. Hall J, McKinnon KI (1998) A class of cycling counter-examples to the EXPAND anti-cycling procedure. Techn. Report Dept. Math. and Statist. Univ. Edinburgh
16. Den Hertog D, Roos C, Terlaky T (1993) The linear complementarity problem, sufficient matrices and the criss-cross method. LAA 187:1–14
17. Hoffman AJ (1953) Cycling in the simplex method. Techn Report Nat Bureau Standards 2974
18. Klafszky E, Terlaky T (1989) Variants of the Hungarian method for solving linear programming problems. Math Oper Statist Ser Optim 20:79–91
19. Klafszky E, Terlaky T (1991) The role of pivoting in proving some fundamental theorems of linear algebra. LAA 151:97–118
20. Klafszky E, Terlaky T (1992) Some generalizations of the criss-cross method for quadratic programming. Math Oper Statist Ser Optim 24:127–139
21. Klee V, Minty GJ (1972) How good is the simplex algorithm? In: Shisha O (ed) Inequalities-III. Acad. Press, New York, pp 1159–1175
22. Murty KG (1974) A note on a Bard type scheme for solving the complementarity problem. Oper Res 11(2–3):123–130
23. Murty KG (1988) Linear complementarity, linear and nonlinear programming. Heldermann, Berlin
24. Murty KG (1992) Network programming. Prentice-Hall, Englewood Cliffs, NJ
25. Roos C (1990) An exponential example for Terlaky's pivoting rule for the criss-cross simplex method. Math Program 46:78–94
26. Terlaky T (1985) A convergent criss-cross method. Math Oper Statist Ser Optim 16(5):683–690
27. Terlaky T, Zhang S (1993) Pivot rules for linear programming: A survey on recent theoretical developments. Ann Oper Res 46:203–233
28. Tucker A (1977) A note on convergence of the Ford–Fulkerson flow algorithm. Math Oper Res 2(2):143–144

Least Squares Orthogonal Polynomials

Claude Brezinski, Ana C. Matos
Lab. d'Anal. Numérique et d'Optimisation,
Université Sci. et Techn. Lille Flandres–Artois,
Lille, France

MSC2000: 33C45, 65K10, 65F20, 65F22

Article Outline

Keywords

Orthogonal polynomials; Least squares; Padé-type approximation; Quadrature methods

Let c be the linear functional on the space of complex polynomials defined by

$$c(x^i) = \begin{cases} c_i \in \mathbb{C}, & i = 0, 1, \dots, \\ 0, & i < 0. \end{cases}$$

It is said that $\{P_k\}$ forms a family of (formal) *orthogonal polynomials* with respect to c if $\forall k$:

- P_k has the exact degree k,
- $c(x^i P_k(x)) = 0$ for $i = 0, \dots, k-1$.

Such a family exists if, $\forall k$, the Hankel determinant

$$H_k^{(0)} = \begin{vmatrix} c_0 & c_1 & \cdots & c_{k-1} \\ c_1 & c_2 & \cdots & c_k \\ \cdots & \cdots & \cdots & \cdots \\ c_{k-1} & c_k & \cdots & c_{2k-2} \end{vmatrix}$$

is different from zero. Such polynomials enjoy most of the properties of the usual orthogonal polynomials, when the functional c is given by

$$c(x^i) = \int_a^b x^i \, d\alpha(x),$$

where α is bounded and non decreasing in $[a, b]$ (see [1] for these properties). In this paper we study the polynomials R_k such that

$$\sum_{i=0}^{m} \left[c(x^i R_k(x))\right]^2$$

is minimized, where m is an integer strictly greater than $k-1$ (since, for $m = k-1$, we recover the previous formal orthogonal polynomials) and which can possibly depend on k. They will be called *least squares (formal) orthogonal polynomials*. They depend on the value of m but for simplicity this dependence will not be indicated in our notations.

Such polynomials arise naturally in problems of *Padé approximation* for power series with perturbed

coefficients, and in *Gaussian quadrature* (as described in the last section). Some properties of these polynomials are derived, together with a recursive scheme for their computation.

Existence and Uniqueness

Since the polynomials R_k will be defined apart from a multiplying factor, and since it is asked that the degree of R_k is exactly k we shall write

$$R_k(x) = b_0 + b_1 x + \cdots + b_k x^k$$

with $b_k = 1$. We set

$$\Phi(b_0, \ldots, b_{k-1}) = \sum_{i=0}^{m} [c(x^i R_k(x))]^2$$

and we seek for the values of $b_0, \ldots, b_{k-1}$ that minimize this quantity. That is, such that

$$\frac{\partial \Phi}{\partial b_j} = 0 \quad \text{for } j = 0, \ldots, k-1. \tag{1}$$

Setting $\gamma_n = (c_n, \ldots, c_{n+m})^\mathsf{T}$, this system can be written

$$b_0(\gamma_0, \gamma_j) + \cdots + b_{k-1}(\gamma_{k-1}, \gamma_j) = -(\gamma_k, \gamma_j) \tag{2}$$

for $j = 0, \ldots, k-1$. Thus R_k exists and is unique if and only if the matrix A_k of this system is non singular. Setting $X = (1, x, \ldots, x^{k-1})$ and calling the right-hand side of the preceding system γ we see that

$$R_k(x) = \frac{\begin{vmatrix} A_k & \gamma \\ X & x^k \end{vmatrix}}{|A_k|}.$$

If we set

$$B_k = \begin{pmatrix} c_0 & \cdots & c_{k-1} \\ \cdots & \cdots & \cdots \\ c_m & \cdots & c_{m+k-1} \end{pmatrix},$$

then $A_k = B_k^\top B_k$, $\gamma = B_k^\top \gamma_k$ and we recover the usual solution of a system of linear equations in the *least squares* sense.

Computation

The polynomials R_k can be recursively computed by inverting the matrix A_k of the above system (2) by the *bordering method*, see [5]. This method is as follows. Set

$$A_{k+1} = \begin{pmatrix} A_k & u_k \\ v_k & a_k \end{pmatrix}$$

where u_k is a column vector, v_k a row vector and a_k a scalar. We then have

$$A_{k+1}^{-1} = \begin{pmatrix} A_k^{-1} + A_k^{-1} u_k \beta_k^{-1} v_k A_k^{-1} & -A_k^{-1} u_k \beta_k^{-1} \\ -\beta_k^{-1} v_k A_k^{-1} & \beta_k^{-1} \end{pmatrix},$$

where $\beta_k = a_k - v_k A_k^{-1} u_k$.

Instead of choosing the normalization $b_k = 1$ we could impose the condition $b_0 = 1$. In that case we have the system

$$b_1'(\gamma_1, \gamma_j) + \cdots + b_k'(\gamma_k, \gamma_j) = -(\gamma_0, \gamma_j) \tag{3}$$

for $j = 1, \ldots, k$, and the bordering method can be used not only for computing the inverses of the matrices of the system recursively but also for obtaining its solution, since the new right-hand side contains the previous one.

Let $A_k{}'$ be the matrix of (3) and $d_k{}'$ be the right-hand side. We then have

$$A'_{k+1} = \begin{pmatrix} A'_k & u'_k \\ v'_k & a'_k \end{pmatrix}, \qquad d'_{k+1} = \begin{pmatrix} d'_k \\ f'_k \end{pmatrix}$$

with

$$u'_k = ((\gamma_{k+1}, \gamma_1), \ldots, (\gamma_{k+1}, \gamma_k))^\top;$$
$$v'_k = ((\gamma_1, \gamma_{k+1}), \ldots, (\gamma_k, \gamma_{k+1}));$$
$$a'_k = (\gamma_{k+1}, \gamma_{k+1});$$
$$d'_k = ((\gamma_0, \gamma_1), \ldots, (\gamma_0, \gamma_k))^\top;$$
$$f'_k = (\gamma_0, \gamma_{k+1}).$$

Setting $z_k{}' = (b_1{}', \ldots, b_k{}')^\mathsf{T}$ we have

$$z'_{k+1} = \begin{pmatrix} z'_k \\ 0 \end{pmatrix} + \frac{f'_k - v'_k z'_k}{\beta'_k} \begin{pmatrix} -A_k'^{-1} u'_k \\ 1 \end{pmatrix}$$

with $\beta_k{}' = a_k{}' - v_k{}' A_k'^{-1} u_k{}'$.

Of course the bordering method can only be used if β_k (or $\beta_k{}'$ in the second case) is different from zero. If it is not the case, instead of adding one new row and one new column to the system it is possible to add several rows and columns until a non singular β_k (which is now a square matrix) has been found (see [3] and [4]).

Location of the Zeros

We return to the normalization $b_k = 1$. As

$$c(x^i R_k(x)) = b_0 c_i + \cdots + b_k c_{i+k}$$

and $\partial c(x^i R_k(x))/\partial b_j = c_{i+j}$, from (1) we obtain

$$\sum_{i=0}^{m} c(x^i R_k(x)) c_{i+j} = 0 \quad \text{for } j = 0, \ldots, k-1. \quad (4)$$

This relation can be written as

$$c(R_k(x)(c_i + c_{i+1}x + \cdots + c_{i+m}x^m)) = 0,$$

for $i = 0, \ldots, k-1$. Let us now assume that

$$c_i = \int_a^b x^i d\alpha(x), \quad i = 0, 1, \ldots,$$

with α bounded and nondecreasing in $[a, b]$. We have

$$\begin{aligned} & c_i + c_{i+1}x + \cdots + c_{i+m}x^m \\ &= \sum_{j=0}^{m} \left[\int_a^b y^{i+j}\, d\alpha(y) \right] x^j \\ &= \int_a^b y^i \left(\sum_{j=0}^{m} x^j y^j \right) d\alpha(y). \end{aligned}$$

Set

$$w(x, \mu) = \int_a^b y^\mu \left(\sum_{j=0}^{m} x^j y^j \right) d\alpha(y).$$

Thus

$$w(x, i) = c_i + c_{i+1}x + \cdots + c_{i+m}x^m$$

and it follows that

$$c(R_k(x)w(x, i)) = \int_a^b R_k(x)w(x, i)\, d\alpha(x) = 0$$

for $i = 0, \ldots, k-1$, which shows that the polynomial R_k is *biorthogonal* in the sense of [7,8]. Let us now study the location of the zeros of R_k. For that purpose we shall apply [7, Thm. 3], also given as [8, Thm. 5]. Set

$$d\Phi(x, \mu) = w(x, \mu) d\alpha(x)$$

and

$$I_k(\mu) = \int_a^b x^k\, d\Phi(x, \mu), \quad k = 0, 1, \ldots.$$

In our case, μ takes the values $\mu_i = i - 1$, $i = 1, 2, \ldots$. Thus

$$\det\left[I_i(\mu_j)\right] = \det\left[(\gamma_{j-1}, \gamma_i)\right]$$

and the condition of regularity of [7,8] is equivalent to our condition for the existence and uniqueness of R_k. According to [7,8], we now have to look at the interpolation property of w. We have

$$w(x_i, \mu_j) = (\gamma_{j-1}, X_i)$$

where $X_i = (1, x_i, \ldots, x_i^m)^\top$, the x_i's being arbitrary distinct points in $[a, b]$, and thus

$$\begin{vmatrix} (\gamma_0, X_1) & \cdots & (\gamma_{k-1}, X_1) \\ \cdots & \cdots & \cdots \\ (\gamma_0, X_k) & \cdots & (\gamma_{k-1}, X_k) \end{vmatrix} = \det(\mathcal{X}_k \Gamma_k)$$

with

$$\mathcal{X}_k = \begin{pmatrix} X_1^\top \\ \vdots \\ X_k^\top \end{pmatrix} \quad \text{and} \quad \Gamma_k = (\gamma_0, \ldots, \gamma_{k-1}).$$

The interpolation property holds if and only if $\det(\mathcal{X}_k \Gamma_k) \neq 0$, that is, if and only if the matrix $\mathcal{X}_k \Gamma_k$ has rank k. Thus, using the theorem of [7,8], we have proved the following result:

Theorem 1 *If A_k is regular and if $\mathcal{X}_k \Gamma_k$ has rank k, then R_k exists and has k distinct zeros in $[a, b]$.*

Remark 2 When $0 \leq a < b$, it can be proved that $\det(\mathcal{X}_k \Gamma_k) \neq 0$ (see [2] for the details).

Applications

Our first application deals with *Padé-type approximation*. Let v_k be an arbitrary polynomial of degree k and let $w_k(t) = a_0 + \cdots + a_{k-1}t^{k-1}$ be defined by

$$a_i = c(x^{-i-1} v_k(x)), \quad i = 0, \ldots, k-1.$$

We set

$$\widetilde{v}_k(t) = t^k v_k(t^{-1}) \quad \text{and} \quad \widetilde{w}_k(t) = t^{k-1} w_k(t^{-1}).$$

Let f be the formal power series

$$f(t) = \sum_{i=0}^{\infty} c_i t^i.$$

Then it can be proved that

$$f(t) - \frac{\widetilde{w}_k(t)}{\widetilde{v}_k(t)} = O(t^k) \quad (t \to 0).$$

The rational function $\frac{\widetilde{w}_k(t)}{\widetilde{v}_k(t)}$ is called a Padé-type approximant of f and it is denoted by $(k-1/k)_f(t)$, [1]. Moreover it can also be proved that

$$f(t) - \frac{\widetilde{w}_k(t)}{\widetilde{v}_k(t)} = \frac{t^k}{\widetilde{v}_k(t)} c\left(\frac{v_k(x)}{1-xt}\right) = \frac{t^k}{\widetilde{v}_k(t)} \cdot c\left(\left(1 + xt + \cdots + x^{k-1}t^{k-1} + \frac{x^k t^k}{1-xt}\right) v_k(x)\right).$$

That is,

$$f(t)\widetilde{v}_k(t) - \widetilde{w}_k(t) = t^k \sum_{i=0}^{\infty} c(x^i v_k(x)) t^i.$$

Thus if the polynomial v_k, which is called the *generating polynomial* of $(k-1/k)$, satisfies

$$c(x^i v_k(x)) = 0 \quad \text{for } i = 0, \ldots, k-1,$$

then

$$f(t) - \frac{\widetilde{w}_k(t)}{\widetilde{v}_k(t)} = O(t^{2k}).$$

In this case v_k is the formal orthogonal polynomial P_k of degree k with respect to c and $\frac{\widetilde{w}_k(t)}{\widetilde{v}_k(t)}$ is the usual Padé approximant $[k-1/k]$ of f.

As explained in [10], Padé approximants can be quite sensitive to perturbations on the coefficients c_i of the series f. Hence the idea arises to take as v_k the least squares orthogonal polynomial R_k of degree k instead of the usual orthogonal polynomial, an idea which in fact motivated our study. Of course such a choice decreases the degree of approximation, since the approximants obtained are only of the Padé-type, but it can increase the stability properties of the approximants and also their precision since $\sum_{i=0}^{m} [c(x^i v_k(x))]^2$ is minimized by the choice $v_k = R_k$. We give a numerical example that illustrates this fact.

We consider the function

$$f(z) = \frac{\ln(1+z)}{z} = \sum_{i=0}^{\infty} c_i z^i$$

and we assume that we know the coefficients c_i with a certain precision. For example, we know approximate values c_i^* such that

$$\left|c_i - c_i^*\right| \le 10^{-8}, \quad i = 0, 1, \ldots.$$

In the following table we compare the number of exact figures given by the Padé approximant with those of the least squares Padé-type approximant, both computed with the same number of coefficients c_i^*. We can see that the least squares Padé-type approximant has better stability properties.

z	Padé approx [7/8]	LS Padé-type approx [6/7] ($m = 8$)
1.5	6.7	7.7
1.9	5.7	7.0
2.1	5.2	6.7

Another application concerns quadrature methods. We have already shown that if the functional c is given by

$$c_i = \int_a^b x^i \, d\alpha(x), \quad i = 0, 1, \ldots, \quad 0 \le a < b,$$

with α bounded and nondecreasing, then the corresponding least squares orthogonal polynomial of degree k, R_k, has k distinct zeros in $[a, b]$. We can then construct quadrature formulas of the interpolatory type.

If $\lambda_1, \ldots, \lambda_k$ are the zeros of R_k, we can approximate the integral

$$I = \int_a^b f(x) \, d\alpha(x)$$

by

$$I_k = A_1 f(\lambda_1) + \cdots + A_k f(\lambda_k) \tag{5}$$

where

$$A_i = \int_a^b \frac{\pi(x)}{\pi'(\lambda_i)(x - \lambda_i)} \, d\alpha(x)$$

and

$$\pi(x) = \prod_{j=1}^{k} (x - \lambda_j).$$

This corresponds to replacing the function f by its interpolating polynomial at the knots $\lambda_1, \ldots, \lambda_k$. The truncation error of (5) is given by

$$I - I_k = E_T = \int_a^b f[\lambda_1, \cdots, \lambda_k, x] R_k(x) \, d\alpha(x).$$

Expanding the divided difference we see

$$f[\lambda_1,\ldots,\lambda_k,x] = \sum_{i=1}^{k} f[\lambda_1,\cdots,\lambda_{k+i}](x-\lambda_{k+1})\cdots(x-\lambda_{k+i-1}) + f[\lambda_1,\ldots,\lambda_{k+m+1},x](x-\lambda_{k+1})\cdots(x-\lambda_{k+m+1})$$

for $\lambda_{k+1},\ldots,\lambda_{k+m+1}$ any points in the domain of definition $\mathcal{D}_f$ of f. If $0\in\mathcal{D}_f$, then we can choose

$$\lambda_{k+1}=\cdots=\lambda_{k+m+1}=0.$$

Setting

$$M_i = f[\lambda_1,\ldots,\lambda_{k+i}]$$

we get

$$f[\lambda_1,\ldots,\lambda_k,x] = \sum_{i=1}^{m+1} M_i x^{i-1} + x^{m+1} f[\lambda_1,\ldots,\lambda_{k+m+1},x]$$

and hence, for the truncation error

$$E_T = \sum_{i=0}^{m} M_{i+1}\left(\int_a^b R_k(x)x^i\,d\alpha(x)\right) + \int_a^b f[\lambda_1,\ldots,\lambda_{k+m+1},x]x^{m+1}R_k(x)\,d\alpha(x)$$

with

$$\sum_{i=0}^{m}\left(\int_a^b R_k(x)x^i\,d\alpha(x)\right)^2$$

minimised. Moreover, if $f\in C^{k+m+1}([a,b])$ and, since x^{m+1} is positive over $[a,b]$, we obtain

$$\int_a^b f[\lambda_1,\ldots,\lambda_{k+m+1},x]x^{m+1}R_k(x)\,d\alpha(x) = \frac{c_{m+1}}{(k+m+1)!}R_k(\lambda)f^{(k+m+1)}(\eta)$$

with $\lambda,\eta\in[a,b]$, and, for the error,

$$E_T = \sum_{i=0}^{m}\frac{f^{(k+i)}(\eta_i)}{(k+i)!}\left(\int_a^b R_k(x)x^i d\alpha(x)\right) + \frac{c_{m+1}}{(k+m+1)!}R_k(\lambda)f^{(k+m+1)}(\eta) \quad (6)$$

with $\eta_i\in[a,b]$, $i=0,\ldots,m$, $\lambda,\eta\in[a,b]$. We remark that in the case where $m=k-1$, R_k is the orthogonal polynomial with respect to the functional c and so (5) corresponds to a Gaussian quadrature formula. An advantage of the quadrature formulas (5) is that they are less sensitive to perturbations on the sequence of moments c_i, as is shown in the following numerical example. Such a case can arise in some applications where the formula giving the moments c_i is sensitive to rounding errors, see [11] for example.

Consider the functional c defined by

$$c_i = \int_0^1 x^i\,dx = \frac{1}{i+1}$$

and perturb the coefficients in the following way

i	c_i^*	i	c_i^*
0	1.00000011	6	0.14285700
1	0.50000029	7	0.12500000
2	0.33333340	8	0.11111109
3	0.25000101	9	0.10000000
4	0.20000070	10	0.09090899
5	0.16666600	11	0.08333300

We can construct from these coefficients the least squares orthogonal polynomials and the corresponding quadrature formulas (5). The precision of the numerical approximations of $I=\int_0^1 f(x)\,dx$ is given in the following table

$f(x)$	$k=5;m=4$ Gauss quad.	$k=5;m=6$ least sq. quad.
$1/(x+0.5)$	$-2.2*10^{-5}$	$-6.2*10^{-6}$
$1/(x+0.3)$	$-2.1*10^{-4}$	$-1.2*10^{-5}$

We can obtain other applications from the following generalization. Instead of minimizing $\sum_{i=0}^{m}[c(x^iR_k(x))]^2$ we can introduce weights and minimize

$$\Phi^*(b_0,\ldots,b_{k-1}) = \sum_{i=0}^{m} p_i\left[c(x^iR_k^*(x))\right]^2$$

with $p_i>0$, $i=0,\ldots,m$. If we choose the inner product

$$(\gamma_i,\gamma_j)^* = \sum_{k=0}^{m} p_k c_{i+k}c_{j+k}$$

the solution of this problem can be computed as in the previous case and all the properties of the polynomials are still true. It can be seen, from numerical examples, that if the sequence of moments c_i has a decreasing precision, we can expect that the least squares Padé-type approximants constructed with a decreasing sequence of weights will give a better result. In the same way, for the quadrature formulas (5), from the expression (6) of the truncation error and the knowledge of the magnitude of the derivatives, we can reduce this error by choosing appropriate weights. Some other possible applications of least squares orthogonal polynomials will be studied in the future.

See also

- ABS Algorithms for Linear Equations and Linear Least Squares
- ABS Algorithms for Optimization
- Gauss–Newton Method: Least Squares, Relation to Newton's Method
- Generalized Total Least Squares
- Least Squares Problems
- Nonlinear Least Squares: Newton-type Methods
- Nonlinear Least Squares Problems
- Nonlinear Least Squares: Trust Region Methods

References

1. Brezinski C (1980) Padé type approximation and general orthogonal polynomials. ISNM, vol 50. Birkhäuser, Basel
2. Brezinski C, Matos AC (1993) Least squares orthogonal polynomials. J Comput Appl Math 46:229–239
3. Brezinski C, Redivo Zaglia M (1991) Extrapolation methods. Theory and practice. North-Holland, Amsterdam
4. Brezinski C, Redivo Zaglia M, Sadok H (1992) A breakdown-free Lanczos type algorithm for solving linear systems. Numer Math 63:29–38
5. Faddeeva VN (1959) Computational methods of linear algebra. Dover, Mineola, NY
6. Gantmacher FR (1959) The theory of matrices. Chelsea, New York
7. Iserles A, Nørsett SP (1985) Bi-orthogonal polynomials. In: Brezinski C, Draux A, Magnus AP, Maroni P, Ronveaux A (eds) Orthogonal Polynomials and Their Applications. Lecture Notes Math. Springer, Berlin, pp 92–100
8. Iserles A, Nørsett SP (1988) On the theory of biorthogonal polynomials. Trans Amer Math Soc 306:455–474
9. Karlin S (1968) Total positivity. Stanford Univ. Press, Palo Alto, CA
10. Mason JC (1981) Some applications and drawbacks of Padé approximants. In: Ziegler Z (ed) Approximation Theory and Appl. Acad. Press, New York, pp 207–223
11. Morandi Cecchi M, Redivo Zaglia M (1991) A new recursive algorithm for a Gaussian quadrature formula via orthogonal polynomials. In: Brezinski C, Gori L, Ronveaux A (eds) Orthogonal Polynomials and Their Applications. Baltzer, Basel, pp 353–358

Least Squares Problems

Åke Björck
Linköping University, Linköping, Sweden

MSC2000: 65Fxx

Article Outline

Keywords

Least squares

Synonyms

LSP

Introduction

Historical Remarks

The linear least squares problem originally arose from the need to fit a linear mathematical model to given observations. In order to reduce the influence of errors in the observations one uses a greater number of measurements than the number of unknown parameters in the model.

The algebraic procedure of the method of least squares was first published by A.M. Legendre [25]. It was justified as a statistical procedure by C.F. Gauss [13]. A famous example of the use of the least squares principle is the prediction of the orbit of the asteroid Ceres by Gauss in 1801. After this success, the method of least squares quickly became the standard procedure for analysis of astronomical and geodetic data.

Gauss gave the method a sound theoretical basis in two memoirs: 'Theoria Combinationis' [11,12]. In them, Gauss proves the optimality of the least squares estimate without any assumptions that the random variables follow a particular distribution.

Statistical Models

In the *general univariate linear model* the vector $b \in \mathbf{R}^m$ of observations is related to the unknown parameter vector $x \in \mathbf{R}^n$ by a linear relation

$$Ax = b + \epsilon, \tag{1}$$

where $A \in \mathbf{R}^{m\times n}$ is a known matrix of full column rank. Further, ϵ is a vector of random errors with zero means and covariance matrix $\sigma^2\ W \in \mathbf{R}^{m\times m}$, where W is known but $\sigma^2 > 0$ unknown. The standard linear model is obtained for $W = I$.

Theorem 1 (*Gauss–Markoff theorem*) *Consider the standard linear model (1) with $W = I$. The best linear unbiased estimator of any linear function $c^\mathsf{T}x$ is $c^\top\widehat{x}$, where $\widehat{x}$ is obtained by minimizing the sum of the squared residuals,*

$$\|r\|_2^2 = \sum_{i=1}^{m} r_i^2, \tag{2}$$

where $r = b - Ax$ and $\|\cdot\|_2$ denotes the Euclidean vector norm. Furthermore, $E(s^2) = \sigma^2$, where s^2 is the quadratic form

$$s^2 = \frac{1}{m-n}(b - A\widehat{x})^\top(b - A\widehat{x}). \tag{3}$$

The variance-covariance matrix of the least squares estimate $\widehat{x}$ is given by

$$V(\widehat{x}) = \sigma^2(A^\top A)^{-1}. \tag{4}$$

The residual vector $\widehat{r} = b - A\widehat{x}$ satisfies $A^\top\widehat{r} = 0$, and hence there are n linear relations among the m components of $\widehat{r}$. It can be shown that the residuals $\widehat{r}$, and therefore also the quadratic form s^2, are uncorrelated with $\widehat{x}$, i. e., $\mathrm{cov}(\widehat{r},\widehat{x}) = 0$, $\mathrm{cov}(s^2,\widehat{x}) = 0$.

If the errors in ϵ are uncorrelated but not of equal variance, then the covariance matrix W is diagonal. Then the least squares estimator is obtained by solving the *weighted least squares problem*

$$\min \|D(Ax - b)\|_2\,, \qquad D = W^{-\frac{1}{2}}. \tag{5}$$

For the general case with no restrictions on A and W, see [23].

The assumption that A is known made in the linear model is frequently unrealistic since sampling or modeling errors often also affect A. In the *errors-in-variables model* one instead assumes a linear relation

$$(A + E)x = b + r, \tag{6}$$

where (E, r) is an error matrix whose rows are independently and identically distributed with zero mean and the same variance. An estimate of the parameters x in the model (6) is obtained from the total least squares (TLS) problem.

Characterization of Least Squares Solutions

Let $\mathcal{S}$ be set of all solutions to a least squares problem,

$$S = \{x \in \mathbb{R}^n\colon\ \|Ax - b\|_2 = \min\}\,. \tag{7}$$

Then $x \in \mathcal{S}$ if and only if $A^\mathsf{T}(b - Ax) = 0$ holds. Equivalently, $x \in \mathcal{S}$ if and only if x satisfies the *normal equations*

$$A^\top Ax = A^\top b. \tag{8}$$

Since $A^\mathsf{T}\,b \in \mathcal{R}(A^\mathsf{T}) = \mathcal{R}(A^\mathsf{T}\,A)$ the normal equations are always consistent. It follows that $\mathcal{S}$ is a nonempty, convex subset of $\mathbf{R}^n$. Any least squares solution x uniquely decomposes the right-hand side b into two orthogonal components

$$b = Ax + r, \quad Ax \in \mathcal{R}(A) \perp r \in \mathcal{N}(A^\top),$$

where $\mathcal{R}(A)$ and $\mathcal{N}(A^\mathsf{T})$ denote the range of A and the nullspace of A^T, respectively.

When rank $A < n$ there are many least squares solutions x, although the residual $b - Ax$ is still uniquely determined. There is always a unique least squares solution in $\mathcal{S}$ of minimum length. The following result applies to both overdetermined and underdetermined linear systems.

Theorem 2 *Consider the linear least squares problem*

$$\min_{x\in S} \|x\|_2, \quad S = \{x \in \mathbb{R}^n : \ \|b - Ax\|_2 = \min\}, \tag{9}$$

where $A \in \boldsymbol{R}^{m\times n}$ and rank(A) = r ≤ min(m, n). This problem always has a unique solution, which is distinguished by the property that

$$x \perp \mathcal{N}(A).$$

Pseudo-inverse and Conditioning

Singular Value Decomposition and Pseudo-inverse

A matrix decomposition of great theoretical and practical importance for the treatment of least squares problems is the singular value decomposition (SVD) of A,

$$A = U\Sigma V^\top = \sum_{i=1}^{n} u_i \sigma_i v_i^\top. \tag{10}$$

Here σ_i are the singular values of A and u_i and v_i the corresponding left and right singular vectors.

Using this decomposition the solution to problem (9) can be written $x = A^\dagger b$, where

$$A^\dagger = V \begin{pmatrix} \Sigma_r^{-1} & 0 \\ 0 & 0 \end{pmatrix} U^\top \in \mathbb{R}^{n\times m}. \tag{11}$$

Here $A^\dagger$ is called the *pseudo-inverse* of A. It is the unique matrix which minimizes $\|AX - I\|_F$, where $\|\cdot\|_F$ denotes the Frobenius norm. Note that the pseudo-inverse $A^\dagger$ is not a continuous function of A, unless one allows only perturbations which do not change the rank of A.

The pseudo-inverse was first introduced by E.H. Moore in 1920. R. Penrose [30] later gave the following elegant algebraic characterization.

Theorem 3 (*Penrose's conditions*) *The pseudo-inverse $X = A^\dagger$ is uniquely determined by the four conditions:*

1) $AXA = A$;
2) $XAX = X$;
3) $(AX)^\mathsf{T} = AX$;
4) $(XA)^\mathsf{T} = XA$.

It can be directly verified that $A^\dagger$ given by (11) satisfies these four conditions.

The *total least squares problem* (TLS problem) involves finding a perturbation matrix (E, r) having minimal Frobenius norm, which lowers the rank of the matrix (A, b). Consider the singular value decomposition of the augmented matrix (A, b):

$$(A, b) = U\Sigma V^\top, \quad \Sigma = \operatorname{diag}(\sigma_1, \ldots, \sigma_{n+1}),$$

where $\sigma_1 \geq \cdots \geq \sigma_{n+1} \geq 0$. Then, in the generic case, $(x, -1)^\mathsf{T}$ is a right singular vector corresponding to σ_{n+1} and $\min \|(E, r)\|_F = \sigma_{n+1}$.

An excellent survey of theoretical and computational aspects of the total least squares problem is given in [22].

Conditioning of the Least Squares Problem

Consider a *perturbed least squares problem* where $\widetilde{A} = A + \delta A$, $\widetilde{b} = b + \delta b$, and let the perturbed solution be $\widetilde{x} = x + \delta x$. Then, assuming that rank(A) = rank$(A + \delta A) = n$ one has the first order bound

$$\|\delta x\|_2 \leq \frac{1}{\sigma_n}\left(\|\delta b_1\|_2 + \|\delta A\|_2 \|x\|_2\right) + \frac{1}{\sigma_n^2}\|\delta A\|_2 \|r\|_2.$$

The *condition number of a matrix* $A \in \mathbf{R}^{m\times n}$ $(A \neq 0)$ is defined as

$$\kappa(A) = \|A\|_2 \left\|A^\dagger\right\|_2 = \frac{\sigma_1}{\sigma_r}, \tag{12}$$

where $\sigma_1 \geq \cdots \geq \sigma_r > 0$, are the nonzero singular values of A. Hence, the *normwise relative condition number* of the least squares problem can be written as

$$\kappa_{LS}(A, b) = \kappa(A) + \kappa(A)^2 \frac{\|r\|_2}{\|A\|_2 \|x\|_2}. \tag{13}$$

For a *consistent problem* $(r = 0)$ the last term is zero. However, in general the condition number depends on the size of r and involves a term proportional to $\kappa(A)^2$.

A more refined perturbation analysis, which applies to both overdetermined and underdetermined systems, has been given in [34]. In order to prove any meaningful result it is necessary to assume that rank$(A + \delta A)$ = rank(A). If rank(A) = min(m, n), the condition $\eta \equiv \|A^\dagger\|_2 \|\delta A\|_2 < 1$ suffices to ensure that this is the case.

Numerical Methods of Solution

The Method of Normal Equations

The first step in the method of normal equations for the least squares problem is to form the cross-products

$$C = A^\top A \in \mathbb{R}^{n\times n}, \quad d = A^\top b \in \mathbb{R}^n. \tag{14}$$

Since the matrix C is symmetric, it is only necessary to compute and store its upper triangular part. When $m \gg n$ this step will result in a great reduction in the amount of data.

The computation of C and d can be performed either using an inner product form (operating on columns of A) or an outer product form (operating on rows of A). Row-wise accumulation of C and d is advantageous if the matrix A is sparse or held in secondary storage. Partitioning A by rows, one has

$$C = \sum_{i=1}^{m} a_{i.} a_{i.}^\top, \quad d = \sum_{i=1}^{m} b_i a_{i.}, \tag{15}$$

where $\widetilde{a}_i^\top$ denotes the ith row of A. This expresses C as a sum of matrices of rank.

Gauss solved the symmetric positive definite system of normal equation by elimination, preserving symmetry, and solving for x by back-substitution. A different sequencing of this algorithm is to compute the *Cholesky factorization*

$$C = R^\top R, \tag{16}$$

where R is upper triangular with positive diagonal elements, and then solve the two triangular systems

$$R^\top z = d, \quad Rx = z, \tag{17}$$

by forward- and back-substitution, respectively. The Cholesky factorization, named after the French officer A.L. Cholesky, who worked on geodetic survey problems in Africa, was published by C. Benoit [1]. (In statistical applications this method is often known as the *square-root method*, although the proper square root of A should satisfy $B^2 = A$.)

The method of normal equations is suitable for moderately ill-conditioned problems but is not a backward stable method. The accuracy can be improved by using fixed precision iterative refinement in solving the normal equations.

Set $x_0 = 0$, $r_0 = 0$, and for $s = 0, 1, \ldots$ until convergence do

$$\begin{aligned} r_s &= b - Ax_s, \\ R^\top (R\delta x_s) &= A^\top r_s, \\ x_{s+1} &= x_s + \delta x_s. \end{aligned}$$

(Here, x_1 corresponds to the unrefined solution of the normal equations.)

The method of normal equations can fail when applied to weighted least squares problems. To see this consider a problem with two different weights γ and 1,

$$\min_x \left\| \begin{pmatrix} \gamma A_1 \\ A_2 \end{pmatrix} x - \begin{pmatrix} \gamma b_1 \\ b_2 \end{pmatrix} \right\|_2, \tag{18}$$

for which the matrix of normal equations is $A^\intercal A = \gamma^2 A_1^\top A_1 + A_2^\top A_2$. When $\gamma \gg 1$ this problem is called *stiff*. In the limit $\gamma \to \infty$ the solution will satisfy the subsystem $A_1 x = b_1$ exactly. If $\gamma > u^{-1/2}$ (u is the unit roundoff), the information in the matrix A_2 may completely disappear when forming $A^\intercal A$. For possible ways around this difficulty, see [4, Chap. 4.4].

Least Squares by QR Factorization

The *QR factorization* and its extensions are used extensively in modern numerical methods for solving least squares problems. Let $A \in \mathbf{R}^{m\times n}$ with $\operatorname{rank}(A) = n$. Then there are an orthogonal matrix $Q \in \mathbf{R}^{m\times m}$ and an upper triangular $R \in \mathbf{R}^{n\times n}$ such that

$$A = Q \begin{pmatrix} R \\ 0 \end{pmatrix} \tag{19}$$

Since orthogonal transformations preserve the Euclidean length, it follows that

$$\|Ax - b\|_2 = \left\| Q^\top (Ax - b) \right\|_2 \tag{20}$$

for any orthogonal matrix $Q \in \mathbf{R}^{m\times m}$. Hence using the QR factorization (19) the solution to the least squares problem can be obtained from

$$Q^\top b = \begin{pmatrix} d_1 \\ d_2 \end{pmatrix}, \quad Rx = d_1. \tag{21}$$

An algorithm based on the QR decomposition by Householder transformations was first developed in a seminal paper by G.H. Golub [18]. Here, Q is compactly represented as a product of Householder ma-

trices $Q = P_1 \cdots P_n$, where $P_k = I - \beta_k u_k u_k^\top$. Only the Householder vectors u_k are stored, and advantage is taken of the fact that the first $k-1$ components of u_k are zero.

Golub's method for solving the standard least squares problem is normwise backward stable, see [24, pp. 90ff]. Surprisingly, this method is stable also for solving the weighted least squares problems (5) provided only that the equations are sorted after decreasing row norms in A, see [8].

Due to storage considerations the matrix Q in a QR decomposition is often discarded when A is large and sparse. This creates a problem, since then it may not be possible to form $Q^\top b$. If the original matrix A is saved one can use the *corrected seminormal equations* (CSNE)

$$R^\top R\overline{x} = A^\top b, \quad \overline{r} = b - A\overline{x},$$
$$\overline{R}^\top \overline{R}\delta x = A^\top \overline{r}, \quad x_c = \overline{x} + \delta x.$$

(Note that unless the correction step is carried out the numerical stability of this method is no better than the method of normal equations.) An error analysis of the CSNE method is given in [2]. A comparison with the bounds for a backward stable method shows that in most practical applications the corrected seminormal equations is forward stable.

Applying the *Gram–Schmidt orthogonalization* process to the columns of A produces Q_1 and R in the factorization

$$A = (a_1, \ldots, a_n) = Q_1 R, \quad Q_1 = (q_1, \ldots, q_n),$$

where Q_1 has orthogonal columns and R is upper triangular. There are two computational variants of Gram–Schmidt orthogonalization, the *classical Gram–Schmidt orthogonalization* (CGS) and the *modified Gram–Schmidt orthogonalization* (MGS). In CGS there may be a catastrophic loss of orthogonality unless reorthogonalization is used. In MGS the loss of orthogonality can be shown to occur in a predictable manner.

Using an equivalence between MGS and Householder QR applied to A with a square matrix of zeros on top, backward stable algorithm based on MGS for solving least squares problems have been developed, see [3].

Rank-Deficient and Ill-Conditioned Problems

The mathematical notion of rank is not always appropriate in numerical computations. For example, if a matrix $A \in \mathbf{R}^{n\times n}$, with (mathematical) rank $k < n$, is randomly perturbed by roundoff, the perturbed matrix most likely has full rank n. However, it should be considered to be 'numerically' rank deficient.

When solving rank-deficient or ill-conditioned least squares problems, correct assignment of the 'numerical rank' of A is often the key issue. The *numerical rank* should depend on a tolerance which reflects the error level. Overestimating the rank may lead to a computed solution of very large norm, which is totally irrelevant. This behavior is typical in problems arising from discretizations of ill-posed problems, see [21].

Assume that the 'noise level' δ in the data is known. Then a numerical rank k, such that $\sigma_k > \delta \geq \sigma_{k+1}$, can be assigned to A, where σ_i are the singular values of A. The approximate solution

$$x = \sum_{i=1}^{k} \frac{c_i}{\sigma_i} v_i, \quad c = U^\top b,$$

is known as the *truncated singular value decomposition solution* (TSVD). This solution solves the related least squares problem $\min_x \|A_k x - b\|_2$, where

$$A_k = \sum_{i=1}^{k} u_i \sigma_i v_i^\top, \quad \|A - A_k\|_2 \leq \delta,$$

is the best rank k approximation of A, The subspace

$$\mathcal{R}(V_2), \quad V_2 = (v_{k+1}, \ldots, v_n),$$

is called the *numerical nullspace* of A.

An alternative to TSVD is *Tikhonov regularization*, where one considers the regularized problem

$$\min_x \|Ax - b\|_2^2 + \tau^2 \|Dx\|_2^2, \tag{22}$$

for some positive diagonal matrix $D = \mathrm{diag}(d_1, \ldots, d_n)$. The problem (22) is equivalent to the least squares problem

$$\min_x \left\| \begin{pmatrix} \tau D \\ A \end{pmatrix} x \begin{pmatrix} \tau 0 \\ b \end{pmatrix} \right\|_2, \tag{23}$$

where the matrix A has been modified by appending the matrix $\tau\ D$ on top. An advantage of using the regularized problem (23) instead of the TSVD is that its solution can be computed from a QR decomposition. When $\tau > 0$ this problem is always of full column rank and has

a unique solution. For $D = I$ it can be shown that $x(\tau)$ will approximately equal the TSVD solution for $\tau = \delta$.

Problem (23) also appears as a subproblem in trust region algorithms for solving nonlinear least squares, and in interior point methods for constrained linear least squares problems. A more difficult case is when the noise level δ is unknown and has to be determined in the solution process. Such problems typically arise in the treatment of discrete ill-posed problems, see [21].

Rank Revealing QR Factorizations

In some applications it is too expensive to compute the SVD. In such cases so called 'rank revealing' QR factorizations, often are a good substitute.

It can be shown that for any $0 < k < n$ a column permutation Π exists such that the QR decomposition of $A\,\Pi$ has the form

$$A\Pi = Q \begin{pmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{pmatrix}, \tag{24}$$

where

$$\sigma_k(R_{11}) \geq \frac{1}{c}\sigma_k, \quad \|R_{22}\|_2 \leq c\sigma_{k+1}, \tag{25}$$

and $c < (n+1)/2$. In particular, if A has numerical δ-rank equal to k, then there is a column permutation such that $\| R_{22} \|_2 \leq c\,\delta$. Such a QR factorization is called a *rank revealing QR factorization* (RRQR). No efficient numerical method is known which can be guaranteed to compute an RRQR factorization satisfying (25), although in practice *Chan's method* [7] often gives satisfactory results.

A related rank revealing factorization is the complete orthogonal decomposition of the form

$$A = U \begin{pmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{pmatrix} V^{\mathsf{T}}, \tag{26}$$

where U and V are orthogonal matrices, $R_{11} \in \mathbf{R}^{k\times k}$, $\sigma_k(R_{11}) \geq \sigma_k/c$, and

$$\left(\|R_{12}\|_F^2 + \|R_{22}\|_F^2\right)^{\frac{1}{2}} \leq c\sigma_{k+1}.$$

This is also often called a *rank revealing URV factorization*. (an alternative lower triangular form ULV is sometimes preferable to use.) If $V = (V_1 V_2)$ is partitioned conformably the orthogonal matrix V_2 can be taken as an approximation to the numerical nullspace $\mathcal{N}(A)$.

Updating Least Squares Solutions

It is often desired to solve a sequence of modified least squares problems

$$\min_x \|Ax - b\|_2, \quad A \in \mathbb{R}^{m\times n}, \tag{27}$$

where in each step rows of data in $(A,\ b)$ are added, deleted, or both. This need arises, e. g., when data are arriving sequentially. In various time-series problems a window moving over the data is used; when a new observation is added, an old one is deleted as the window moves to the next step in the sample. In other applications columns of the matrix A may be added or deleted. Such modifications are usually referred to as updating (downdating) of least squares solutions.

Important applications where modified least squares problems arise include statistics, optimization, and signal processing. In statistics an efficient and stable procedure for adding and deleting rows to a regression model is needed; see [6]. In regression models one may also want to examine the different models, which can be achieved by adding or deleting columns (or permuting columns).

Recursive Least Squares

Applications in signal processing often require near real-time solutions. It is then critical that the modification should be performed with as few operations and as little storage requirement as possible.

Methods based on the normal equations and/or updating of the Cholesky factorization are still often used in statistics and signal processing, although these algorithms lack numerical stability. Consider a least squares problem where an observation $w^{\mathsf{T}}x = \beta$ is added. The updated solution $\widetilde{x}$ then satisfies the modified normal equations

$$(A^{\mathsf{T}}A + ww^{\mathsf{T}})\widetilde{x} = A^{\mathsf{T}}b + \beta w. \tag{28}$$

A straightforward method for computing $\widetilde{x}$ is based on updating the (scaled) covariance matrix $C = (A^{\mathsf{T}}\,A)^{-1}$. By the *Sherman–Morrison formula* one obtains $\widetilde{C}^{-1} = C^{-1} + ww^{\mathsf{T}}$, and

$$\widetilde{C} = C - \frac{1}{1 + w^{\mathsf{T}}u}uu^{\mathsf{T}}, \quad u = Cw. \tag{29}$$

From this follows the updating formula

$$\widetilde{x} = x + (\beta - w^{\mathsf{T}}x)\widetilde{u}, \quad \widetilde{u} = \widetilde{C}w. \tag{30}$$

The equations (29), (30) define a *recursive least squares* (RLS) algorithm. They can, with slight modifications, also be used for 'deleting' observations. The simplicity of this updating algorithm is appealing, but a disadvantage is its serious sensitivity to roundoff errors.

Modifying Matrix Factorizations

The first area where algorithms for *modifying matrix factorizations* seems to have been systematically used is optimization. Numerous aspects of updating various matrix factorizations are discussed in [17].

There is a simple relationship between the problem of updating matrix factorizations and that of updating least squares solutions. If A has full column rank and the R-factor of the matrix (A, b) is

$$\begin{pmatrix} R & z \\ 0 & \rho \end{pmatrix}, \tag{31}$$

then the solution to the least squares problem (27) is given by

$$Rx = z, \quad \|Ax - b\|_2 = \rho. \tag{32}$$

Hence updating algorithms for the QR or Cholesky factorization can be applied to (A, b) in order to give updating algorithms for least squares solutions.

Backward stable algorithms, which require $O(m^2)$ multiplications, exist for updating the QR decomposition for three important kinds of modifications:

- General rank one change of A.
- Deleting (adding) a column of A.
- Adding (deleting) a row of A.

In these algorithms, $Q \in \mathbf{R}^{m\times m}$ is stored explicitly as an $m \times m$ matrix. In many applications it suffices to update the 'Gram–Schmidt' QR decomposition

$$A = Q_1 R, \quad Q_1 \in \mathbb{R}^{m\times n}, \tag{33}$$

where $Q_1 \in \mathbf{R}^{m\times n}$ consists of the first n columns of Q, [10,31]. These only require $O(mn)$ storage and operations.

J.R. Bunch and C.P. Nielsen [5] have developed methods for updating the SVD

$$A = U \begin{pmatrix} \Sigma \\ 0 \end{pmatrix} V^{\top},$$

where $U \in \mathbf{R}^{m\times m}$ and $V \in \mathbf{R}^{n\times n}$, when A is modified by adding or deleting a row or column. However, their algorithms require $O(mn^2)$ flops.

Rank revealing QR factorizations can be updated more cheaply, and are often a good alternative to use. G.W. Stewart [33] has shown how to compute and update a rank revealing complete orthogonal decomposition from an RRQR decomposition.

Most updating algorithms can be modified in a straightforward fashion to treat cases where a block of rows/columns are added or deleted. which are more amenable to efficient implementation on vector and parallel computers.

Sparse Problems

The gain in operations and storage in solving the linear least squares problems where the matrix A is sparse can be huge, making otherwise intractable problems possible to solve. *Sparse least squares problems* of huge size arise in a variety of applications, such as geodetic surveys, photogrammetry, molecular structure, gravity field of the earth, tomography, the force method in structural analysis, surface fitting, and cluster analysis and pattern matching.

Sparse least squares problems may be solved either by direct or iterative methods. Preconditioned iterative methods can often be considered as hybrids between these two classes of solution. Below direct methods are reviewed for some classes of sparse problems.

Banded Least Squares Problems

A natural distinction is between sparse matrices with regular zero pattern (e. g., banded structure) and matrices with an irregular pattern of nonzero elements.

A rectangular *banded matrix* $A \in \mathbf{R}^{m\times n}$ has the property that the nonzero elements in each row lie in a narrow band. A is said to have *row bandwidth* w if

$$w(A) = \max_{1\le i\le m} (l_i(A) - f_i(A) + 1). \tag{34}$$

where

$$f_i(A) = \min\{j:\ a_{ij} \neq 0\},$$
$$l_i(A) = \max\{j:\ a_{ij} \neq 0\}$$

are column subscripts of the first and last nonzeros in the ith row of A. For this structure to have practical significance one needs to have $w \ll n$. Note that, although the row bandwidth is independent of the row ordering, it will depend on the column ordering. To permute the

columns in A so that a small bandwidth is achieved the method of choice is the reverse Cuthill–McKee ordering, see [15].

It is easy to see that if the row bandwidth of A is w then the matrix of normal equations $C = A^{\mathsf{T}}A$ has at most upper bandwidth $p = w - 1$, i. e.,

$$|j - k| \geq w \quad \Rightarrow \quad (A^{\top}A)_{jk} = \sum_{i=1}^{m} a_{ij}a_{ik} = 0.$$

If advantage is taken of the band structure, the solution of a least squares problem where A has bandwidth w by the method of normal equations requires a total of

$$\tfrac{1}{2}\,(mw(w+3) + n(w-1)(w+2)) + n(2w-1)$$

flops.

Similar savings can be obtained for methods based on Givens QR decomposition used to solve banded least squares problem. However, then it is essential that the rows of A are sorted so that the column indices $f_i(A)$, $i = 1, \ldots, m$, of the first nonzero element in each row form a nondecreasing sequence, i. e.,

$$i \leq k \quad \Rightarrow \quad f_i(A) \leq f_k(A).$$

A matrix whose rows are sorted in this way is said to be in *standard form*. Since the matrix R in the QR factorization has the same structure as the Cholesky factor, it must be a banded matrix with nonzero elements only in the first $p = w - 1$ superdiagonals. In the *sequential row orthogonalization scheme* an upper triangular matrix R is initialized to zero. The orthogonalization then proceeds row-wise, and R is updated by adding a row of A at a time.

If A has constant bandwidth and is in standard form then in the ith step of reduction the last $(n - l_i(A))$ columns of R have not been touched and are still zero as initialized. Further, the first $(f_i(A) - 1)$ rows of R are already finished at this stage and can be read out to secondary storage. Thus, as with the Cholesky method, very large problems can be handled since primary storage is needed only for the active part of R. The complete orthogonalization requires about $2mw^2$ flops, and can be performed in $w(w + 3)/2$ locations of primary storage.

The Givens rotations could also be applied to one or several right-hand sides b. Only if right-hand sides which are not initially available are to be treated, need the Givens rotations be saved. The algorithm can be modified to also handle problems with variable row bandwidth w_i.

For the case when $m \gg n$ a more efficient schemes uses Householder transformations, see [24, Chap. 11]. Let A_k consist of the rows of A for which the first nonzero element is in column k. Then, in step k of this algorithm, the A_k is merged with R_{k-1}, by computing the QR factorization

$$Q_k^{\top}\begin{pmatrix} R_{k-1} \\ A_k \end{pmatrix} = R_k.$$

Note that this and later steps will not involve the first $k - 1$ rows and columns of R_{k-1}. Hence the first $k - 1$ rows of R_{k-1} are rows in the final matrix R.

The reduction using this algorithm takes about $w(w + 1)(m + 3n/2)$ flops, which is approximately half as many as for the Givens method. As in the Givens algorithm the Householder transformations can also be applied to one or several right-hand sides b to produce $c = Q^{\mathsf{T}}b$. The least squares solution is then obtained from $Rx = c_1$ by back-substitution.

It is essential that the Householder transformations be subdivided as outlined above, otherwise intermediate fill will occur and the operation count increase greatly, see the example in [32].

Block Angular Form

There is often a substantial similarity in the structure of large sparse least squares problems. The matrices possess a block structure, perhaps at several levels, which reflects a 'local connection' structure in the underlying physical problem. In particular, the problem can often be put in the following bordered block diagonal or *block angular form*:

$$A = \begin{pmatrix} A_1 & & & B_1 \\ & \ddots & & \vdots \\ & & A_M & B_M \end{pmatrix}, \tag{35}$$

$$x = \begin{pmatrix} x_1 \\ \vdots \\ x_{M+1} \end{pmatrix}, \quad b = \begin{pmatrix} b_1 \\ \vdots \\ b_M \end{pmatrix}. \tag{36}$$

From (35) it follows that the variables $x_1, \ldots, x_M$ are coupled only to the variables x_{M+1}. Some applications

where the form (35) arises naturally are photogrammetry, Doppler radar positioning [27], and geodetic survey problems [20].

Problems of block angular form can be efficiently treated either by using normal equations of by QR factorization. It is easily seen that the matrix R from Cholesky or QR will have a block structure similar to that of A,

$$R = \begin{pmatrix} R_1 & & & S_1 \\ & \ddots & & \vdots \\ & & R_M & S_M \\ & & & R_{M+1} \end{pmatrix}, \tag{37}$$

where the $R_i \in \mathbf{R}^{n_i \times n_i}$ are upper triangular. This factor can be computed by first performing a sequence of orthogonal transformations yielding

$$Q_i^\top (A_i, B_i) = \begin{pmatrix} R_i & S_i \\ 0 & T_i \end{pmatrix}, \quad Q_i^\top b_i = \begin{pmatrix} c_i \\ d_i \end{pmatrix}.$$

Any sparse structure in the blocks A_i should be exploited. The last block row R_{M+1}, c_{M+1} is obtained by computing the QR decomposition

$$\widetilde{Q}_{M+1}^\top \begin{pmatrix} T & d \end{pmatrix} = \begin{pmatrix} R_{M+1} & c_{M+1} \\ 0 & d_{M+1} \end{pmatrix},$$

where

$$T = \begin{pmatrix} T_1 \\ \vdots \\ T_M \end{pmatrix}, \quad d = \begin{pmatrix} d_1 \\ \vdots \\ d_M \end{pmatrix}.$$

The unknown x_{M+1} is determined from the triangular system $R_{M+1}\, x_{M+1} = c_{M+1}$. Finally $x_M, \ldots, x_1$ are computed by back-substitution in the sequence of triangular systems $R_i x_i = c_i - S_i\, x_{M+1}$, $i = M, \ldots, 1$. Note that a large part of the computations can be performed in parallel on the M subsystems.

Several modifications of this basic algorithm have been suggested in [19] and [9].

General Sparse Problems

If A is partitioned by rows, then (15) can be used to compute the matrix $C = A^\mathsf{T} A$. Make the 'no-cancellation assumption' that whenever two nonzero numerical quantities are added or subtracted, the result is nonzero. Then it follows that the nonzero structure of $A^\mathsf{T} A$ is the direct sum of the nonzero structures of $a_i . a_i^\top$, $i = 1, \ldots, m$, where $a_i^\top$ denotes the ith row of A. Hence the undirected graph $G(A^\mathsf{T} A)$ representing the structure of $A^\mathsf{T} A$ can be constructed as the direct sum of all the graphs $G(a_i . a_i^\top)$, $i = 1, \ldots, m$. The nonzeros in row $a_i^\top$ will generate a subgraph, where all pairs of nodes are connected. Such a subgraph is called a *clique*.

From the graph $G(A^\mathsf{T} A)$ the structure of the Cholesky factor R can be predicted by using a graph model of Gaussian elimination. The fill under the factorization process can be analyzed by considering a sequence of undirected graphs $G_i = G(A^{(i)})$, $i = 0, \ldots, n-1$, where $A^{(0)} = A$. These *elimination graphs* can be recursively formed in the following way. Form G_i from $G_{(i-1)}$ by removing the node i and its incident edges and adding fill edges. The fill edges in eliminating node v in the graph G are

$$\{(j,k) \colon\ (j,k) \in \mathrm{Adj}_G(v),\ j \neq k\}.$$

Thus, the fill edges correspond to the set of edges required to make the adjacent nodes of v pairwise adjacent. The filled graph $G_F(A)$ of A is a graph with n vertices and edges corresponding to all the elimination graphs G_i, $i = 0, \ldots, n-1$. The filled graph bounds the structure of the Cholesky factor R,

$$G(R^\top + R) \subset G_F(A). \tag{38}$$

This also give an upper bound for the structure of the factor R in the QR decomposition.

A reordering of the columns of AP of A corresponds to a symmetric reordering of the rows and columns of $A^\mathsf{T} A$. Although this will not affect the number of nonzeros in $A^\mathsf{T} A$, only their positions, it may greatly affect the number of nonzeros in the Cholesky factor R. Before carrying out the Cholesky or QR factorization numerically, it is therefore important to find a permutation matrix P such that $P^\mathsf{T} A^\mathsf{T} AP$ has a sparse Cholesky factor R.

By far the most important local ordering method is the *minimum degree ordering* In terms of the Cholesky factorization this ordering is equivalent to choosing the ith pivot column as one with the minimum number of nonzero elements in the unreduced part of $A^\mathsf{T} A$. This will minimize the number of entries that will be modified in the next elimination step. Remarkably fast

symbolic implementations of the minimum degree algorithm exist, which use refinements of the elimination graph model of the Cholesky factorization. See [16] for a survey of the extensive development of efficient versions of the minimum degree algorithm.

Another important ordering method is *substructuring* or *nested dissection*, which results in a nested block angular form. Here the idea is to choose a set of nodes $\mathcal{B}$ in the graph $G(A^\intercal A)$, which separates the other nodes into two sets $\mathcal{A}_1$ and $\mathcal{A}_2$ so that node variables in $\mathcal{A}_1$ are not connected to node variables in $\mathcal{A}_2$. The variables are then ordered so that those in $\mathcal{A}_1$ appear first, those in $\mathcal{A}_2$ second, and those in $\mathcal{B}$ last. Finally the equations are ordered so that those including $\mathcal{A}_1$ come first, those including $\mathcal{A}_2$ next, and those only involving variables in $\mathcal{B}$ come last. This dissection can be continued recursively, first dissecting the regions $\mathcal{A}_1$ and $\mathcal{A}_2$ each into two subregions, and so on.

An algorithm using the normal equations for solving sparse linear least squares problems is usually split in a symbolical and a numerical phase as follows.

1) Determine symbolically a column permutation P_c such that $P_c^\top A^\intercal A P_c$ has a sparse Cholesky factor R.
2) Perform the Cholesky factorization of $P_c^\top A^\intercal A P_c$ symbolically to generate a storage structure for R.
3) Compute $B = P_c^\top A^\intercal A P_c$ and $c = P_c^\top A^\intercal b$ numerically, storing B in the data structure of R.
4) Compute the Cholesky factor R numerically and solve $R^\intercal z = c$, $Ry = z$, giving the solution $x = P_c y$.

Here, steps 1 and 2 involve only symbolic computation and apply also to a sparse QR algorithm. For details of the implementation of the numerical factorization see [15, Chap. 5]. For moderately ill-conditioned problems a sparse Cholesky factorization, possibly used with iterative refinement, is a satisfactory choice.

Orthogonalization methods are potentially more accurate since they work directly with A. The number of operations needed to compute the QR decomposition depends on the row ordering, and the following heuristic row ordering algorithm should be applied to A before the numerical factorization takes place:

First sort the rows after increasing $f_i(A)$, so that $f_i(A) \le f_k(A)$ if $i < k$. Then for each group of rows with $f_i(A) = k$, $k = 1, \ldots, \max_i f_i(A)$, sort all the rows after increasing $L_i(A)$.

In the sparse case, applying the usual sequence of Householder reflections may cause a lot of *intermediate fill-in*, with consequent cost in operations and storage. In the row sequential algorithm by J.A. George and M.T. Heath [14], this problem is avoided by using a row-oriented method employing Givens rotations. Even more efficient are *multifrontal methods*, in which Householder transformations are applied to a sequence of small dense subproblems.

Note that in most sparse QR algorithms the orthogonal factor Q is not stored. The corrected seminormal equations are used for treating additional right-hand sides. The reason is that for rectangular matrices A the matrix Q is usually much less sparse than R. In the multifrontal algorithm, however, Q can efficiently be represented by the Householder vectors of the frontal orthogonal transformations, see [26].

A Fortran multifrontal sparse QR subroutine, called QR27, has been developed by P. Matstoms [28]. He [29] has also developed a version of this to be used with MATLAB, implemented as four M-files and available from netlib.

See also

- ABS Algorithms for Linear Equations and Linear Least Squares
- ABS Algorithms for Optimization
- Gauss, Carl Friedrich
- Gauss–Newton Method: Least Squares, Relation to Newton's Method
- Generalized Total Least Squares
- Least Squares Orthogonal Polynomials
- Nonlinear Least Squares: Newton-type Methods
- Nonlinear Least Squares Problems
- Nonlinear Least Squares: Trust Region Methods

References

1. Benoit C (1924) Sur la méthode de résolution des, équationes normales, etc. (Procédés du commandant Cholesky). Bull Géodésique 2:67–77
2. Björck Å (1987) Stability analysis of the method of seminormal equations for least squares problems. Linear Alg & Its Appl 88/89:31–48
3. Björck Å (1994) Numerics of Gram–Schmidt orthogonalization. Linear Alg & Its Appl 197-198:297–316
4. Björck Å (1996) Numerical methods for least squares problems. SIAM, Philadelphia
5. Bunch JR, Nielsen CP (1978) Updating the singular value decomposition. Numer Math 31:111–129

6. Chambers JM (1971) Regression updating. J Amer Statist Assoc 66:744–748
7. Chan TF (1987) Rank revealing {QR}-factorizations. LAA 88/89:67–82
8. Cox AJ, Higham NJ (1997) Stability of Householder QR factorization for weighted least squares problems. Numer Anal Report Manchester Centre Comput Math, Manchester, England, 301
9. Cox MG (1990) The least-squares solution of linear equations with block-angular observation matrix. In: Cox MG, Hammarling SJ (eds) Reliable Numerical Computation. Oxford Univ. Press, Oxford, pp 227–240
10. Daniel J, Gragg WB, Kaufman L, Stewart GW (1976) Reorthogonalization and stable algorithms for updating the Gram–Schmidt QR factorization. Math Comput 30:772–95
11. Gauss CF (1880) Theoria combinationis observationum erroribus minimis obnoxiae, pars posterior. In: Werke, IV. Königl. Gesellschaft Wissenschaft, Göttingen, pp 27–53, First published in 1823.
12. Gauss CF (1880) Theoria combinationis observationum erroribus minimis obnoxiae, pars prior. In: Werke, IV. Königl. Gesellschaft Wissenschaft. Göttingen, Göttingen, pp 1–26, First published in 1821.
13. Gauss CF (1963) Theory of the motion of the heavenly bodies moving about the Sun in conic sections. Dover, Mineola, NY (Translation by Davis CH); first published in 1809
14. George JA, Heath MT (1980) Solution of sparse linear least squares problems using Givens, rotations. Linear Alg & Its Appl 34:69–83
15. George JA, Liu JW-H (1981) Computer solution of large sparse positive definite systems. Prentice-Hall, Englewood Cliffs, NJ
16. George JA, Liu JW-H (1989) The evolution of the minimum degree ordering algorithm. SIAM Rev 31:1–19
17. Gill PE, Golub GH, Murray W, Saunders MA (1974) Methods for modifying matrix factorizations. Math Comput 28:505–535
18. Golub GH (1965) Numerical methods for solving least squares problems. Numer Math 7:206–216
19. Golub GH, Manneback P, Toint P (1986) A comparison between some direct and iterative methods for large scale geodetic least squares problems. SIAM J Sci Statist Comput 7:799–816
20. Golub GH, Plemmons RJ (1980) Large-scale geodetic least-squares adjustment by dissection and orthogonal decomposition. Linear Alg & Its Appl 34:3–28
21. Hansen PC (1998) Rank-deficient and discrete ill-posed problems. Numerical aspects of linear inversion. SIAM, Philadelphia
22. Van Huffel S, Vandewalle J (1991) The total least squares problem: Computational aspects and analysis. Frontiers in Appl Math, vol 9. SIAM, Philadelphia
23. Kourouklis S, Paige CC (1981) A constrained approach to the general Gauss–Markov, linear model. J Amer Statist Assoc 76:620–625
24. Lawson CL, Hanson RJ (1974) Solving least squares problems. Prentice-Hall, Englewood Cliffs, NJ
25. Legendre AM (1805) Nouvelles méthodes pour la détermination des orbites des comètes. Courcier, Paris
26. Lu S-M, Barlow JL (1996) Multifrontal computation with the orthogonal factors of sparse matrices. SIAM J Matrix Anal Appl 17:658–679
27. Manneback P, Murigande C, Toint PL (1985) A modification of an algorithm by Golub and Plemmons for large linear least squares in the context of Doppler positioning. IMA J Numer Anal 5:221–234
28. Matstoms P (1992) QR27-specification sheet. Techn. Report Dept. Math. Linköping Univ.
29. Matstoms P (1994) Sparse QR factorization in MATLAB. ACM Trans Math Softw 20:136–159
30. Penrose R (1955) A generalized inverse for matrices. Proc Cambridge Philos Soc 51:406–413
31. Reichel L, Gragg WB (1990) FORTRAN subroutines for updating the QR decomposition. ACM Trans Math Softw 16:369–377
32. Reid JK (1967) A note on the least squares solution of a band system of linear equations by Householder reductions. Computer J 10:188–189
33. Stewart GW (1992) An updating algorithm for subspace tracking. IEEE Trans Signal Processing 40:1535–1541
34. Wedin P-Å (1973) Perturbation theory for pseudo-inverses. BIT 13:217–232

Leibniz, Gottfried Wilhelm

Sandra Duni Eksioglu
Industrial and Systems Engineering Department, University Florida, Gainesville, USA

MSC2000: 01A99

Article Outline

Keywords

Gottfried Wilhelm Leibniz; Integration; Differentiation; Theory of envelops; Infinitesimal calculus

G.W. Leibniz (1646–1716) was a well-known German philosopher and mathematician. He is considered a de-

scendant of German idealism and a pioneer of the Enlightenment. Leibniz is known as the inventor of the differential and integral calculus [7].

Leibniz's contribution in philosophy is as significant as in mathematics. In philosophy Leibniz is known for his fundamental philosophical ideas and principles including truth, necessary and contingent truths, possible worlds, the principle of sufficient reason (i. e., there is a reason behind everybody's action), the principle of pre-established harmony (i. e., the universe is created in such a way that corresponding mental and physical events occur simultaneously), and the principle of noncontradiction (i. e., if a contradiction can be derived from a proposition, this proposition is false). Leibniz was fond on the idea that the principles of reasoning could be organized into a formal symbolic system, an algebra or calculus of thought, where disagreements could be settled by calculations [4].

Leibniz was the son of a professor of moral philosophy at Leipzig Univ. Leibniz learned to read from his father before going to school. He taught himself Latin and Greek by age 12, so that he could read the books in his father's library. He studied law at the Univ. of Leipzig from 1661 to 1666. In 1666 he was refused the degree of doctor of laws at Leipzig. He went to the Univ. of Altdorf, which awarded him doctorate in jurisprudence in 1667 [1].

Leibniz started his career at the courts of Mainz where he worked until 1672. The Elector of Mainz promoted him to diplomatic services. In 1672 he visited Paris to try to dissuade Louis XIV from attacking German areas. Leibniz remained in Paris until 1676, where he continued to practice law. In Paris he studied mathematics and physics under Chr. Huygens. During this period he developed the basic features of his version of the calculus. He spent the rest of his life, from 1676 until his death (November 14, 1716) at Hannover [6].

Leibniz's most important achievement in mathematics was the discovery of *infinitesimal calculus*. The significance of calculus is so important that it was marked as the starting point of modern mathematics. Leibniz's formulations were different from previous investigation by I. Newton. Newton was mainly concentrated in the geometrical representation of calculus, while Leibniz took it towards analysis. Newton considered variables changing with time. Leibniz thought of variables x, y as ranging over sequences of infinitely close values. For Newton integration and differentiation were inverses, while Leibniz used *integration* as a summation. At that time, neither Leibniz nor Newton thought in terms of functions, both always thought in terms of graphs.

In November 1675 he wrote a manuscript using the notation $\int f(x)\,dx$ for the first time [5]. In the same manuscript he presented the product rule for *differentiation*. The quotient rule first appeared two years later, in July 1677. In 1676 Leibniz arrived in the conclusion that he was in possession of a method that was highly important because of its generality. Whether a function was rational or irrational, algebraic or transcendental (a word that Leibniz coined), his operations of finding sums and differences could always be applied.

In November 1676 Leibniz discovered the familiar notation $d(x^n) = nx^{n-1}\,dx$ for both integral and fractional n. Newton claimed that: 'not a single previously unsolved problem was solved here', but the formalism of Leibniz's approach proved to be vital in the development of the calculus. Leibniz never thought of the derivative as a limit. This does not appear until the work of J. d'Alembert. Leibniz was convinced that good mathematical notations were the key to progress so he experimented with different notation for coefficient systems. His language was fresh and appropriate, incorporating such terms as *differential*, *integral*, *coordinate* and *function* [8]. His notations which we still use today, were clear and elegant. His unpublished manuscripts contain more than 50 different ways of writing coefficient systems, which he worked on during a period of 50 years beginning in 1678.

Leibniz used the word *resultant* for certain combinatorial sums of terms of a determinant. He proved various results on resultants including what is essentially Cramer's rule. He also knew that a determinant could be expanded using any column, what is now called Laplace expansion. As well as studying coefficient systems of equations which led him to determinants, Leibniz also studied coefficient systems of quadratic forms which led naturally towards matrix theory [9]. He thought about continuity, space and time [2].

In 1684 Leibniz published details of his differentiable calculus in 'Acta Eruditorum', a journal established in Leipzig two years earlier. He described a general method for finding *maxima* and *minima*, and drawing tangents to curves. The paper contained the

rules for computing the derivatives of powers, products and quotient.

In 1686 Leibniz published a paper on the principles of new calculus [3] in 'Acta Eruditorum'. Leibniz emphasized the inverse relationship between differentiation and integration in the fundamental theorem of calculus.

In 1692 Leibniz wrote a paper that set the basis of the *theory of envelopes*. This was further developed in another paper published on 1694 where he introduced for the first time the terms *coordinates* and *axes of coordinates*.

Leibniz published many papers on mechanical subjects as well [1]. In 1700 Leibniz founded the Berlin Academy and was its first president.

Leibniz's principal works are:

1) 'De Arte Combinatoria' (On the Art of Combination), 1666;
2) 'Hypothesis Physica Nova' (New Physical Hypothesis), 1671;
3) 'Dicours de Metaphysique' (Discourse on Metaphysics), 1686;
4) Unpublished Manuscripts on the Calculus of Concepts, 1690;
5) 'Nouveaux Essais sur L'entendement Humaine' (New Essays on Human Understanding), 1705;
6) 'Theodicee' (Theodicy), 1710;
7) 'Monadologia' (The Monadology), 1714.

See also

► History of Optimization

References

1. Aiton EJ (1985) Leibniz, A biography. Adam Hilger Ltd, Bristol
2. Anapolitanos D (1999) Leibniz: Representation, continuity and the spatiotemporal. Kluwer, Dordrecht
3. Boyer BB (1968) A history of mathematics. Wiley, New York
4. MacDonald GR (1984) Leibniz. Oxford Univ. Press, Oxford
5. O'Connor JJ (Oct. 1998) Gottfried Wilhelm von Leibniz. Dept. Math. and Statist. Univ. St. Andrews, Scotland), http://www.history.mcs.st-andrews.ac.uk/history/Mathematicians/Leibniz.html
6. Pereira ME (2000) Gottfried Wilhelm von Leibniz. http://www.geocities.com/Athens/Delphi/6061/Leibniz.html
7. Wingereid B (2000) Gottfried Wilhelm von Leibniz. http://www.phs.princeton.k12.oh.us/Public/Lessons/enl/wing.html
8. Woolhouse RS (ed) (1981) Leibniz: Metaphysics and philosophy of science. Oxford Univ. Press, Oxford
9. Zalta EN (2000) Gottfried Wilhelm von Leibniz. http://mally.stanford.edu/leibniz

Lemke Method

Lemke Algorithm

MICHAEL M. KOSTREVA
Department Math. Sci., Clemson University, Clemson, USA

MSC2000: 90C33

Article Outline

Keywords

Linear complementarity; Pivoting

The *linear complementarity problem* (LCP) is a well known problem in mathematical programming. Applications of the LCP to engineering, game theory, economics, and many other scientific fields have been found. The monograph of K.G. Murty [8] is a compendium of LCP developments. One of the most significant approaches to the solution of the linear complementarity problem is called Lemke's method or Lemke's algorithm. Two descriptions of the algorithm [6,7] provide many algorithmic proofs and details for the interested reader. Our treatment here is a sketch of the algorithm, together with pointers to related work in the literature.

There are some important related works for those who wish to solve LCP. A. Ravindran [10] provided a FORTRAN implementation of Lemke's algorithm in a set-up similar to the revised simplex method. C.B. Garcia [2] described some classes of matrices for which the associated LCPs can be solved by Lemke's algorithm. J.J.M. Evers [1] enlarged the range of application

of Lemke's algorithm, and showed that it could solve the bimatrix game. P.M. Pardalos and J.B. Rosen [9] presented a global optimization approach to LCP. D. Solow and P. Sengupta [11] proposed a finite descent theory for the linear complementarity problem. M.M. Kostreva [4] showed that without the *nondegeneracy assumption*, Lemke's algorithm may *cycle*, and showed that the minimum length of such a cycle is four.

The linear complementarity problem considered is: Given an $(n \times n)$-matrix M and an $(n \times 1)$ column vector q, problem LCP(q, M) is to find x (or prove that no such x exists) in $\mathbf{R}^n$ satisfying

$$y = Mx + q, \tag{1}$$

$$y_i \geq 0, \tag{2}$$

$$x_i \geq 0, \tag{3}$$

$$y_i \cdot x_i = 0, \tag{4}$$

for all i, $i = 1, \ldots, n$.

Clearly these conditions are equivalent to $y^\intercal x = 0$. The variables (y_i, x_i) are called a *complementary pair of variables*. Lemke's algorithm is organized relative to the following extended system of equations:

$$y = Mx + q + x_0 d, \tag{5}$$

where d is an $(n \times 1)$ column vector, and $x_0 \geq 0$. Relative to the vector d, it is only required that $(q + x_0 d) \geq 0$ for some $x_0 \geq 0$. It is assumed that the system of equations (5) is nondegenerate, that is, any solution has at most $n + 1$ zero values among the variables (y, x, x_0).

Lemke's Algorithm

If $q > 0$, terminate with a complementary feasible solution, $y = q$, $x = 0$.

If q has some negative component, then on the first pivot x_0 is increased until for the first time $y = q + x_0 d \geq 0$. When this occurs, some y variable, say y_r becomes zero. The first pivot is to exchange the variables x_0 and y_r. Now the variable x_0 is basic, and the variables y_r and x_r are two complementary non basic variables. If a pivot can be made on variable x_r (complement of the most recently pivoted member of the complementary pair), then it leads to another similar situation with another pair of complementary variables. If a pivot cannot be made, the sequence is terminated. If the variable x_0 becomes non basic (zero), a solution is at hand. If not, the pivoting continues uniquely, with each new set of equations containing a non basic complementary pair of variables, one of which is most recently made non basic. Due to the unique choices of pivot row and pivot column, finite termination must occur.

Under certain conditions, including the positive semidefinite matrices, the condition of termination without finding a pivot (also called secondary ray termination) can be shown to imply that the set $\{x\colon y = Mx + q \geq 0, \quad x \geq 0\}$ is empty. Under such conditions, Lemke's algorithm is said to *process the LCP*: either it is solved, or it is shown not to have a feasible solution. The set of all LCPs which Lemke's algorithm will process is unknown, but some recent papers shed light on its processing domain. Kostreva and M.M. Wiecek [5] use a multiple objective optimization approach which eventually results in a larger dimensioned LCP, while G. Isac, Kostreva and Wiecek [3] point out a set of problems which is impossible for Lemke's method to process.

Example 1 Consider the LCP corresponding to the quadratic programming problem

$$\begin{cases} \min & x_1^2 - 2x_1x_2 + x_2^2 + 3x_1 + x_2 \\ \text{s.t.} & 3x_1 + x_2 \geq 4 \\ & x_1 \geq 0,\ x_2 \geq 0. \end{cases}$$

Then $q = (-4, 3, 1)^\intercal$ and $M = [(0, -3, -1)^\intercal, (3, 2, -2)^\intercal, (1, -2, 2)^\intercal]$, and Lemke's algorithm requires four pivots to obtain the solution $x^* = (1, 1)^\intercal$, using the vector $d = (1, 1, 1)^\intercal$. It is noteworthy that the nondegeneracy assumption is not satisfied in this example, but Lemke's algorithm works anyway.

See also

- Convex-simplex Algorithm
- Linear Complementarity Problem
- Linear Programming
- Parametric Linear Programming: Cost Simplex Algorithm
- Sequential Simplex Method

References

1. Evers JJM (1978) More with the Lemke complementarity algorithm. Math Program 15:214–219
2. Garcia CB (1973) Some classes of matrices in linear complementarity theory. Math Program 5:299–310
3. Isac G, Kostreva MM, Wiecek MM (1995) Multiple objective approximation of feasible but unsolvable linear complementarity problems. J Optim Th Appl 86:389–405
4. Kostreva MM (1979) Cycling in linear complementarity problems. Math Program 16:127–130
5. Kostreva MM, Wiecek MM (1993) Linear complementarity problems and multiple objective programming. Math Program 60:349–359
6. Lemke CE (1965) Bimatrix equilibrium points and mathematical programming. Managem Sci 11:681–689
7. Lemke CE (1968) On complementary pivot theory in mathematics of the decision sciences. In: Dantzig GB, Veinott AF (eds) Amer. Math. Soc.
8. Murty KG (1988) Linear complementarity, linear and nonlinear programming. Heldermann, Berlin
9. Pardalos PM, Rosen JB (1988) Global optimization approach to the linear complementarity problem. SIAM J Sci Statist Comput 9:341–353
10. Ravindran A (1972) Algorithm 431-H. A computer routine for quadratic and linear programming problems. Comm ACM 15:818–820
11. Solow D, Sengupta P (1985) A finite descent theory for linear programming, piecewise linear minimization and the linear complementarity problem. Naval Res Logist Quart 32:417–431

Lexicographic Pivoting Rules

LexPr

TAMÁS TERLAKY
Department Comput. & Software,
McMaster University, West Hamilton, Canada

MSC2000: 90C05, 90C20, 90C33, 05B35, 65K05

Article Outline

Keywords

Pivot rules; Anticycling; Lexicographic ordering; LP; LCP; Oriented matroids

The general linear optimization (LO), linear programming (cf. ► Linear programming), problem will be considered in the standard primal form

$$\min\left\{c^\top x:\ Ax = b,\ x \geq 0\right\},$$

together with its standard dual

$$\max\left\{b^\top y:\ A^\top y \leq c\right\}.$$

One of the most efficient, and for a long time the only, practical method to solve LO problems was the simplex method of G.B. Dantzig. The *simplex method* is a *pivot algorithm* that traverses through feasible *basic solutions* while the objective value is improving. The simplex method is practically one of the most efficient algorithms but it is theoretically a finite algorithm only for *nondegenerate problems*.

A basis is called *primal degenerate* if at least one of the basic variables is zero; it is called *dual degenerate* if the reduced cost of at least one nonbasic variable is zero. In general, the basis is degenerate if it is either primal or dual, or both primal and dual degenerate. The LO problem is degenerate, if it has a degenerate basis. A pivot is called degenerate when after the pivot the objective remains unchanged. When the problem is *degenerate* the objective might stay the same in subsequent iterations and the simplex algorithm may *cycle*, i. e. starting from a basis, after some iterations the same basis is revisited and this process is repeated endlessly. Because the simplex method produces a sequence with monotonically improving objective values, the objective stays constant in a cycle, thus each pivot in the cycle must be degenerate. The possibility of cycling was recognized shortly after the invention of the simplex algorithm. Cycling examples were given by E.M.L. Beale [2] and by A.J. Hoffman [10]. Recently (1999) a scheme to construct cycling LO examples is presented in [9]. These examples made evident that extra techniques are needed to ensure finite termination of simplex methods. The first and widely used such tool is the *lexico-*

graphic simplex rule. Other techniques, like the least-index anticycling rules (cf. ► Least-index anticycling rules) and more general recursive schemes were developed more recently.

Lexicographic Simplex Methods

First we need to define an ordering, the so-called *lexicographic ordering* of vectors.

Lexicographic Ordering

An n-dimensional vector $u = (u_1, \dots, u_n)$ is called *lexicographically positive* or, in other words, *lexico-positive* if its first nonzero coordinate is positive, i. e. for a certain $j \leq n$ one has $u_i = 0$ for $i < j$ and $x_u > 0$. Observe, that the zero vector is the only lexico-nonnegative vector which is not lexico-positive. The vector u^0 is said to be lexicographically smaller than a vector u^1 when the difference $u^1 - u^0$ of the two vectors is lexico-positive. Further, if a finite set of vectors $\{u^0, \dots, u^k\}$ is given, then the vector u^0 is said to be lexico-minimal in the given set, when u^0 is lexicographically smaller than u^i for all $1 \leq i \leq k$.

The Lexicographic Primal Simplex Method

Cycling of the simplex method is possible only when the LO problem is degenerate. In that case possibly many variables are eligible to enter and to leave the basis. The lexicographic primal simplex rule makes the selection of the leaving variable uniquely determined when the entering variable is already chosen.

The Use of Lexicographic Ordering

At start a feasible *lexico-positive basis tableau* is given. A basis tableau is called lexico-positive if, except the reduced cost row, all of its row vectors are lexico-positive. Any feasible basis tableau can be made lexico-positive by a simple rearrangement of its columns. Specifically, we can take the solution column as the first one, and then take the current basic variables, in an arbitrary order, followed by the nonbasic variables, again in an arbitrary ordering.

The following lexicographic simplex pivot selection rule was first proposed by Dantzig, A. Orden and P. Wolfe [7].

0	Initialization. Let $T(B)$ be a given primal feasible lexico-positive basis tableau. (Fix the order of the variables.)
1	Entering variable selection. Choose a dual infeasible variable, i.e. one with negative reduced cost. Let its index be q. IF no such variable exists, THEN STOP; The tableau $T(B)$ is optimal and this way a pair of optimal solutions is obtained.
2	Leaving variable selection. Collect in column q all the candidate pivot elements that satisfy the usual pivot selection conditions of the primal simplex method. Let $K = \{i_1, \dots, i_k\}$ be the set of the indices of the candidate leaving variables. IF there is no pivot candidate, THEN STOP; The primal problem is unbounded, and so the dual problem is infeasible. IF there is a unique pivot candidate $\{p\} = K$ to leave the basis, THEN go to Step 3. IF there are more pivot candidates, THEN look at the row vectors t^i, $i \in K$, of the basis tableau (note that by construction x_i is the first coordinate of t^i). Let p be the pivot row if t^p is lexico-minimal in this set of row vectors.
3	Basis transformation. Pivot on (p, q). Go to Step 1.

The lexicographic primal simplex rule

The following two observations are important. First note that lexicographic selection plays role only when the leaving variable is selected. In that case some rows of the tableau are compared in the lexicographic ordering. If the basis variables were originally out right after the solution column, as proposed in order to get a lexico-positive initial tableau, then this comparison is already decided when one considers only the columns corresponding to the initial basis. This claim holds, because those columns form a basis, thus the related row vectors are linearly independent as well.

On the other hand, when the initial basis is the unit matrix, then at each pivot the basis inverse can be found, in the place of the initial unit matrix. When these

two observations are put together, it can be concluded that instead of using the rows of the basis tableau, the rows of the basis inverse headed by the corresponding solution coordinate, can be used in Step 2. to determine the unique leaving variable. As a consequence one do not need to calculate and store the complete basis tableau when implementing the lexicographic pivot rule. The solution and the basis inverse provide all the necessary information.

The lexicographic simplex method is finite. The finiteness proof is based on the following simple properties: There is a finite number of different basis tableaus. The first row of the tableau, i. e. the vector, having the objective value as its first coordinate followed by the reduced cost vector, strictly increases lexicographically at each iteration. This fact ensures that no basis can be revisited, thus cycling is impossible.

Lexicographic Ordering and Perturbation

Independent of [7], A. Charnes [4] developed a technique of *perturbation*, that resulted in a finite simplex algorithm. This algorithm turned out to be equivalent to the lexicographic rule. The perturbation technique is as follows. Let ϵ be a sufficiently small number. Let us replace b_i by $b_i + \sum_j a_{ij}\epsilon^j$ for all i. If ϵ is small enough then the resulted problem is nondegenerate. Moreover, starting from a given primal feasible basis, the primal simplex method applied to the new problem produces exactly the same pivot sequence as the lexicographic simplex method on the original problem.

In particular, when the problem is initialized with a feasible basis solution, it suffices to use the perturbation $b_i + \epsilon^i$. This way only the basis part of the coefficient matrix is used in Charnes' perturbation technic.

An appealing property of the perturbation technique is that actually it is not needed to perform the perturbation with a concrete ϵ. It can be done symbolically.

Lexicographic Dual Simplex Method

The dual simplex method is nothing else, than the primal simplex method applied to the dual problem, when the dual problem is brought in the primal standard form. This way it is straightforward to develop the lexicographic, or the equivalent perturbation technique for the dual simplex method.

Extensions

The lexicographic rule is extensively used in proving finiteness of pivot algorithms, see e. g. [1] for an application in a monotonic build-up scheme, [14] for further references in LO and [5] for references when lexicographic degeneracy resolution is applied for complementarity problems.

Lexicography and Oriented Matroids

Based on the perturbation interpretation, analogous lexicographic techniques and lexicographic pivoting rules were developed for oriented matroid programming [3] (cf. also ► Oriented matroids). These techniques were particularly interesting, because nondegenerate cycling [3,8] is possible in oriented matroids. An apparent difference between the linear and the oriented matroid context is that for oriented matroids none of the finite – recursive or least index type – rules yield a simplex method, i. e. a pivot method that preserves feasibility of the basis throughout. This discrepancy is also due to the possibility of nondegenerate cycling.

Interestingly, in the case of oriented matroid programming the finite lexicographic method of M.J. Todd [15,16] is the only one which preserves feasibility of the basis and therefore yields a finite simplex algorithm for oriented matroids.

The equivalence of Dantzig's self—dual parametric algorithm [6] and Lemke's complementary pivot algorithm [11,12] applied to the linear complementarity problem (cf. also ► Linear complementarity problem) defined by the primal and dual LO problem was proved by I. Lustig [13]. Todd's lexicographic pivot rule is essentially a lexicographic Lemke method (or the parametric perturbation method), when applied to the specific linear complementary problem defined by the primal-dual pair of LO problems. Hence, using the equivalence mentioned above a simplex algorithm for LO can be derived. However, it is more complicated to present this method in the linear optimization than in the complementarity context. Now Todd's rule will be sketched for the linear case.

0	Initialization. Let a lexico-positive feasible tableau $T(B)$ be given.
1	Entering variable selection. Collect all the dual infeasible variables as the set of candidate entering variables. Let their set of indices be denoted by K_D. IF no such variable exists, THEN STOP; The tableau $T(B)$ is optimal and this way a pair of optimal solutions is obtained. IF there is a unique $\{q\} = K_D$ candidate to enter the basis, THEN go to Step 2. IF there are more pivot candidates, THEN let q be the index of that variable whose column is lexico-minimal in the set K_D. (Analogous to the dual lexicographic simplex selection rule).
2	Leaving variable selection. Collect in column q all the candidate pivot elements that satisfy the usual pivot selection conditions of the primal simplex method. Let K_P be the set of the indexes of the candidate leaving variables. IF there is no pivot candidate, THEN STOP; the primal problem is unbounded, and so the dual problem is infeasible. IF there is a unique $\{p\} = K_P$ pivot candidate to leave the basis, THEN go to Step 3. IF there are more pivot candidates, THEN let p be the index of that variable whose row is lexico-minimal in the set K_P. (Analogous to the primal lexicographic simplex selection rule.)
3	Basis transformation. Pivot on $(p,\ q)$. Go to Step 1.

Todd's lexicographic Lemke rule (Phase II)

In Todd's rule the perturbation is done first in the right-hand side and then in the objective (with increasing order of the perturbation parameter ϵ). It finally gives a two phase simplex method. For illustration only the second phase [14] is presented here. Complete description of the algorithm can be found in [3,16].

This algorithm is not only a unique simplex method for oriented matroids, but it is a novel application of lexicography in LO as well.

See also

- ▶ Criss-cross Pivoting Rules
- ▶ Least-index Anticycling Rules
- ▶ Linear Programming
- ▶ Pivoting Algorithms for Linear Programming Generating Two Paths
- ▶ Principal Pivoting Methods for Linear Complementarity Problems
- ▶ Probabilistic Analysis of Simplex Algorithms

References

1. Anstreicher KM, Terlaky T (1994) A monotonic build-up simplex algorithm. Oper Res 42:556–561
2. Beale EML (1955) Cycling in the dual simplex algorithm. Naval Res Logist Quart 2:269–275
3. Bjorner A, Las Vergnas M, Sturmfels B, White N, Ziegler G (1993) Oriented matroids. Cambridge Univ. Press, Cambridge
4. Charnes A (1952) Optimality and degeneracy in linear programming. Econometrica 20(2):160–170
5. Cottle R, Pang JS, Stone RE (1992) The linear complementarity problem. Acad. Press, New York
6. Dantzig GB (1963) Linear programming and extensions. Princeton Univ. Press, Princeton
7. Dantzig GB, Orden A, Wolfe P (1955) Notes on linear programming: Part I – The generalized simplex method for minimizing a linear form under linear inequality restrictions. Pacific J Math 5(2):183–195
8. Fukuda K (1982) Oriented matroid programming. PhD Thesis Waterloo Univ.
9. Hall J, McKinnon KI (1998) A class of cycling counterexamples to the EXPAND anti-cycling procedure. Techn. Report Dept. Math. Statist. Univ. Edinburgh
10. Hoffman AJ (1953) Cycling in the simplex method. Techn Report Nat Bureau Standards 2974
11. Lemke CE (1965) Bimatrix equilibrium points and mathematical programming. Managem Sci 11:681–689
12. Lemke CE (1968) On complementary pivot theory. In: Dantzig GB, Veinott AF (eds) Mathematics of the Decision Sci. Part I. Lect Appl Math 11. Amer. Math. Soc., Providence, RI, pp 95–114
13. Lustig I (1987) The equivalence of Dantzig's self-dual parametric algorithm for linear programs to Lemke's algorithm for linear complementarity problems applied to linear programming. SOL Techn Report Dept Oper Res Stanford Univ 87(4)
14. Terlaky T, Zhang S (1993) Pivot rules for linear programming: A survey on recent theoretical developments. Ann Oper Res 46:203–233
15. Todd MJ (1984) Complementarity in oriented matroids. SIAM J Alg Discrete Meth 5:467–485
16. Todd MJ (1985) Linear and quadratic programming in oriented matroids. J Combin Th B 39:105–133

Linear Complementarity Problem

RICHARD W. COTTLE
Stanford University, Stanford, USA

MSC2000: 90C33

Article Outline

Keywords

Quadratic programming; Bimatrix games; Matrix classes; Equilibrium problems

Synonyms

LCP

Definition

In its standard form, a linear complementarity problem (LCP) is an inequality system stated in terms of a mapping $f\colon \mathbf{R}^n \to \mathbf{R}^n$ where $f(x) = q + Mx$. Given f, one seeks a vector $x \in \mathbf{R}^n$ such that for $i = 1, \ldots, n$,

$$x_i \geq 0, \quad f_i(x) \geq 0, \quad \text{and} \quad x_i f_i(x) = 0. \tag{1}$$

Because the affine mapping f is specified by the vector $q \in \mathbf{R}^n$ and the matrix $M \in \mathbf{R}^{n\times n}$, the problem is ordinarily denoted LCP(q, M) or sometimes just (q, M). A system of the form (1) in which f is not affine is called a *nonlinear complementarity problem* and is denoted NCP(f). The notation CP(f) is meant to cover both cases.

If $\overline{x}$ is a solution to (1) satisfying the additional *nondegeneracy condition* $\overline{x}_i + f_i(\overline{x}) > 0$, $i = 1, \ldots, n$, the indices i for which $\overline{x}_i > 0$ or $f_i(\overline{x}) > 0$ form complementary subsets of $\{1, \ldots, n\}$. This is believed to be the origin of the term *complementary slackness* as used in linear and nonlinear programming. It was this terminology that inspired the name *complementarity problem*.

Sources of Linear Complementarity Problems

The linear complementarity problem is associated with the Karush–Kuhn–Tucker necessary conditions of local optimality found in quadratic programming. This connection (as well as the more general connection of nonlinear complementarity problems with other types of nonlinear programs) was brought out in [1,2] and later in [3]. Finding solutions to such systems was one of the original motivations for studying the subject. Another was the finding of equilibrium points in bimatrix and polymatrix games. This kind of application was emphasized in [16] and [22]. These early contributions also included essentially the first algorithms for this class of problems. There are numerous applications of the linear and nonlinear complementarity problems in computer science, economics, various engineering disciplines, finance, game theory, and mathematics. One application of the LCP is in algorithms for the nonlinear complementarity problem. Descriptions of (and references to) these applications can be found in [5,27] and [17]. The survey article [10] is a rich compendium on engineering and economic applications of linear and nonlinear complementarity problems.

Equivalent Formulations

Whether linear or nonlinear, the complementarity problem expressed by the system (1) can be formulated in several equivalent ways. An obvious one calls for a solution (x, y) to the system

$$y - f(x) = 0, \quad x \geq 0, \quad x^\top y = 0. \tag{2}$$

Another is to find a zero x of the mapping

$$g(x) = \min\{x, f(x)\}, \tag{3}$$

where the symbol min $\{a, b\}$ denotes the componentwise minimum of the two n-vectors a and b. A third equivalent formulation asks for a fixed point of the mapping

$$h(x) = x - g(x),$$

that is, a vector $x \in \mathbf{R}^n$ such that $x = h(x)$.

The formulation given in (3) is related to the (often nonconvex) optimization problem:

$$\begin{cases} \min & x^\top f(x) \\ \text{s.t.} & f(x) \geq 0 \\ & x \geq 0. \end{cases} \tag{4}$$

In such a problem, the objective is bounded below by zero, thus any feasible solution of (4) for which the objective function $x^\intercal f(x) = 0$ must be a global minimum as well as a solution of (1). As it happens, there are circumstances (for instance, the monotonicity of the mapping f) under which all the local minima for the mathematical programming problem (4) must in fact be solutions of (3). See [28] for an extended discussion of this matter.

Also noteworthy is a result in [8] showing that the LCP is equivalent to solving a system of equations $y = \varphi(x)$ where the mapping $\varphi: \mathbf{R}^n \to \mathbf{R}^n$ is piecewise linear. In particular, LCP(q, M) is equivalent to finding a vector u such that

$$q + Mu^+ - u^- = 0,$$

where (for $i = 1, \ldots, n$, $u_i^+ = \max\{0, u_i\}$ and $u_i^- = -\min\{0, u_i\}$.

The Importance of Matrix Classes

The extensive literature of the LCP exhibits several main directions of study: the existence and uniqueness (or number of) solutions, mathematical properties of the problem, generalizations of the problem, algorithms, applications, and implementations.

Much of the theory of the linear complementarity problem is intimately linked in various ways to matrix classes. For instance, one of the earliest theorems on the existence of solutions to LCPs is due H. Samelson, R.M. Thrall and O. Wesler [30]. Motivated by a problem in structural mechanics, they showed that the LCP(q, M) has a unique solution for every $q \in \mathbf{R}^n$ if and only if the matrix M has positive principal minors. (That is, the determinant of every principal submatrix of M is positive.) The class of such matrices has come to be known as **P**, and its members are called **P**-*matrices*. (The Samelson–Thrall–Wesler theorem characterizes this class of matrices in terms of the LCP.) The class **P** includes all *positive definite* (**PD**) matrices, i. e., those square matrices M for which $x^\intercal Mx > 0$ for all $x \neq 0$. In the context of the LCP, the term **PD** does not require symmetry. An analogous definition (and usage) holds for *positive semidefinite* (**PSD**) matrices, namely, M is **PSD** if $x^\intercal Mx \geq 0$ for all x. Some authors refer to such matrices as *monotone* because of their connection with monotone mappings. **PSD**-matrices have the property that associated LCPs (q, M) are solvable whenever they are feasible, whereas LCPs (q, M) in which $M \in$ **PD** are always feasible and (since **PD**$\subset$**PSD**) are always solvable. This distinction is given a more general matrix form in [25,26]. There **Q** is defined as the class of all square matrices for which LCP(q, M) has a solution for all q and $\mathbf{Q}_0$ as the class of all square matrices for which LCP(q, M) has a solution whenever it is feasible. Although the goal of usefully characterizing the classes **Q** and $\mathbf{Q}_0$ has not yet been realized, much is known about some of their special subclasses. Indeed, there are now literally dozens of matrix classes for which LCP existence theorems have been established. See [5,27] and [17] for an abundance of information on this subject.

From the theoretical standpoint, the class of 'sufficient matrices' [6] illustrates the intrinsic role of matrix classes in the study of the LCP. A matrix $M \in \mathbf{R}^{n\times n}$ is *column sufficient* if

$$[x_i(Mx)_i \leq 0 \quad \forall i] \quad \Rightarrow \quad [x_i(Mx)_i = 0 \quad \forall i]$$

and *row sufficient* if $M^\intercal$ is column sufficient. When M is both row and column sufficient, it is called *sufficient*. Row sufficient matrices always have nonnegative principal minors, hence so do (column) sufficient matrices. These classes include both **P** and **PSD** as distinct subclasses. The row sufficient matrices form a subclass of $\mathbf{Q}_0$; this is not true of column sufficient matrices however. The column sufficient matrices $M \in \mathbf{R}^{n\times n}$ are characterized by the property that the solution set of LCP(q, M) is convex for every $q \in \mathbf{R}^n$. In the same spirit, a real $n \times n$ matrix M is row sufficient if and only if for every $q \in \mathbf{R}^n$, the solutions of the LCP(q, M) are precisely the optimal solutions of the associated quadratic program (4). Rather surprisingly, the class of sufficient matrices turns out to be identical to the matrix class $\mathbf{P}_*$ introduced in [19]. See [13] and [34].

Algorithms for Solving LCPs

The algorithms for solving linear complementarity problems are of two major types: pivoting (or, direct)

and iterative (or, indirect). Algorithms of the former type are finite procedures that attempt to transform the problem (q, M) to an equivalent system of the form (q', M') in which $q' \geq 0$. Doing this is not always possible; it depends on the problem data, usually on the matrix class (such as **P**, **PSD**, etc.) to which M belongs. When this approach works, it amounts to carrying out a *principal pivotal transformation* on the system of equations

$$w = q + Mz.$$

To such a transformation there corresponds an index set α (with complementary index set $\overline{\alpha} = \{1, \ldots, n\} \setminus \alpha$) such that the *principal submatrix* $M_{\alpha\alpha}$ is nonsingular. When this (block pivot) operation is carried out, the system

$$w_\alpha = q_\alpha + M_{\alpha\alpha} z_\alpha + M_{\alpha\overline{\alpha}} z_{\overline{\alpha}},$$
$$w_{\overline{\alpha}} = q_{\overline{\alpha}} + M_{\overline{\alpha}\alpha} z_\alpha + M_{\overline{\alpha}\overline{\alpha}} z_{\overline{\alpha}}$$

becomes

$$z_\alpha = q'_\alpha + M'_{\alpha\alpha} w_\alpha + M'_{\alpha\overline{\alpha}} z_{\overline{\alpha}},$$
$$w_{\overline{\alpha}} = q'_{\overline{\alpha}} + M_{\overline{\alpha}'\alpha} w_\alpha + M_{\overline{\alpha}'\overline{\alpha}} z_{\overline{\alpha}},$$

where

$$q'_\alpha = -M_{\alpha\alpha}^{-1} q_\alpha,$$
$$q'_{\overline{\alpha}} = q_{\overline{\alpha}} - M_{\overline{\alpha}\alpha} M_{\alpha\alpha}^{-1} q_\alpha,$$
$$M'_{\alpha\alpha} = M_{\alpha\alpha}^{-1},$$
$$M'_{\overline{\alpha}\alpha} = M_{\overline{\alpha}\alpha} M_{\alpha\alpha}^{-1},$$
$$M'_{\alpha\overline{\alpha}} = -M_{\alpha\alpha}^{-1} M_{\alpha\overline{\alpha}},$$
$$M'_{\overline{\alpha}\overline{\alpha}} = M_{\overline{\alpha}\overline{\alpha}} - M_{\overline{\alpha}\alpha} M_{\alpha\alpha}^{-1} M_{\alpha\overline{\alpha}}.$$

There are two main pivoting algorithms used in processing LCPs. The more robust of the two is due to C.E. Lemke [21]. *Lemke's method* embeds the LCP (q, M) in a problem having an extra 'artificial' nonbasic (independent) variable z_0 with coefficients specially chosen so that when z_0 is sufficiently large, all the basic variables become nonnegative. At the least positive value of z_0 for which this is so, there will (in the nondegenerate case) be (exactly) one basic variable whose value is zero. That variable is exchanged with z_0. Thereafter the method executes a sequence of (almost complementary) simple pivots. In each case, the variable *becoming* basic is the complement of the variable that became nonbasic in the previous exchange. The method terminates if either z_0 decreases to zero (in which case the problem is solved) or else there is no basic variable whose value decreases as the incoming nonbasic variable is increased. The latter outcome is called *termination on a secondary ray*. For certain matrix classes, termination on a secondary ray is an indication that the given LCP has no solution. Lemke's method is studied from this point of view in [7].

The other pivoting algorithm for the LCP is called the *principal pivoting method* (PPM), expositions of which are given in [3] and [5]. The algorithm two versions: symmetric and asymmetric. The former executes a sequence of principal (block) pivots or order 1 or 2, whereas the latter does sequences of almost complementary pivots, each of which results in a block principal pivot or order potentially larger than 2.

Iterative methods are often favored for the solution of very large linear complementarity problems. In such problems, the matrix M tends to be sparse (i. e., to have a small percentage of nonzero elements) and structured. Since iterative methods do not modify the problem data, these features of large scale problems can be used to advantage. Ordinarily, however, an iterative method does not terminate finitely; instead, it generates a convergent sequence of trial solutions. The older iterative LCP algorithms are based on equation-solving methods (e. g., *Gauss–Seidel*, *Jacobi*, and *successive overrelaxation*); the more contemporary ones are varieties of the *interior point* type. In addition to the usual concerns about practical performance, considerable interest attaches to the development of polynomial time algorithms. Not unexpectedly, the allowable analysis and applicability of iterative algorithms depend heavily on the matrix class to which M belongs. Details on several such algorithms are presented in [36,37], and the monographs [5,27] and [17].

Software

For decades researchers have experimented with computer codes for various linear (and nonlinear) complementarity algorithms. By the late 1990s, this activity reached the stage where the work could be distributed as something approaching commercial software. An overview of available software for complementarity problems (mostly nonlinear), is available as [35].

Some Generalizations

Both linear and nonlinear complementarity problems have been generalized in numerous ways. One of the earliest generalizations, given in [14] and [18], is the problem CP(K, f) of finding a vector x in the closed convex cone K such that $f(x) \in K^*$ (the dual cone) and $x^\intercal f(x) = 0$. Through this formulation, a connection can be made between complementarity problems and *variational inequality problems*, that is, problems VI(X, f) wherein one seeks a vector $x^* \in X$ (a nonempty subset of $\mathbf{R}^n$) such that

$$f(x^*)^\top (y - x^*) \geq 0 \quad \text{for all } y \in X.$$

It was established in [18] that when X is a closed convex cone, say K, with dual cone K^*, then CP(K, f) and VI(X, f) have exactly the same solutions (if any). See [15] for connections with variational inequalities, etc.

In [29] the generalized complementarity problem CP(K, f) defined above is considered as an instance of a *generalized equation*, namely to find a vector $x \in \mathbf{R}^n$ such that

$$0 \in f(x) + \partial \psi_K(x),$$

where ψ_K is the indicator function of the closed convex cone K and ∂ denotes the subdifferential operator as used in convex analysis.

Among the diverse generalizations of the linear complementarity problem, the earliest appears in [30]. There, for given $n \times n$ matrices A and B and n-vector c, the authors considered the problem of the finding n-vectors x and y such that

$$Ax + By = c, \quad x, y \geq 0 \quad \text{and} \quad x^\top y = 0.$$

A different generalization was introduced in [4]. In this sort of problem, one has an affine mapping $f(x) = q + Nx$ where N is of order $\sum_{j=1}^{k} p_j \times n$ partitioned into k blocks; the vectors q and $y = f(x)$ are partitioned conformably. Thus,

$$y^j = q^j + N^j x \quad \text{for } j = 1, \ldots, k.$$

The problem is to find a solution of the system

$$\begin{aligned} &y = q + Nx, \\ &x, y \geq 0, \\ &x_j \prod_{i=1}^{p_j} y_i^j = 0, \quad j = 1, \ldots, k. \end{aligned}$$

In recently years, many publications, e. g. [9] and [24], have further investigated this *vertical linear complementarity problem* (VLCP). Interest in the model which is at the heart of [30] and is now called the *horizontal linear complementarity problem* (HLCP) was revived in [38] where it is used as the conceptual framework for the convergence analysis of infeasible interior point methods. (The problem also comes up in [20].) In some cases, HLCPs can be reduced to ordinary LCPs. This subject is explored in [33] which gives an algorithm for doing this when it is possible. A further generalization called *extended linear complementarity problem* (ELCP) was introduced in [23] and subsequently developed in [11,12] and [32]. To this collection of LCP variants can be added the ELCP presented in [31]. The form of this model captures the previously mentioned HLCP, VLCP and ELCP.

See also

- ▶ Convex-simplex Algorithm
- ▶ Equivalence Between Nonlinear Complementarity Problem and Fixed Point Problem
- ▶ Generalized Nonlinear Complementarity Problem
- ▶ Integer Linear Complementary Problem
- ▶ LCP: Pardalos–Rosen Mixed Integer Formulation
- ▶ Lemke Method
- ▶ Linear Programming
- ▶ Order Complementarity
- ▶ Parametric Linear Programming: Cost Simplex Algorithm
- ▶ Principal Pivoting Methods for Linear Complementarity Problems
- ▶ Sequential Simplex Method
- ▶ Splitting Method for Linear Complementarity Problems
- ▶ Topological Methods in Complementarity Theory

References

1. Cottle RW (1964) Nonlinear programs with positively bounded Jacobians. Univ. Calif., Berkeley, CA
2. Cottle RW (1966) Nonlinear programs with positively bounded Jacobians. SIAM J Appl Math 14:147–158
3. Cottle RW, Dantzig GB (1968) Complementary pivot theory of mathematical programming. Linear Alg & Its Appl 1:103–125
4. Cottle RW, Dantzig GB (1970) A generalization of the linear complementarity problem. J Combin Th 8:79–90

5. Cottle RW, Pang JS, Stone RE (1992) The linear complementarity problem. Acad. Press, New York
6. Cottle RW, Pang JS, Venkateswaran V (1989) Sufficient matrices and the linear complementarity problem. Linear Alg & Its Appl 114/115:231–249
7. Eaves BC (1971) The linear complementarity problem. Managem Sci 17:612–634
8. Eaves BC, Lemke CE (1981) Equivalence of LCP and PLS. Math Oper Res 6:475–484
9. Ebiefung AA (1995) Existence theory and Q-matrix characterization for generalized linear complementarity problem. Linear Alg & Its Appl 223/224:155–169
10. Ferris MC, Pang JS (1997) Engineering and economic applications of complementarity problems. SIAM Rev 39:669–713
11. Gowda MS (1995) On reducing a monotone horizontal LCP to an LCP. Appl Math Lett 8:97–100
12. Gowda MS (1996) On the extended linear complementarity problem. Math Program 72:33–50
13. Guu S-M, Cottle RW (1995) On a subclass of P0. Linear Alg & Its Appl 223/224:325–335
14. Habetler GJ, Price AJ (1971) Existence theory for generalized nonlinear complementarity problems. J Optim Th Appl 7:223–239
15. Harker PT, Pang JS (1990) Finite-dimensional variational inequality and nonlinear complementarity problems: A survey of theory, algorithms and applications. Math Program B 48:161–220
16. Howson JT Jr (1963) Orthogonality in linear systems. Rensselaer Inst. Techn., Troy, NY
17. Isac G (1992) Complementarity problems. Lecture Notes Math, vol 1528. Springer, Berlin
18. Karamardian S (1971) Generalized complementarity problem. J Optim Th Appl 8:161–168
19. Kojima M, Megiddo N, Noma T, Yoshise A (1991) A unified approach to interior point algorithms for linear complementarity problems. Lecture Notes Computer Sci, vol 538. Springer, Berlin
20. Kuhn D, Löwen R (1987) Piecewise affine bijections of $\mathbf{R}^n$ and the equation $Sx^+ - Tx^- = y$. Linear Alg & Its Appl 96:109–129
21. Lemke CE (1965) Bimatrix equilibrium points and mathematical programming. Managem Sci 11:681–689
22. Lemke CE, Howson JT Jr (1964) Equilibrium points of bimatrix games. SIAM J Appl Math 12:413–423
23. Mangasarian OL, Pang JS (1995) The extended linear complementarity problem. SIAM J Matrix Anal Appl 16:359–368
24. Mohan SR, Neogy SK (1997) Vertical block hidden Z-matrices and the generalized linear complementarity problem. SIAM J Matrix Anal Appl 18:181–190
25. Murty KG (1968) On the number of solutions to the complementarity problem and spanning properties of complementary cones. Univ. Calif., Berkeley, CA
26. Murty KG (1972) On the number of solutions to the complementarity problem and spanning properties of complementary cones. Linear Alg & Its Appl 5:65–108
27. Murty KG (1988) Linear complementarity: linear and nonlinear programming. Heldermann, Berlin
28. Pang JS (1995) Complementarity problems. In: Horst R, Pardalos PM (eds) Handbook Global Optim. Kluwer, Dordrecht, pp 271–338
29. Robinson SM (1979) Generalized equations and their solutions, Part I: Basic theory. Math Program Stud 10:128–141
30. Samelson H, Thrall RM, Wesler O (1958) A partition theorem for Euclidean n-space. Proc Amer Math Soc 9:805–807
31. De Schutter B, De Moor B (1996) The extended linear complementarity problem. Math Program 71:289–326
32. Sznajder R, Gowda MS (1995) Generalizations of P_0- and P-properties; Extended vertical and horizontal LCPs. Linear Alg & Its Appl 223/224:695–715
33. Tütüncü RH, Todd MJ (1995) Reducing horizontal linear complementarity problems. Linear Alg & Its Appl 223/224:717–730
34. Väliaho H (1996) P_*-matrices are just sufficient. Linear Alg & Its Appl 239:103–108
35. Website: www.cs.wisc.edu/cpnet
36. Ye Y (1993) A fully polynomial-time approximation algorithm for computing a stationary point of the general linear complementarity problem. Math Oper Res 18:334–346
37. Yoshise A (1996) Complementarity problems. In: Terlaky T (ed) Interior point methods of mathematical programming. Kluwer, Dordrecht, pp 297–367
38. Zhang Y (1994) On the convergence of a class of infeasible interior-point algorithm for the horizontal linear complementarity problem. SIAM J Optim 4:208–227

Linear Optimization: Theorems of the Alternative

ThAlt

KEES ROOS
Department ITS/TWI/SSOR,
Delft University Technol.,
Delft, The Netherlands

MSC2000: 15A39, 90C05

Article Outline

Keywords

Inequality systems; Duality; Certificate; Transposition theorem

If one has two systems of *linear relations*, where each relation is either an *linear equation* (or *linear equality relation*) or a *linear inequality relation* (of type $>$, $\geq$, $<$, $\leq$ or $\neq$), and exactly one of the two systems has a solution, then one says that the two given systems are each others *alternative*. A mathematical theorem stating that two systems are alternative systems is called a *theorem of the alternative*, or also a *transposition theorem*. Many such theorems are known. The table lists ten results of this type, with their inventors and dates. The table is a modified version of tables of H. Greenberg [16] and in [8]. In each case the alternative systems are labelled by *a* and *b*, respectively.

Consider by way of example the two systems 4*a* and 4*b* in the table. The corresponding theorem of the alternative is known as *Farkas' lemma*. Assume that 4*a* has a solution x, so $Ax \leq b$. Then we have for each nonnegative vector y that $y^\top Ax \leq y^\top b$. Hence, if $y^\top A = 0$ then we will have $y^\top b \geq 0$. Thus it follows that if 4*a* has a solution then 4*b* does not have a solution. This is the easy part of the proof of Farkas' lemma. The proof of the other implication is much harder. For a discussion of several proof techniques, see ► Farkas lemma.

In the above example we used that for $y \geq 0$ the inequality $y^\top Ax \leq y^\top b$ is implied by the system $Ax \leq b$. Note that the *implied inequality* $y^\top Ax \leq y^\top b$ is obtained from the separate inequalities in $Ax \leq b$ by combining them in a linear fashion. Fixing y, one easily understands that the implied inequality has no solution x if and only if $y^\top A = 0$ and $y^\top b < 0$. Together with $y \geq 0$ these are precisely the relations in the alternative system 4*b*. Thus, it may be concluded that Farkas' lemma can be restated by saying that the system $Ax \leq b$ is feasible if and only if it does not imply (in a linear fashion) the 'contradiction' $0^\top x < 0$. The 'if'-part is obvious: if the system has an implied inequality $0^\top x < 0$ then it must be inconsistent. But the 'only if'-part is a very deep result: it states that if the system has no contradictory implied inequality then it has a solution. The other theorems of the alternative in the table admit a similar interpretation.

The relevance of a theorem of the alternative is the following. Given some system S of relations the crucial question is whether the system has a solution or not. Knowing the answer to this question one is able to answer many other questions. For example, if one has a *linear optimization problem LO* in the *standard form*

$$\min_x \left\{ c^\top x : \; Ax = b, \; x \geq 0 \right\},$$

a given real number $\underline{z}$ is a strict lower bound for the optimal value of the problem if and only if the system

$$Ax = b, \quad c^\top x \leq \underline{z}, \quad x \geq 0,$$

has no solution, i. e. is *infeasible*. On the other hand, a given real number $\overline{z}$ is an upper bound for the optimal

1	J.B.J. Fourier (1826) [4]
a	$Ax \leq 0, \; Bx < 0, \; Cx = 0$
b	$y^\top A + v^\top B + w^\top C = 0,$ $y \geq 0, \; 0 \neq v \geq 0$
2	P. Gordan (1873) [7]
a	$Ax > 0$
b	$y^\top A = 0, \; 0 \neq y \geq 0$
3	J.Farkas (1902) [3]
a	$Ax = b, \; x \geq 0$
b	$y^\top A \geq 0, \; y^\top b < 0$
4	Farkas (1902) [3]
a	$Ax \leq b$
b	$y \geq 0, \; y^\top A = 0, \; y^\top b < 0$
5	E. Stiemke (1915) [13]
a	$Ax = 0, \; x > 0$
b	$y^\top A \geq 0, \; y^\top A \neq 0$
6	W.B. Carver (1912) [2]
a	$Ax < b$
b	$y^\top A = 0, \; y \geq 0, \; y^\top b \leq 0, \; y \neq 0$
7	T.S. Motzkin (1936) [10]
a	$Ax \leq 0, \; Bx < 0$
b	$y^\top A + v^\top B = 0, \; y \geq 0, \; v \geq 0, \; v \neq 0$
8	J. Ville (1938) [15]
a	$Ax > 0, \; x > 0$
b	$y^\top A \leq 0, \; y \geq 0, \; y \neq 0,$ or $A^\top y \neq 0$
9	A.W. Tacket (1956) [14]
a	$Ax \geq 0, \; Ax \neq 0, \; Bx \geq 0, \; Cx = 0$
b	$y^\top A + v^\top B + w^\top C = 0, \; y > 0, \; v \geq 0$
10	D. Gale (1960) [5]
a	$Ax \leq b$
b	$y^\top A = 0, \; y^\top b = -1, y \geq 0$

Ten pairs of alternative systems

value of the problem if and only if the system

$$Ax = b, \quad c^\top x \geq z, \quad x \geq 0,$$

has a solution, i. e. is *feasible*.

If a system S has a solution then this is easy to certify, namely by giving a solution of the system. The solution then serves as a *certificate* for the feasibility of S. If S is infeasible, however, it is more difficult to give an easy certificate. One is then faced with the problem of how to certify a negative statement. This is in general a very nontrivial problem that also occurs in many real life situations. For example, when accused for murder, how should one prove his innocence? In circumstances like these it may be impossible to find an easy to verify certificate for the negative statement 'not guilty'. A practical solution is the rule 'a person is innocent until his/her guilt is certified'. Clearly, from the mathematical point of view this approach is unsatisfactory.

Now suppose that there is an alternative system T and there exists a theorem of the alternative for S and T. Then we know that exactly one of the two systems has a solution. Therefore, S has a solution if and only if T has no solution. In that case, any solution of T provides a certificate for the unsolvability of S. Thus it is clear that a theorem of the alternative provides an easy to verify certificate for the unsolvability of a system of linear relations.

The proof of any theorem of the alternative consists of two parts. Assuming the existence of a solution of one system one needs to show that the other system is infeasible, and vice versa. It has been demonstrated above for Farkas' lemma that one of the two implications is easy to prove. This seems to be true for each theorem of the alternative: in all cases one of the implications is almost trivial, but the other implication is highly nontrivial and very hard to prove. On the other hand, having proved one theorem of the alternative the other theorems of the alternative easily follow. In this sense one might say that all the listed theorems of the alternative are equivalent: accepting one of them to be true, the validity of each of the other theorems can be verified easily. The situation resembles a number of cities on a high plateau. Travel between them is not too difficult; the hard part is the initial ascent from the plains below [1].

It should be pointed out that Farkas' lemma, or each of the other theorems of the alternative, is equivalent to the most deep result in linear optimization, namely the duality theorem for linear optimization: this theorem can be easily derived from Farkas' lemma, and vice versa (cf. also ► Linear programming). In fact, in many textbooks on linear optimization the duality theorem is derived in this way [5,17], whereas in other textbooks the opposite occurs: the duality theorem is proved first and then Farkas' lemma follows as a corollary [11]. This phenomenon is a consequence of a simple, and basic, logical principle that any duality theorem is actually equivalent to a theorem of the alternative, as has been shown in [9].

Both the Farkas' lemma and the duality theorem for linear optimization can be derived from a more general result which states that for any *skew-symmetric matrix* K (i. e., $K = -K^\top$) there exists a vector x such that

$$Kx \geq 0, \quad x \geq 0, \quad x + Kx > 0.$$

This result is due to Tucker [14] who also derives Farkas' lemma from it, whereas A.J. Goldman and Tucker [6] show how this result implies the duality theorem for linear optimization. For recent proofs, see [12].

See also

- ► Farkas Lemma
- ► Linear Programming
- ► Motzkin Transposition Theorem
- ► Theorems of the Alternative and Optimization
- ► Tucker Homogeneous Systems of Linear Relations

References

1. Broyden CG (1998) A simple algebraic proof of Farkas' lemma and related theorems. Optim Methods Softw 3:185–199
2. Carver WB (1921) Systems of linear inequalities. A-MATH 23(2):212–220
3. Farkas J (1902) Theorie der Einfachen Ungleichungen. J Reine Angew Math 124:1–27
4. Fourier JBJ (1826) Solution d'une question particulière du calcul des inégalités. Nouveau Bull. Sci. Soc. Philomath. Paris, 99–100
5. Gale D (1960) The theory of linear economic models. McGraw-Hill
6. Goldman AJ, Tucker AW (1956) Theory of linear programming. In: Kuhn HW, Tucker AW (eds) Linear Inequalities and Related Systems. Ann Math Stud. Princeton Univ. Press, Princeton, 53–97

7. Gordan P (1873) Über die Auflösung Linearer Gleichungen mit Reelen Coefficienten. Math Ann 6:23–28
8. Mangasarian OL (1994) Nonlinear programming. No. 10 in Classics Appl Math. SIAM, Philadelphia
9. McLinden L (1975) Duality theorems and theorems of the alternative. Proc Amer Math Soc 53(1):172–175
10. Motzkin TS Beiträge zur Theorie der Linearen Ungleichungen, PhD Thesis, Baselxs
11. Padberg M (1995) Linear optimization and extensions. Algorithms and Combinatorics, vol 12. Springer, Berlin
12. Roos C, Terlaky T, Vial J-Ph (1997) Theory and algorithms for linear optimization. An interior approach. Wiley, New York
13. Stiemke E (1915) Über Positive Lösungen Homogener Linearer Gleichungen. Math Ann 76:340–342
14. Tucker AW (1956) Dual systems of homogeneous linear relations. In: Kuhn HW, Tucker AW (eds) Linear Inequalities and Related Systems. Ann Math Stud. Princeton Univ. Press, Princeton, 3–18
15. Ville J (1938) Sur la thèorie gènèrale des jeux où intervient l'habileté des joueurs. In: Ville J (ed) Applications aux Jeux de Hasard. Gauthier-Villars, Paris, pp 105–113
16. Website: www-math.cudenver.edu/~hgreenbe
17. Zoutendijk G (1976) Mathematical programming methods. North-Holland, Amsterdam

Linear Ordering Problem

LOP

PAOLA FESTA
Dip. Mat. e Inform., Universitá Salerno, Baronissi (SA), Italy

MSC2000: 90C10, 90C11, 90C20

Article Outline

Keywords

Combinatorial optimization; Greedy technique; Graph optimization; Branch and bound; Linear programming

The linear ordering problem (LOP) has a wide range of applications in several fields, such as scheduling, sports, social sciences, and economics. Due to its combinatorial nature, it has been shown to be *NP*-hard [5]. Like many other computationally hard problems, the linear ordering problem has captured the researcher attention for developing efficient solution procedures. A comprehensive treatment of the state-of-art approximation algorithms for solving the linear order problem is contained in [15]. The scope of this article is to introduce the reader to this problem, providing its definition and some of the algorithms proposed in literature for solving it efficiently.

Problem Description

The linear ordering problem (LOP) can be formulated as follows: Given a *complete digraph* $D_n = (V_n, E_n)$ on n nodes and given arc weights $c(i, j)$ for each arc $(i, j) \in E_n$, find a *spanning acyclic tournament* in D_n such that the sum of the weights of its arcs is as large as possible.

An equivalent mathematical formulation of LOP ([11]) is the following: Given a matrix of weights $E = \{e_{ij}\}_{m\times m}$, find a permutation p of the columns (and rows) in order to maximize the sum of the weights in the upper triangle. Formally, the problem is to maximize

$$C_E(p) = \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} e_{p_i p_j},$$

where p_i is the index of the column (and row) occupying the position i in the permutation.

The best known among the applications of LOP occurs in economics. In fact, it is equivalent to the so-called *triangulation problem for input-output tables*. In this economical application, the economy (regional or national) is subdivided into sectors. An $m \times m$ input-output matrix is then created, whose entry (i, j) represents the flow of money from the sector i to the sector j. The sectors have to be ordered so that suppliers tend to come first followed by costumers. This scope can be achieved by permuting the rows and the columns of the built matrix so that the sum of entries above the diagonal is maximized, which is exactly the objective of the linear ordering problem.

Review of Exact and Approximation Algorithms

The pioneer heuristic method for solving LOP has been proposed by H.B. Chenery and T. Watanabe [3]. Their method tries to obtain plausible rankings of the sectors of an input-output table in the triangulation problem by ranking first those sectors that have a small share of inputs from other sectors and of outputs to final demand. An extensive discussion about the heuristics proposed until 1981 can be found in [16], while more recent work has been done in [2,11]. In [11] a heuristic algorithm is proposed based on the *tabu search methodology* and incorporating strategies for search intensification and diversification are given. For search intensification M. Laguna and others experimented with *path relinking*, a strategy proposed in connection with tabu search by F. Glover and Laguna [6] and still rarely used in actual implementations. In [2] an algorithm is presented implementing a *scatter search* strategy, which is a population-based method that has been shown to lead to promising outcomes for solving combinatorial and nonlinear difficult problems.

The development of exact algorithms for LOP can be seen connected to the development of methods for solving general integer programming problems, since any such method can be slightly modified to solve the triangulation problem. Most of those exact algorithms belong either to the branch and bound family or to the linear programming methods.

Branch and Bound Algorithms

One of the earliest published computational results using a branch and bound strategy is due to J.S. DeCani in 1972 [4]. He originally studied how to rank n objects on the basis of a number of paired comparisons. Since k persons have to pairwise compare n objects according to some criterion, a matrix $E = \{e_{ij}\}$ is built, where e_{ij} is the number of persons that prefer object i to object j. The problem is to find a linear ranking of the objects reflecting the outcome of the experiment as closely as possible. In the branch and bound strategy proposed by DeCani partial rankings are built up and each branching operation in the tree corresponds to inserting a further object at some position in the partial ranking. At level n of the tree a complete ranking of the objects is found. The upper bounds are exploited in the usual way for backtracking and excluding parts of the tree from further consideration.

A further method for solving LOP is the *lexicographic search algorithm* proposed in [9,10]. It lexicographically enumerates all permutations of the n sectors by fixing at level k of the enumeration tree the kth position of the permutations. In more detail, if at level k a node is generated, then the first k positions $\sigma(1)$, …, $\sigma(k)$ are fixed. Based on this fixing several Helmstädter's conditions can be tested. If one of them is violated, then there is no relatively optimum having $\sigma(1)$, …, $\sigma(k)$ in the first k positions. Therefore, the node currently under consideration can be ignored and a backtracking is performed. By using this method all relatively optimum solutions are enumerated, since there is no bounding according to objective function values. At the end the best one among them is kept. Starting from lexicographic search, [8] proposed a lexicographic branch and bound scheme.

Other authors have proposed branch and bound methods, such as [7,12], and [14].

Linear Programming Algorithms

All linear programming approaches are based on the consideration that the triangulation problem can be formulated as a 0–1 integer programming problem using the 3-dicycle inequalities. In [13] the LP relaxation using the tournament polytope P_C^n is proposed and the corresponding full linear program is solved in its dual version. In [1] LP relaxation is used for solving scheduling problems with precedence constraints. It is easy to see that the scheduling problem of minimizing the total weighted completion time of a set of processes on a single processor can be formulated as a linear ordering problem.

Other possibilities for theoretically solving linear ordering problems are methods as *dynamic programming* or by formulating the problem as *quadratic assignment problem* ([10]).

See also

- ► Assignment and Matching
- ► Assignment Methods in Clustering
- ► Bi-objective Assignment Problem
- ► Communication Network Assignment Problem
- ► Complexity Theory: Quadratic Programming

- ▶ Feedback Set Problems
- ▶ Frequency Assignment Problem
- ▶ Generalized Assignment Problem
- ▶ Graph Coloring
- ▶ Graph Planarization
- ▶ Greedy Randomized Adaptive Search Procedures
- ▶ Maximum Partition Matching
- ▶ Quadratic Assignment Problem
- ▶ Quadratic Fractional Programming: Dinkelbach Method
- ▶ Quadratic Knapsack
- ▶ Quadratic Programming with Bound Constraints
- ▶ Quadratic Programming Over an Ellipsoid
- ▶ Quadratic Semi-assignment Problem
- ▶ Standard Quadratic Optimization Problems: Algorithms
- ▶ Standard Quadratic Optimization Problems: Applications
- ▶ Standard Quadratic Optimization Problems: Theory

References

1. Boenchendorf K (1982) Reihenfolgenprobleme/Mean-flow-time sequencing. Math Systems in Economics. Athenäum–Hain–Scriptor–Hanstein, Königstein/Ts.
2. Campos V, Glover F, Laguna M, Martí R (1999) An experimental evaluation of a scatter search for the linear ordering problem. Manuscript Apr
3. Chenery HB, Watanabe T (1958) International comparisons of the structure of production. Econometrica 26(4):487–521
4. DeCani JS (1972) A branch &bound algorithm for maximum likelihood paired comparison ranking. Biometrika 59:131–135
5. Garey MR, Johnson DS (1979) Computers and intractability: A guide to the theory of NP-completeness. Freeman, New York
6. Glover F, Laguna M (1997) Tabu search. Kluwer, Dordrecht
7. Hellmich K (1970) Ökonomische Triangulierung. Heft 54. Rechenzentrum Graz, Graz
8. Kaas R (1981) A branch&bound algorithm for the acyclic subgraph problem. Europ J Oper Res 8:355–362
9. Korte B, Oberhofer W (1968) Zwei Algorithmen zur Lösung eines Komplexen Reihenfolgeproblems. Unternehmensforschung 12:217–231
10. Korte B, Oberhofer W (1969) Zur Triangulation von Input-Output Matrizen. Jahrbuch f Nat Ok u Stat 182:398–433
11. Laguna M, Martí R, Campos V (1999) Intensification and diversification with elite tabu search solutions for the liner ordering problem. Comput Oper Res 26:1217–1230
12. Lenstra jr. HW (1973) The acyclic subgraph problem. Techn Report Math Centrum Amsterdam BW26
13. Marcotorchino JF, Mirchaud P (1979) Optimisation en analyse ordinale des donnees. Masson, Paris
14. Poetsch G (1973) Lösungsverfahren zur Triangulation von Input-Output Tabellen. Heft 79. Rechenzentrum Graz, Graz
15. Reinelt G (1985) The linear ordering problem: Algorithms and applications. In: Hofmann HH, Wille R (eds) Res. and Exposition in Math., vol 8. Heldermann, Berlin
16. Wessels H (1981) Computers and intractability: A guide to the theory of NP-completeness. Beiträge zur Strukturforschung, vol 63. Deutsches Inst. Wirtschaftsforschung, Berlin
17. Whitney H (1935) On the abstract properties of linear dependence. Amer J Math 57:509–533

Linear Programming

LP

PANOS M. PARDALOS
Center for Applied Optim., Department Industrial and Systems Engineering, University Florida, Gainesville, USA

MSC2000: 90C05

Article Outline

Keywords

Linear programming; Basic solution; Simplex method; Pivoting; Nondegenerate

Linear programming (LP) is a fundamental optimization problem in which a linear objective function is to be optimized subject to a set of linear constraints. Due to the wide applicability of linear programming models, an immense amount of work has appeared regarding theory and algorithms for LP, since G.B. Dantzig proposed the simplex algorithm in 1947. It is not surprizing that in a recent survey of Fortune 500 companies, 85% of those responding said that they had used linear programming. The history, theory, and applications of linear programming may be found in [3]. Several books

have been published on the subject (see the references section).

Problem Description

Consider the linear programming problem (in standard form):

$$\begin{cases} \min & c^\top x \\ \text{s.t.} & Ax = b, \\ & x \geq 0 \end{cases} \tag{1}$$

where $c \in \mathbf{R}^n$, $b \in P^m$ and A is an $m \times n$ matrix of rank m (i. e. we do not have any redundant constraints). The feasible domain

$$P = \{x \in \mathbb{R}^n : \ Ax = b, \ x \geq 0\}$$

is a polytope. We assume that (1) has a finite optimal solution. Let B be a submatrix of A formed by m linearly independent columns. We may assume that $A = [B, N]$, i. e. the first columns of A are linearly independent. Then the linear system $Bx_B = b$ has a unique solution. If $x = (x_B, 0)$ then $Ax = b$ and $x = (x_B, 0)$ is called a *basic solution*. The components of x associated with the columns of B are called *basic variables*. If one of the basic variables in a basic solution is zero, that solution is called a *degenerate basic solution*. A basic solution that is feasible (i. e. $x \geq 0$) is called a *basic feasible solution*.

The following theorem identifies the special importance of the basic feasible solutions.

Theorem 1 *Assume that P in (1) is nonempty. Then a feasible point $x \in P$ is a vertex of P if and only if x is a basic feasible solution.*

Existence of basic feasible solutions is established by the following fundamental theorem of linear programming.

Theorem 2 *Given the linear programming problem (1), the following statements are true:*

1) *If P is nonempty, there exists a basic feasible solution.*
2) *If (1) has an optimal solution, then there is an optimal basic feasible solution.*

Therefore, the linear programming problem can be solved by searching among its basic feasible solutions (i. e. vertices of P). Since there are at most

$$\binom{m}{n}$$

basic solutions, the above theorem gives a finite, but a very inefficient algorithm. A more systematic search among the basic feasible solutions, is given by the simplex method, which was developed by Dantzing in 1947.

The Simplex Method

The simplex method has a simple geometric motivation which is described by the following two phases.

I	An initial vertex x_0 of P (basic feasible solution) is computed.
II	Starting from the vertex x_0, a sequence of vertices $x_0, \ldots, x_N$ is computed such a way that x_{i+1} is adjacent to x_i, $i = 0, \ldots, N-1$, and such that $c^\top x_{i+1} < c^\top x_i$. The method terminates if either none of the edges adjacent to x_N is decreasing the objective function (i.e., x_N is the solution) or if an unbounded edge adjacent to x_N is found, improving the objective function (i.e. the problem is unbounded).

Each step of the simplex method, moving from one vertex to an adjacent one, is called *pivoting*. The integer N gives the number of pivot steps in the simplex method. Phase I can be solved in a similar way to Phase II. In problems of the canonical form:

$$\begin{cases} \min & c^\top x \\ \text{s.t.} & Ax = b, \\ & x \geq 0, \quad b \geq 0, \end{cases} \tag{2}$$

there is no need for Phase I, because an initial vertex ($x_0 = 0$) is at hand. We start by considering Phase II of the simplex method, by assuming that an initial vertex (basic feasible solution) is available. Let x_0 be a basic feasible solution with $x_{10}, \ldots, x_{m0}$ its basic variables, and let $B = \{A_{B(i)} : i = 1, \ldots, m\}$ the corresponding *basis*. If A_j denotes the jth column of A, ($A_j \notin B$), then

$$\sum_{i=1}^{m} x_{ij} A_{B(i)} = A_j. \tag{3}$$

In addition,

$$\sum_{i=1}^{m} x_{i0} A_{B(i)} = b. \tag{4}$$

Multiply (3) by $\theta > 0$ and subtract the result from (4) to obtain:

$$\sum_{i=1}^{m}(x_{i0} - \theta x_{ij})A_{B(i)} + \theta A_j = b. \tag{5}$$

Assume that x_0 is nondegenerate. How much can we increase θ and still have a solution? We can increase θ until the first component of $(x_{i0} - \theta x_{ij})$ becomes zero or equivalently

$$\theta_0 = \min_i \left\{ \frac{x_{i0}}{x_{ij}} :\ x_{ij} > 0 \right\}. \tag{6}$$

If $\theta_0 = x_{l0}/x_{lj}$, then column A_l leaves the basis and A_j enters the basis.

If a tie occurs in (6), then the new solution is degenerate. In addition, if all $x_{ij} \le 0$, then we move arbitrarily far without becoming infeasible. In that case the problem is unbounded.

Define the new point $x_0{}'$ by

$$x'_{i0} = \begin{cases} x_{i0} - \theta x_{ij}, & i \neq l, \\ \theta_0, & i = l, \end{cases} \tag{7}$$

and

$$B'(i) = \begin{cases} B(i), & i \neq l, \\ j, & i = l. \end{cases}$$

It is easy to see that the m columns $A_{B'(i)}$ are linearly independent. Let

$$\sum_{i=1}^{m} x_i A_{B'(i)} = x_l A_j + \sum_{\substack{i=1 \\ i \neq l}}^{m} x_i A_{B(i)=0}.$$

Using (3) we have:

$$\sum_{\substack{i=1 \\ i \neq l}}^{m}(a_i x_{ij} + a_i)A_{B(i)} + a_l x_{lj} A_{B(l)} = 0$$

and by linear independence of the columns $A_{B(i)}$ we have

$$a_l = 0, \quad a_i(l + x_{ij}) = 0 \to a_1, \ldots, a_m = 0.$$

Hence, the new point $x_0{}'$ whose basic variables are given by (7) is a new basic feasible solution. When the basic feasible solution x_0 is degenerate then some of the basic variables are zero. Therefore more than $n-m$ of the constraints $x_j \ge 0$ are satisfied as equations (are active) and so x_0 satisfies more than n equations. From (6) it follows that if $x_{i0} = 0$ and the corresponding $x_{ij} > 0$, then $\theta_0 = 0$ and therefore we remain at the same vertex.

Note that when a basic feasible solution x_0 is degenerate, there can be an enormous number of basis associated with it. In fact, if x_0 has $k > m$ positive components, then there may be as many as $\binom{n-k}{n-m}$ different bases. In that case we may compute x_0 as many times as there are basis, but the set of variables that we label basic and nonbasic are different.

The cost (value of objective function) as a basic feasible solution x, with corresponding basis B is:

$$z_0 = \sum_{l=1}^{n} x_{i_0} c_{B(i)}$$

Suppose we bring column A_j into the new basis. The following economic interpretation can be used to select the pivot column A_j: For every unit of the variable x_j that enters the basis, an amount x_{ij} of each of the variables $x_{B(i)}$ must leave. Hence, a unit increase in the variable x_j results in a net change in the cost, equal to:

$$\bar{c}_j = c_j - z_j$$

(relative cost of column j), where $z_j = \sum_{i=1}^{m} x_{ij} c_{B(i)}$. It is profitable to bring column j into the basis exactly when $\bar{c}_j < 0$. Choosing the most negative $\bar{c}_j$ corresponds to a kind of steepest descent. However, many other selection criteria can be used (e. g., Blad's rule, etc).

If all reduced costs satisfy $\bar{c}_j \ge 0$, then we are at an optimal solution and the simplex method terminates. Note that relations (1) can be expressed in matrix notation by:

$$BX = A \quad \text{or} \quad X = B^{-1}A,$$

that is, the matrix $X = (x_{ij})$ is obtained by diagonalizing the basic columns of A. Then

$$z_j = \sum_{l=1}^{m} x_{ij} c_{B(i)} \quad \text{or} \quad z^\top = c_B^\top X = c_B^\top B^{-1} A.$$

Suppose $\bar{c} = c - z \ge 0$. Let y be a feasible point. Then,

$$c^\top y \ge z^\top y \ge c_B^\top B^{-1} A y = c_B^\top B^{-1} b = c^\top x_0$$

and therefore x_0 is an optimal solution.

Under the assumption of nondegeneracy with our pivot selection, $x_{l0} > 0$ (see (6)) and

$$\bar{z}_0 = z_0 - \frac{x_{l0}}{x_{lj}}(z_j - c_j) > z_0 \quad (z_j - c_j < 0).$$

Note that corresponding to any basis there is a unique z_0, and hence, we can never return to a previous basis. Therefore, each iteration gives a different basis and the simplex method terminator after $N \leq \binom{n}{m}$ pivots.

See also

- Affine Sets and Functions
- Carathéodory Theorem
- Convex-simplex Algorithm
- Criss-cross Pivoting Rules
- Farkas Lemma
- Gauss, Carl Friedrich
- Global Optimization in Multiplicative Programming
- History of Optimization
- Kantorovich, Leonid Vitalyevich
- Krein–Milman Theorem
- Least-index Anticycling Rules
- Lemke Method
- Lexicographic Pivoting Rules
- Linear Complementarity Problem
- Linear Optimization: Theorems of the Alternative
- Linear Space
- Motzkin Transposition Theorem
- Multiparametric Linear Programming
- Multiplicative Programming
- Parametric Linear Programming: Cost Simplex Algorithm
- Pivoting Algorithms for Linear Programming Generating Two Paths
- Principal Pivoting Methods for Linear Complementarity Problems
- Probabilistic Analysis of Simplex Algorithms
- Sequential Simplex Method
- Simplicial Pivoting Algorithms for Integer Programming
- Tucker Homogeneous Systems of Linear Relations

References

1. Ahuja RK, Magnanti TL, Orlin JB (1993) Network flows: Theory, algorithms and applications. Prentice-Hall, Englewood Cliffs, NJ
2. Bertsimas D, Tsistklis JN (1997) Introduction to linear optimization. Athena Sci., Belmont, MA
3. Dantzig GB (1963) Linear programming and extensions. Princeton Univ. Press, Princeton
4. Fang S-C, Puthenpura S (1993) Linear optimization and extensions. Prentice-Hall, Englewood Cliffs, NJ
5. Papadimitriou CH, Steiglitz K (1982) Combinatorial optimization: Algorithms and complexity. Prentice-Hall, Englewood Cliffs, NJ
6. Roos C, Terlaky T, Vial J-Ph (1998) Theory and algorithms for linear optimization: An interior point approach. Wiley, New York

Linear Programming: Interior Point Methods

KURT M. ANSTREICHER
University Iowa, Iowa City, USA

MSC2000: 90C05

Article Outline

Keywords

Linear programming; Interior point methods; Polynomial time algorithm

An enormous amount of research on interior point algorithms for linear programming (LP) has been conducted since N.K. Karmarkar [8] announced his celebrated *projective algorithm* in 1984. Interior point algorithms for LP are interesting for two different reasons. First, many interior point methods are *polynomial time algorithms* for LP. Consider a standard form problem:

$$\text{LP} \quad \begin{cases} \min & c^\top x \\ \text{s.t.} & Ax = b \\ & x \geq 0, \end{cases}$$

where A is an $m \times n$ matrix. For the purpose of characterizing the complexity of algorithms it is common to assume that the data of LP is integral. If L is the number of bits required to encode the data, then an algorithm for LP is polynomial time if the number of

operations required to solve LP is a polynomial function of n, m, and L. Throughout we use 'operations' to refer to arithmetic operations in infinite precision, although for an algorithm to be rigorously polynomial time in the rational model of computation the number of bits required to perform all computations should also be polynomially bounded. Karmarkar's projective algorithm solves LP in $O(nL)$ iterations and $O(n^4L)$ total operations. This overall complexity is required to obtain a solution of LP whose objective is within a tolerance $2^{-O(L)}$ of optimality; an exact optimal solution can then be obtained using a 'rounding' procedure in $O(n^3)$ operations. Karmarkar also described a *partial updating* procedure that reduced the overall complexity of his algorithm to $O(n^{3.5}L)$ operations. The idea of partial updating is to allow for some error in the specification of the projection equations that are solved on each iteration of the algorithm.

Interior point algorithms are also interesting because they perform well in practice. When the projective algorithm was first announced Karmarkar made well-publicized claims that his algorithm was several times faster than the simplex method in solving large LP problems. It was eventually discovered that most of Karmarkar's claims were actually for an implementation of the *affine scaling algorithm*, a simplified version of Karmarkar's algorithm that avoids the use of projective transformations. Initial attempts to replicate Karmarkar's results were mainly failures, but eventually it was convincingly established that interior point algorithms are highly competitive with the simplex method on large scale problems.

The announcement of Karmarkar's algorithm led to the development of a variety of different types of interior point methods for LP. The simplest of these are affine scaling methods, which were independently devised by E. Barnes [2] and R.J. Vanderbei, M.J. Meketon, and B.A. Freedman [21]. It was eventually realized that in fact the affine scaling method was discovered by I.I. Dikin [3] in 1967. The affine scaling method is not a polynomial time algorithm for LP, and it is now known that the algorithm may not even converge if the stepsize is too long [12]. Nevertheless its practical performance is often quite good, as indicated by Karmarkar's early claims.

Another type of interior point method, the *path following algorithm*, was discovered by J. Renegar [17]. Renegar's algorithm requires only $O(\sqrt{n}L)$ iterations to solve LP, as opposed to $O(nL)$ iterations for Karmarkar's algorithm. By adapting Karmarkar's partial updating technique to the path following framework, C.C. Gonzaga [5] and P.M. Vaidya [19] devised the first algorithms for LP with overall complexities of $O(n^3L)$ operations. The iterates of path following algorithms lie in a small neighborhood of the *central path* or *central trajectory*, which is defined to be the set of minimizers of the *logarithmic barrier function*

$$f_\mu(x) = \frac{c^\top x}{\mu} - \sum_{i=1}^{n} \ln(x_i),$$

over $\{x: Ax = b, x > 0\}$, for $\mu \in (0, \infty)$. Later C. Roos and J.-Ph. Vial [18], and Gonzaga [6] developed 'long step' path following algorithms. These algorithms are based on properties of the central path, but the iterates are not constrained to remain in a small neighborhood of the path. Long step path following algorithms are very closely related to the classical sequential unconstrained minimization technique (SUMT) of A.V. Fiacco and G.P. McCormick [4].

A different class of interior point algorithms is based on Karmarkar's use of a *potential function*, a surrogate for the original objective, to monitor the progress of his projective algorithm. Gonzaga [7] and Y. Ye [23] devised the first *potential reduction algorithms*. These algorithms are based on reducing a potential function but do not employ projective transformations. Ye's potential reduction algorithm requires $O(\sqrt{n}L)$ iterations, like path following algorithms, and provides an $O(n^3L)$ algorithm for LP when implemented with partial updating.

All of the algorithms mentioned to this point are based on solving LP, or alternatively the dual problem:

$$\text{LD} \quad \begin{cases} \max & b^\top y \\ \text{s.t.} & A^\top y + s = c \\ & s \geq 0. \end{cases}$$

Algorithms for solving LP typically generate feasible solutions to LD, and vice versa, but the algorithms are not symmetric in their treatment of the two problems. A different class of interior point methods, known as *primal-dual algorithms*, is completely symmetric in the

variables x and s. Primal-dual algorithms are based on applying Newton's method directly to the system of equations:

$$\mathrm{PD}(\mu) \quad \begin{cases} Ax = b \\ A^\top y + s = c \\ x \circ s = \mu e, \end{cases}$$

where $e \in \mathbf{R}^n$ is the vector of ones, μ is a positive scalar, and $x \circ s$ is the vector whose ith component is $x_i s_i$. Solutions $x > 0$ and $s > 0$ to PD(μ) are exactly on the central paths for LP and LD, respectively. Most primal-dual algorithms fit into the path following framework. The idea of a primal-dual path following algorithm was first suggested by N. Megiddo [13], and complete algorithms were first devised by R.C. Monteiro and I. Adler [15] and M. Kojima, S. Mizuno, and Y. Yoshise [10]. It is widely believed that primal-dual methods are in practice the best performing interior point algorithms for LP.

One advantage of the system PD(μ) is that Newton's method can be applied even when the current $x > 0$ and $s > 0$ are not feasible in LP and LD. This *infeasible interior point* (IIP) strategy was first employed in the OB1 code of I.J. Lustig, M.E. Marsten, and D.F. Shanno [11]. The solution to the Newton equations with $\mu = 0$ is referred to as the predictor, or primal-dual affine scaling direction, while the solution with $\mu = x^\top s/n$, for the current solutions x and s, is called the *corrector*, or *centering direction*. The primal-dual *predictor-corrector algorithm* alternates between the use of these two directions. One implementation of the IIP predictor-corrector strategy, due to S. Mehrotra [14], has worked particularly well in practice. Despite the fact that primal-dual IIP algorithms were very successfully implemented it proved to be quite difficult to characterize the convergence of these methods. The first such analyses, by Kojima, Megiddo, and Mizuno [9], and Y. Zhang [25], were followed by a large number of papers giving convergence/complexity results for various IIP algorithms. Ye, M.J. Todd, and Mizuno [24] devised a 'selfdual homogeneous' interior point method that has many of the practical features of IIP methods but at the same time has stronger convergence properties. An implementation of the homogeneous algorithm [22] exhibits excellent behavior, particularly when applied to infeasible or near-infeasible problems.

Many interior point algorithms for LP can be extended to more general optimization problems. Primal-dual algorithms generalize very naturally to the monotone linear complementarity problem (LCP; cf. ► linear complementarity problem); in fact many papers on primal-dual algorithms (for example [25]) are written in terms of the LCP. As a result these algorithms immediately provide interior point solution methods for convex quadratic programming (QP) problems. Interior point algorithms can also be generalized to apply to quadratically constrained quadratic programming (QCQP), optimization over *second order cone* (SOC) constraints, and semidefinite programming (SDP); for details on these and other extensions see [16]. The application of interior point methods to SDP has particularly rich applications, as described in [1], and [20], and remains the topic of extensive research.

See also

- ► Entropy Optimization: Interior Point Methods
- ► Homogeneous Selfdual Methods for Linear Programming
- ► Interior Point Methods for Semidefinite Programming
- ► Linear Programming: Karmarkar Projective Algorithm
- ► Potential Reduction Methods for Linear Programming
- ► Sequential Quadratic Programming: Interior Point Methods for Distributed Optimal Control Problems
- ► Successive Quadratic Programming: Solution by Active Sets and Interior Point Methods

References

1. Alizadeh F (1995) Interior point methods in semidefinite programming with applications to combinatorial optimization. SIAM J Optim 5:13–51
2. Barnes ER (1986) A variation on Karmarkar's algorithm for solving linear programming problems. Math Program 36:174–182
3. Dikin II (1967) Iterative solution of problems of linear and quadratic programming. Soviet Math Dokl 8:674–675
4. Fiacco AV, McCormick GP (1990) Nonlinear programming, sequential unconstrained minimization techniques. SIAM, Philadelphia
5. Gonzaga CC (1989) An algorithm for solving linear programming problems in O(n3L) operations. In: Megiddo N

(ed) Progress in Mathematical Programming. Springer, Berlin, pp 1–28
6. Gonzaga CC (1991) Polynomial affine algorithms for linear programming. Math Program 49:7–21
7. Gonzaga CC (1991) Large-step path-following methods for linear programming, Part I: Barrier function method. SIAM J Optim 1:268–279
8. Karmarkar N (1984) A new polynomial-time algorithm for linear programming. Combinatorica 4:373–395
9. Kojima M, Megiddo N, Mizuno S (1993) A primal-dual infeasible-interior-point algorithm for linear programming. Math Program 61:263–280
10. Kojima M, Mizuno S, Yoshise A (1989) A primal-dual interior point algorithm for linear programming. In: Megiddo N (ed) Progress in Mathematical Programming. Springer, Berlin, 29–47
11. Lustig IJ, Marsten RE, Shanno DF (1991) Computational experience with a primal-dual interior point method for linear programming. Linear Alg Appl 152:191–222
12. Mascarenhas WF (1997) The affine scaling algorithm fails for stepsize 0.999. SIAM J Optim 7:34–46
13. Megiddo N (1989) Pathways to the optimal set in linear programming. In: Megiddo N (ed) Progress in Mathematical Programming. Springer, Berlin, pp 131–158
14. Mehrotra S (1992) On the implementation of a primal-dual interior point method. SIAM J Optim 2:575–601
15. Monteiro RC, Adler I (1989) Interior path following primal-dual algorithms. Part I: linear programming. Math Program 44:27–41
16. Nesterov Y, Nemirovskii A (1994) Interior-point polynomial algorithms in convex programming. SIAM, Philadelphia
17. Renegar J (1988) A polynomial-time algorithm, based on Newton's method, for linear programming. Math Program 40:59–93
18. Roos C, Vial J-Ph (1990) Long steps with the logarithmic penalty barrier function in linear programming. In: Gabszewicz J, Richard J-F, Wolsey L (eds) Economic Decision Making: Games, Economics, and Optimization. Elsevier, Amsterdam, pp 433–441
19. Vaidya PM (1990) An algorithm for linear programming which requires O(((m+n)n2 + (m+n)1.5n)L) arithmetic operations. Math Program 47:175–201
20. Vandenberghe L, Boyd S (1996) Semidefinite programming. SIAM Rev 38:49–95
21. Vanderbei RJ, Meketon MJ, Freedman BA (1986) A modification of Karmarkar's linear programming algorithm. Algorithmica 1:395–407
22. Xu X, Hung P-F, Ye Y (1996) A simplified homogeneous self-dual linear programming algorithm and its implementation. Ann Oper Res 62:151–171
23. Ye Y (1991) An O(n3L) potential reduction algorithm for linear programming. Math Program 50:239–258
24. Ye Y, Todd MJ, Mizuno S (1994) An $O(nL)O(\sqrt{n}L)$-iteration homogeneous and self-dual linear programming algorithm. Math Oper Res 19:53–67
25. Zhang Y (1994) On the convergence of a class of infeasible interior-point algorithms for the horizontal linear complementarity problem. SIAM J Optim 4:208–227

Linear Programming: Karmarkar Projective Algorithm

Karmarkar Algorithm

KURT M. ANSTREICHER
University Iowa, Iowa City, USA

MSC2000: 90C05

Article Outline

Keywords

Linear programming; Interior point methods; Projective transformation; Potential function; Polynomial time algorithm

In his groundbreaking paper [6], N.K. Karmarkar described a new *interior point method* for linear programming (LP). As originally described by Karmarkar, his algorithm applies to a LP problem of the form:

$$\text{KLP} \quad \begin{cases} \min & c^\top x \\ \text{s.t.} & Ax = 0 \\ & x \in S, \end{cases}$$

where $x \in \mathbf{R}^n$, A is an $m \times n$ matrix, and S is the simplex $S = \{x \in \mathbf{R}^n : x \geq 0,\ e^\top x = n\}$. Throughout e denotes the vector with each component equal to one. It is assumed that e is feasible in KLP, and that the optimal objective value in KLP is exactly zero. These assumptions may seem restrictive, but it is easy to show that a standard form LP problem:

$$\begin{cases} \min & c^\top x \\ \text{s.t.} & Ax = b \\ & x \geq 0, \end{cases} \tag{1}$$

can be converted into a problem of the form KLP by combining the problem with its dual, and minimizing the gap between the two problems.

In addition to the special form of the LP problem, Karmarkar employed two novel ingredients in the specification of his algorithm. The first was the use of a *projective transformation* in the construction of the algorithm's iterative process. The algorithm is initialized at $x^0 = e$. For an iterate $x^k > 0$, $k \geq 0$, let X^k be the diagonal matrix with $X^k_{ii} = x^k_i$, $i = 1, \ldots, n$. On the kth iteration, the algorithm uses a projective change of coordinates $T^k: S \to S$,

$$T_k(x) = \frac{n(X^k)^{-1}x}{e^\top (X^k)^{-1}x},$$

to map the point x^k to e. Under the assumption that the optimal value in KLP is zero, KLP is equivalent to the transformed problem:

$$\begin{cases} \min & \overline{c}^\top \overline{x} \\ \text{s.t.} & \overline{A}\overline{x} = 0 \\ & \overline{x} \in S, \end{cases}$$

where $\overline{x} = T^k(x)$, $\overline{c} = X^k c$ and $\overline{A} = AX^k$. The algorithm then takes a projected gradient step in the transformed problem, and uses the inverse projective transformation to define the next iterate in the original coordinates:

$$x^{k+1} = T_k^{-1}\left(e - \alpha \frac{\overline{c}_p}{\|\overline{c}_p\|}\right), \tag{2}$$

where α is a positive steplength and $\overline{c}_p$ is the projection of $\overline{c}$ onto the nullspace of $\overline{A}$ and $e^\top$.

Karmarkar's second innovation was the use of a *potential function* to monitor the algorithm's progress. Karmarkar's potential function is:

$$f(x) = n \ln(c^\top x) - \sum_{i=1}^{n} \ln(x_i).$$

Karmarkar proved that on each iteration, the steplength α in (2) can be chosen so that $f(\cdot)$ is reduced by an absolute constant δ. It is then easy to show that the iterates satisfy $c^\intercal x^k \leq e^{-k\delta/n} c^\intercal x^0$ for all k. For any positive L, it follows that if $c^\intercal x^0 \leq 2^{O(L)}$, then the algorithm obtains an iterate x^k having $c^\intercal x^k \leq 2^{-O(L)}$ in $k = O(nL)$ iterations, each requiring $O(n^3)$ operations. For a problem of the form KLP with integer data, it can be shown that if $c^\intercal x^k \leq 2^{-O(L)}$, where L is the number of bits required to represent the problem, then an exact optimal solution can be obtained from x^k via a 'rounding' procedure. These facts together imply that Karmarkar's algorithm is a *polynomial time algorithm* for linear programming, requiring $O(n^4L)$ operations for a problem with n variables, and integer data of bit size L. Karmarkar also described a *partial updating* technique that reduces the total complexity of his algorithm to $O(n^{3.5}L)$ operations. Partial updating is based on using a scaling matrix $\widetilde{X}^k$ which is an approximation of X^k, and only 'updating' components $\widetilde{X}^k_{ii}$ which differ from X^k_{ii} by more than a fixed factor.

Karmarkar's algorithm created a great deal of interest for two reasons. First, the algorithm was a polynomial time method for LP. Second, Karmarkar claimed that unlike the *ellipsoid algorithm*, the other well-known polynomial time method for LP, his method performed extremely well in practice. There was some controversy at the time regarding these claims, and eventually it was discovered that most of Karmarkar's computational results were based on the *affine scaling algorithm*, a simplified version of his algorithm that avoids the use of projective transformations. In any case it soon became clear that the performance of interior point algorithms for LP could be highly competitive with the *simplex method*, the usual solution technique, on large problems.

There is a great deal of research connected with Karmarkar's algorithm. Several authors ([1,3,4,5,9]) showed that the special form of KLP was unnecessary, and instead the projective algorithm could be directly applied to a standard form problem (1). This 'standard form variant' adds logic which maintains a lower bound on the unknown optimal value in (1). Later it was shown that the projective transformations could also be eliminated, giving rise to so-called *potential reduction algorithms* for LP. The best known potential reduction algorithm, due to Y. Ye [8], requires only $O(\sqrt{n}L)$ iterations, and with an adaptation of Karmarkar's partial updating technique has a total complexity of $O(n^3L)$ operations. The survey articles [2] and [7] give extensive references to research connected with Karmarkar's algorithm, and related potential reduction methods.

See also

- ▶ Entropy Optimization: Interior Point Methods
- ▶ Homogeneous Selfdual Methods for Linear Programming
- ▶ Interior Point Methods for Semidefinite Programming
- ▶ Linear Programming: Interior Point Methods
- ▶ Potential Reduction Methods for Linear Programming
- ▶ Sequential Quadratic Programming: Interior Point Methods for Distributed Optimal Control Problems
- ▶ Successive Quadratic Programming: Solution by Active Sets and Interior Point Methods

References

1. Anstreicher KM (1986) A monotonic projective algorithm for fractional linear programming. Algorithmica 1:483–498
2. Anstreicher KM (1996) Potential reduction algorithms. In: Terlaky T (ed) Interior point methods of mathematical programming. Kluwer, Dordrecht, pp 125–158
3. de Ghellinck G, Vial J-Ph (1986) A polynomial Newton method for linear programming. Algorithmica 1:425–453
4. Gay DM (1987) A variant of Karmarkar's linear programming algorithm for problems in standard form. Math Program 37:81–90
5. Gonzaga CC (1989) Conical projection algorithms for linear programming. Math Program 43:151–173
6. Karmarkar N (1984) A new polynomial-time algorithm for linear programming. Combinatorica 4:373–395
7. Todd MJ (1997) Potential-reduction methods in mathematical programming. Math Program 76:3–45
8. Ye Y (1991) An O(n3L) potential reduction algorithm for linear programming. Math Program 50:239–258
9. Ye Y, Kojima M (1987) Recovering optimal dual solutions in Karmarkar's polynomial algorithm for linear programming. Math Program 39:305–317

Linear Programming: Klee–Minty Examples

Konstantinos Paparrizos[1], Nikolaos Samaras[1], Dimitrios Zissopoulos[2]

[1] Department Applied Informatics, University Macedonia, Thessaloniki, Greece

[2] Department Business Admin., Techn. Institute West Macedonia, Kozani, Greece

MSC2000: 90C05

Article Outline

Keywords

Klee–Minty examples; Linear programming; Simplex algorithm; Pivoting rules

The problem of determining the worst-case behavior of the simplex algorithm remained an outstanding open problem for more than two decades. In the beginning of the 1970s, V. Klee and G.J. Minty [9] solved this problem by constructing linear examples on which an exponential number of iterations is required before optimality occurs. In this article we present the Klee–Minty examples and show how they can be used to show exponential worst-case behavior for some well known *pivoting rules*.

Introduction

The problem of determining the worst-case behavior of the simplex algorithm remained an outstanding open problem for more than two decades. In the beginning of the 1970s, Klee and Minty in their classical paper [9] showed that the most commonly used pivoting rule, i. e., *Dantzig's largest coefficient pivoting rule* [5], performs exponentially bad on some specially constructed linear problems, known today as Klee–Minty examples. Later on, R.G. Jeroslow [8] showed similar behavior for the maximum improvement pivoting rule. He showed this result by slightly modifying Klee–Minty examples.

The Klee–Minty examples have been used by several researchers to show exponential worst-case behavior for the great majority of the practical pivoting rules. D. Avis and V. Shvatal [1] and independently, K.G. Murty [10, p. 439] showed exponential behavior for *Bland's least index pivoting rule* [2] and D. Goldfarb and W. Sit [7] for the *steepest edge simplex method* [5]. Recently,

C. Roos [13] established exponential behavior for *Terlaky's criss-cross method* [14] and K. Paparrizos [11] for a number of pivoting rules some of which use past history. Similar results have been derived by Paparrizos [12] for his *dual exterior point algorithm* and K. Dosios and Paparrizos [6] for a new primal dual pivoting rule [3].

In this paper we present the Klee–Minty examples and show some of their properties that are used in deriving complexity results of the simplex algorithm. These properties are then used to show exponential behavior for two pivoting rules; the least index and the maximum coefficient pivoting rule.

The paper is self contained. Next section describes a particular form of the *simplex algorithm*. The Klee–Minty examples and their properties are presented in Section 3. Section 4 is devoted to complexity results.

Simplex Algorithm

In describing our results we find it convenient to use the dictionary form [4] of the simplex algorithm. We will see in the next section that this form exhibits some advantages in describing the properties of the Klee–Minty examples.

Consider the linear problem in standard form

$$\begin{cases} \max & z = c^\top x \\ \text{s.t.} & Ax = b, \\ & x \geq 0, \end{cases} \tag{1}$$

where $c, x \in \mathbf{R}^n$, $b \in \mathbf{R}^m$, $A \in \mathbf{R}^{m\times n}$ and superscript T denotes transposition. Without loss of generality we may assume that A is of full row rank, i. e., $\operatorname{rank}(A) = m$ ($m < n$).

A basis for problem (1) is a set of indices $B \subset \{1, \ldots, n\}$ containing exactly m indices. The element of B, the components of c and x and the columns of A indexed by B are called *basic* while the remaining ones are called *nonbasic*. The set of nonbasic indices is denoted by N, $N = \{1, \ldots, n\} \sim B$. We also denote by $B(N)$ the submatrix of A containing the columns indexed by $B(N)$. The components of a vector x indexed by $B(N)$ are denoted by $x_B(x_N)$.

With this notation at hand the equality constraints of (1) are written in the form

$$Bx_B + Nx_N = b. \tag{2}$$

If B is a nonsingular matrix we can set $x_N = 0$ and compute x_B from (2). Then, we find $x_B = B^{-1}b$. The non singular matrix B is called *basic matrix* or *basis*. The solution $x_N = 0$ and $x_B = B^{-1}b$ is called *basic solution*. If, in addition, it is $x_B = B^{-1}b \geq 0$, then x_B, x_N is a *basic feasible solution*. Geometrically, a basic feasible solution of (1) corresponds to a vertex of the polyhedral set of the feasible region.

If B is nonsingular, we can express the basic variables x_B as a function of the non basic variable x_N. We have from (2) that

$$x_B = -B^{-1}Nx_N + B^{-1}b. \tag{3}$$

Using (3), the objective function of problem (1) is written in the form

$$\begin{aligned} z &= c_B^\top x_B + c_N^\top x_N \\ &= c_B^\top(-B^{-1}Nx_N + B^{-1}b) + c_N^\top x_N \\ &= (-c_B^\top B^{-1}N + c_N^\top)x_N + c_B^\top B^{-1}b. \end{aligned} \tag{4}$$

At every iteration the simplex algorithm constructs the system of equations (3) and (4).

Let the current feasible basis be B. The corresponding system of equations is written in the form

$$\begin{aligned} z &= (-c_B^\top B^{-1}N + c_N^\top)x_N + c_B^\top B^{-1}b, \\ x_B &= -B^{-1}Nx_N + B^{-1}b. \end{aligned} \tag{5}$$

We denote the coefficients of x_N and the constant terms of (5) by H, i. e.,

$$\begin{pmatrix} c_N^\top - c_B^\top B^{-1}N & c_B^\top B^{-1}b \\ -B^{-1}N & B^{-1}b \end{pmatrix} = H.$$

The top row of H, row zero, is devoted to the objective function. Some times we call it *cost row*. The remaining rows are numbered $1, \ldots, m$. The ith row, $1 \leq i \leq m$, corresponds to the basic variable $x_{B[i]}$, where $B[i]$ denotes the ith element of B. Similarly, the jth column of H, $1 \leq j \leq n-m$, corresponds to the nonbasic variable $x_{N[j]}$. The last column of H corresponds to the constant terms. We denote the entries of H by h_{ij}.

It is well known that if $h_{0j} \leq 0$, for $j = 1, \ldots, n-m$, then x_B, x_N is an optimal solution to (1). In this case the algorithm terminates. Otherwise, a nonbasic variable $x_{N[q]} = x_l$ such that $h_{0,N[q]} > 0$ is chosen. Variable x_l is called *entering variable*. If the condition

$$h_{i,N[q]} \geq 0, \quad \text{for } i = 1, \ldots, m,$$

holds, problem (1) is unbounded and the algorithm stops. Otherwise, the basic variable $x_{B[p]} = x_k$, is determined by the following *minimum ratio test*

$$\frac{x_{B[p]}}{-h_{r,N[q]}} = \min\left\{\frac{h_{i,n-m+1}}{-h_{i,N[q]}}:\ 1 \le i \le m, h_{i,N[q]} < 0\right\}.$$

The basic variable x_k is called *leaving variable*. Then, the entering variable x_l takes the place of the leaving variable and vice versa, i. e., it is set

$$B[p] \leftarrow N[q] \quad \text{and} \quad N[q] \leftarrow B[p].$$

Thus, a new basis $\overline{B}$ is constructed and the procedure is repeated. Let $\overline{H}$ be the tableau corresponding to the new basis $\overline{B}$. It is easily seen that

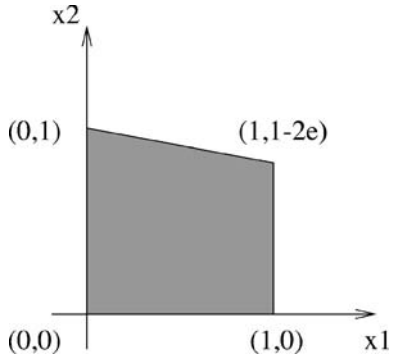
$$\overline{h}_{ij} = \begin{cases} -\frac{h_{pj}}{h_{pq}} & \text{if } i = p, \\ \frac{h_{iq}}{h_{pq}} & \text{if } i \ne p, j = q, \\ h_{ij}\frac{h_{pj}}{h_{pq}} & \text{otherwise.} \end{cases} \tag{6}$$

Klee–Minty Examples

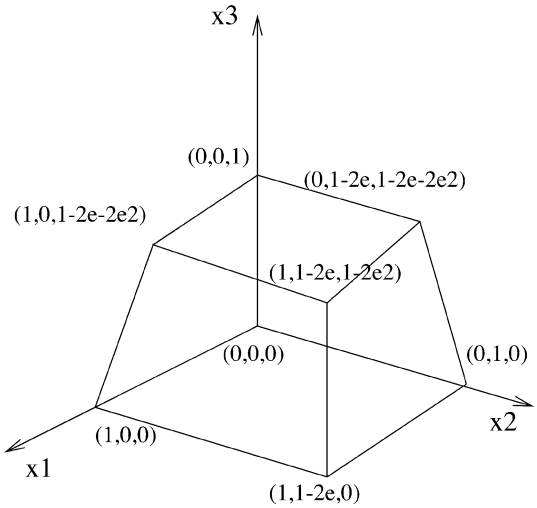
The Klee–Minty examples of order n are the linear problems of the form

$$\begin{cases} \max & \sum_{j=1}^{n} \varepsilon^{n-j} x_j \\ \text{s.t.} & x_1 \le 1 \\ & 2\sum_{j=1}^{i-1} \varepsilon^{i-j} x_j + x_i \le 1, \\ & \quad i = 2, \ldots, n, \\ & x_j \ge 0, \quad j = 1, \ldots, n, \end{cases} \tag{7}$$

where $0 < \varepsilon \le 1/3$. In this section we will show that the feasible region of (7) is a slightly perturbed cube of dimension n, see Fig. 1 and Fig. 2. The optimal solution is $(0, 0, \ldots, 1) \in \mathbf{R}^n$ and the optimal value is 1. A cube of dimension n has 2^n vertices. In the next section we will describe pivoting rules that force the simplex method to pass through all the vertices of the Klee–Minty examples. These pivoting rules require $2^n - 1$ iterations before optimality is reached and, hence, they are exponential.



Linear Programming: Klee–Minty Examples, Figure 1
Feasible region of Klee–Minty example of order $n = 2$



Linear Programming: Klee–Minty Examples, Figure 2
Feasible region of Klee–Minty example of order $n = 3$

The standard form of linear problem (1) is

$$\begin{cases} \max & \sum_{j=1}^{n} \varepsilon^{n-j} x_j \\ \text{s.t.} & x_1 + x_{n+1} = 1, \\ & 2\sum_{j=1}^{i-1} \varepsilon^{i-j} x_j + x_{n+i} = 1, \\ & \quad i = 2, \ldots, n, \\ & x_j \ge 0, \quad j = 1, \ldots, 2n, \end{cases} \tag{8}$$

where x_{n+i}, $1 \le i \le n$, is the slack variable corresponding to the ith inequality constraint of problem (7).

We will be interested in basic solutions of (8) such that, for each $j = 1, \ldots, n$ either x_j or x_{n+j} is basic but

not both. Such a basic solution is called *distinguished*. A tableau corresponding to a dis- tinguished basis is called *distinguished tableau*. In order to facilitate the presentation it is convenient to introduce the set Q of all zero-one n-sequences $(a_1, \ldots, a_n)$ such that

$$a_j = \begin{cases} 0 & \text{if } x_j \text{ is nonbasic and } x_{n+j} \text{ is basic,} \\ 1 & \text{if } x_{n+j} \text{ is nonbasic and } x_j \text{ is basic.} \end{cases}$$

We denote the distinguished basis corresponding to the sequence $(0, \ldots, 0)$ by $\widehat{B}$ and the tableau corresponding to $\widehat{B}$ by $\widehat{H}$. We have $\widehat{B} = \{n+1, \ldots, 2n\}$. It is easily verified that

$$\widehat{h}_{ij} = \begin{cases} \varepsilon^{n-j} & \text{if } i = 0,\ j \le n, \\ -1 & \text{if } 1 \le i = j \le n, \\ 0 & \text{if } 1 \le i < j,\ j \ne n+1, \\ -2\varepsilon^{i-j} & \text{if } i > j, \\ 1 & \text{if } i \ge n \text{ and } j = n+1. \end{cases} \tag{9}$$

Tableau $\widehat{H}$ is sometimes called *initial*.

A distinguished tableau H corresponding to $(a_1, \ldots, a_n) \in Q$ is constructed by starting from $\widehat{H}$ and pivoting only on elements h_{pp} such that $a_p = 1$. Using this procedure and relations (6) and (9) we easily conclude that

$$\begin{aligned} &h_{pp} = -1 \text{ for } p = 1, \ldots, n, \\ &h_{ij} = 0 \text{ for } 1 \le i \le n-1,\ i \le j \le n, \end{aligned} \tag{10}$$

for each distinguished tableau H.

Lemma 1 *Let B be an arbitrary distinguished basis and H the corresponding tableau. Then*

$$h_{ij} + h_{pj}h_{ip} = -h_{ij},\ j < p < n,\ i \ge 2, \tag{11}$$

$$h_{0j} + h_{pj}h_{0j} = 0,\ j < p \le n,\ i = 0. \tag{12}$$

Proof It suffices to show the following induction hypothesis. If the distinguished tableau $\overline{H}$ satisfies (11) and (12) and a pivot operation is performed on h_{rr}, $1 \le r \le n$, resulting in tableau H, then H satisfies (11) and (12) as well. Observe that relations (11) and (12) are satisfied by the initial tableau $\widehat{H}$.

So, assume that H satisfies (11) and (12) and a pivot operation is performed on element $h_{rr} = -1$. Then, we have from (6) and (10)

$$h_{ij} = \begin{cases} \overline{h}_{ij} + \overline{h}_{rj}\overline{h}_{ir}, & i \ne r,\ j \ne r, \\ -\overline{h}_{ij}, & i \ne r,\ j = r, \\ \overline{h}_{ij}, & \text{otherwise.} \end{cases} \tag{13}$$

Combining (13) and the induction hypothesis we have

$$h_{ij} = \begin{cases} -\overline{h}_{ij} & \text{if } i = 0,\ j \le r, \\ -\overline{h}_{ij} & \text{if } i > r,\ j \le r, \\ -\overline{h}_{ij} + \overline{h}_{rj}\overline{h}_{ir} & \text{if } i > r,\ j = n+1, \\ -\overline{h}_{ii} + \overline{h}_{rj}\overline{h}_{ir} & \text{if } i = 0,\ j = n+1, \\ -\overline{h}_{ij}, & \text{otherwise.} \end{cases} \tag{14}$$

There are two cases to be considered, $p \le r$ and $p > r$. From relations (14) we have, for $p \le r$,

$$h_{ij} = -\overline{h}_{ij}, \quad h_{ip} = -\overline{h}_{ip}, \quad h_{pj} = \overline{h}_{pj}$$

and for $p > r$

$$h_{ij} = -\overline{h}_{ij}, \quad h_{ip} = \overline{h}_{ij}, \quad h_{pj} = -\overline{h}_{pj}.$$

In both cases,

$$\begin{aligned} h_{ij} + h_{pj}h_{ip} &= -\overline{h}_{ij} - \overline{h}h_{ip}h_{pj} \\ &= \overline{h}_{ij} = -h_{ij}. \end{aligned}$$

This proves (11). The proof of (12) is similar.

Lemma 1 shows that pivoting on element h_{pp} of a distinguished tableau H is very easily performed. Just change the signs of the entries h_{ij} such that $i = p$ and $j \le p$ or $i > p$ and $j \le p$ and set

$$h_{i,n+1} \leftarrow h_{i,n+1} + h_{ip}h_{p,n+1}$$

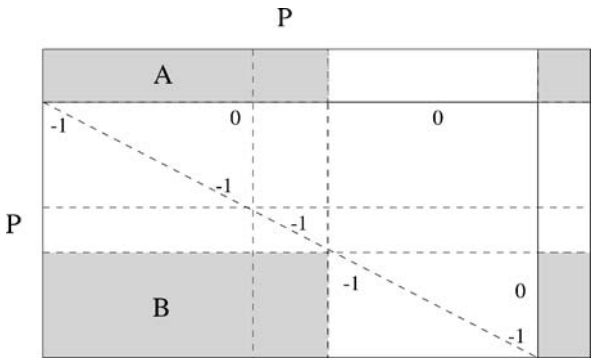
for $i = 0$ or $i > p$.

Figure 3 illustrates the entries of H that change value when pivoting on h_{pp}. In particular, the entries in areas A and B just change sign.

Theorem 2 *Let H be a distinguished tableau of problem (8) and $a = (a_1, \ldots, a_n)$ be the corresponding n-sequence. Then the following relations hold.*

For $i = 1, \ldots, n$ and $j = 1, \ldots, n$ we have

$$h_{ij} = \begin{cases} -1, & i = j, \\ 0, & i < j, \end{cases} \tag{15}$$



Linear Programming: Klee–Minty Examples, Figure 3
Entries of a distinguished tableau *H* that change value after pivoting on element h_{pp}

while for $i > j$,

$$h_{ij} = \begin{cases} -2\varepsilon^{i-j}, & \sum_{k=j}^{i-1} a_k \text{ even}, \\ 2\varepsilon^{i-j}, & \sum_{k=j}^{i-1} a_k \text{ odd}. \end{cases} \tag{16}$$

For $i = 0$ and $j = 1, \ldots, n$ we have

$$h_{0j} = \begin{cases} \varepsilon^{n-j}, & \sum_{k=j}^{n} a_k \text{ even}, \\ -\varepsilon^{n-j}, & \sum_{k=j}^{n} a_k \text{ odd}. \end{cases} \tag{17}$$

For $i = 1, \ldots, n$ and $j = n + 1$ we have

$$h_{i,n+1} = \begin{cases} 1, & i = 1, \\ 1 - \sum_{k=1}^{i-1} a_k h_{ik}, & 2 \le i \le n. \end{cases} \tag{18}$$

Proof The proof is by induction. We assume that distinguished tableau $\overline{H}$ satisfies (15)–(18), and show that tableau H computed by pivoting on $\overline{h}_{pp}$ satisfies (15)–(18) as well. Observe that initial tableau $\widehat{H}$ satisfies (15)–(18).

Let $\overline{a} = (\overline{a}_1, \ldots, \overline{a}_n)$ be the sequence corresponding to tableau $\overline{H}$. Then

$$\overline{a}_j = \begin{cases} a_j, & j \ne p, \\ 1 - a_j, & j = p. \end{cases}$$

Proof of (15)–(16)}. Relations (15) have already been shown. From Lemma 1 we have

$$\overline{h}_{ij} = h_{ij} \quad \text{and} \quad \sum_{k=j}^{i-1} a_k = \sum_{k=j}^{i-1} \overline{a}_k$$

for $i \le p$ or $i > p$ and $j > p$. For $i > p$ and $j \le p$ we have

$$\sum_{k=j}^{i-1} \overline{a}_k = \sum_{k=j}^{i-1} a_k - 2a_p.$$

Hence, if $\sum_{k=j}^{i-1} \overline{a}$ is odd (even), $\sum_{k=j}^{i-1} \overline{a}_k$ is even (odd). Also, from Lemma 1 we have $\overline{h}_{ij} = -h_{ij}$. Hence, (16) holds in all cases.

Proof of (17). It is easily seen that

$$\text{sign}(h_{0j}) = \text{sign}(h_{mj}), \quad \text{for } i \le n.$$

Now the proof comes from the proof of (16).

Proof of (18). If $i \le p$ we have $\overline{h}_{i,n+1} = h_{i,n+1}$. Hence, (18) holds trivially from the induction hypothesis. If $i > p$, then

$$\begin{aligned} \overline{h}_{i,n+1} &= h_{i,n+1} + h_{p,n+1} h_{ip} \\ &= 1 - \sum_{k=1}^{i-1} a_k h_{ik} + \left(1 - \sum_{k=1}^{p-1} a_k h_{pk}\right) h_{ip} \\ &= 1 - \sum_{k=1}^{p-1} a_k (h_{ik} + h_{pk} h_{ip}) \\ &\quad + h_{ip} - a_k h_{pp} h_{ip} - \sum_{k=1}^{i-1} a_k h_{ik} \\ &= 1 - \sum_{k=1}^{p-1} \overline{a} \overline{h}_{ik} - (1 - a_p) \overline{h}_{ip} - \sum_{k=p+1}^{i-1} \overline{a}_k \overline{h}_{ik} \\ &= 1 - \sum_{k=1}^{i-1} \overline{a}_k \overline{h}_{ik}. \end{aligned}$$

Theorem 3 *The feasible region of problem (8) is a slightly perturbed cube.*

Proof It suffices to show that the feasible region has precisely 2^n vertices. We show that each distinguished tableau, H, is feasible and all the adjacent tableaux are distinguished.

From Theorem 2 we have

$$\begin{aligned} h_{i,n+1} &= 1 - \sum_{k=1}^{i-1} a_k h_{ik} \\ &> 1 - 2\varepsilon(1 + \varepsilon + \varepsilon^2 + \cdots) \\ &= 1 - \frac{2\varepsilon}{1-\varepsilon} \geq 0. \end{aligned}$$

We show that if $x_{N[p]}$ is entering, then $x_{B[p]}$ is leaving variable. Let $h_{ip} < 0$ $(i < p)$. We show that

$$\frac{h_{p,n+1}}{h_{pp}} \leq \frac{h_{i,n+1}}{-h_{ip}}.$$

The last relation is equivalently written

$$-h_{ip}(1 - \sum_{k=1}^{p-1} a_k h_{pk}) < 1 - \sum_{k=1}^{i-1} a_k h_{ik}.$$

Using Lemma 1 we get

$$-h_{ip} - 2\sum_{k=1}^{p-1} a_k h_{ik} < 1 - \sum_{k=1}^{i-1} a_k h_{ik}$$

or

$$\begin{aligned} 0 &< 1 + h_{ip} + \sum_{k=1}^{p-1} a_k h_{ik} - \sum_{k=p}^{i-1} a_k h_{ik} \\ &= 1 + \sum_{k=1}^{p-1} a_k h_{ik} + (1 - a_k) h_{ip} + \sum_{k=p+1}^{i-1} a_k h_{ik}. \end{aligned}$$

We have already shown that the last relation holds.

Applications

Now, we are ready to show exponential behavior for some pivoting rules. Let $a \neq b$ be sequences of Q. We write $a < b$ if for the largest index j such that $a_j \neq b_j$ it is $\sum_{j=1}^{n} a_k$ even and $\sum_{k=1}^{n} b_k$ odd.

Let now $f(a)$ be the objective value at the vertex corresponding to $a \in Q$. It is easily seen that $f(a) < f(b)$, if $a < b$. The immediate successor of a sequence $a \in Q$ is the sequence $(a_1, \ldots, a_r, 1 - a_p, a_{p+1}, \ldots, a_n)$, where p is the smallest index such that $\sum_{j=p}^{n} a_j$ is even.

Given a distinguished tableau H, a nonbasic variable $x_{N[q]}$ is called *eligible* if $h_{0,\,N[p]} > 0$.

A pivoting rule that forces the simplex algorithm to pass through all vertices of Klee–Minty examples is the following. For the ease of reference we call it *generic pivoting rule*. Let $a \in Q$ be the sequence corresponding to H. The entering variable is $x_{N[p]}$, where p is the smallest index such that $\sum_{k=p}^{n} a_k$ is even. From Theorem 2 we see that $h_{0,\,N[p]} = \varepsilon^{n-p} > 0$. Hence, $x_{N[p]}$ is eligible, and the generic pivoting rule requires $2^n - 1$ iterations on Klee–Minty examples of order n.

Smallest Index Rule

In the *smallest index rule*, the entering variable is the eligible variable with the smallest index. We show that the smallest index rule, called also *Bland's rule*, performs exponentially on the slightly modified Klee–Minty examples

$$\begin{cases} \max & \displaystyle\sum_{j=1}^{n} \varepsilon^{n-j} x_{2j-1} \\ \text{s.t.} & x_1 \leq 1 \\ & \displaystyle 2\sum_{j=1}^{i-1} \varepsilon^{i-j} x_{2j-1} + x_{2i-1} \leq 1, \\ & \quad i = 2, \ldots, n, \\ & x_j \geq 0, \quad j = 1, \ldots, n. \end{cases} \tag{19}$$

We introduce the slack variable x_{2i} to the ith constraint of problem (19).

Theorem 4 *The least index pivoting rule performs exponentially on example (19).*

Proof We show that the simplex algorithm employing the least index pivoting rule requires $2^n - 1$ iterations when applied to problem (19) and initialized with the basis corresponding to the sequence $(0, \ldots, 0) \in Q$. Clearly, all the bases generated by the algorithm are distinguished i. e. for each i either x_{2i-1} or x_{2i} is basic but not both. Let H be the current distinguished tableau corresponding to the sequence $a \in Q$. Let also p be the smallest index such that $\sum_{k=p}^{n} a_k$ is even.

Then $h_{0,\,N[p]} > 0$ and, hence, $x_{N[p]}$ is eligible. Because of the indexing of the variable in problem (19), $N[p] = 2p$ or $2p - 1$. If q is another index such that $h_{0,\,N[q]} > 0$ ($\sum_{k=q}^{n} a_k$ is even), then $q > p$ and, hence, $N[q] > N[p]$. Hence, the next basis corresponds to the immediate successor of $a \in Q$. This completes the proof.

Largest Coefficient Rule

In the *largest coefficient rule* the entering variable $x_{N[p]}$ is chosen so that

$$h_{0,N[p]} = \max\left\{h_{0,N[j]}:\ h_{0,N[j]} > 0\right\}.$$

This rule solves problem (7) in one iteration when the initial basis is $(0,\ldots,0) \in Q$.

We modify problem (7) as follows. We set

$$\begin{aligned} \varepsilon &= \frac{1}{\mu}, \\ x_j &= y_j e^{2(j-1)}, \end{aligned} \tag{20}$$

and divide the ith constraint by $\varepsilon^{2(i-1)}$ and the objective function by $\varepsilon^{2(n-1)}$. Then, problem (7) is written in the equivalent form

$$\begin{cases} \max & \displaystyle\sum_{j=1}^{n} \mu^{n-j} y_j \\ \text{s.t.} & \displaystyle 2\sum_{j=1}^{i-1} \mu^{i-j} y_j + y_i \le \mu^{2(i-1)}, \\ & \quad i = 1,\ldots,n, \\ & y_j \ge 0, \quad j = 1,\ldots,n, \end{cases} \tag{21}$$

where $\mu = 1/\epsilon \ge 3$.

Theorem 5 *The largest coefficient rule performs exponentially on problem (21).*

Proof Problem (21) is a scaled version of problem (7). Let x_{n+i} be the slack of constraint i. Then, all the results of the previous section, except those involving the RHS, hold true for problem (21). Because of relation (20), $c_j \ge 0$ if and only if $y_j \ge 0$. Hence, every distinguished basis of (21) is feasible. Now, it suffices to show that the generic and the largest coefficient rule coincide when applied to problem (21). However, this statement holds because $\mu > 1$.

See also

- ▶ Criss-cross Pivoting Rules
- ▶ Least-index Anticycling Rules
- ▶ Lexicographic Pivoting Rules
- ▶ Linear Programming

References

1. Avis D, Chvátal V (1978) Notes on Bland's pivoting rule. Math Program Stud 8:24–34
2. Bland RG (1977) New finite pivoting rules for the simplex method. Math Oper Res 2:103–107
3. Chen H, Pardalos PM, Saunders MA (1994) The simplex algorithm with a new primal and dual pivot rule. Oper Res Lett 16:121–127
4. Chvátal V (1983) Linear programming. Freeman, New York
5. Dantzig GB (1963) Linear programming and extentions. Princeton Univ. Press, Princeton
6. Dosios K, Paparrizos K (1996) Resolution of the problem of degeneracy in a primal and dual simplex algorithm. Oper Res Lett 20:45–50
7. Goldfarb D, Sit W (1979) Worst case behavior of the steepest edge simplex method. Discrete Appl Math 1:277–285
8. Jeroslow RG (1973) The simplex algorithm with the pivot rule of maximizing improvement criterion. Discret Math 4:367–377
9. Klee V, Minty GJ (1972) How good is the simplex algorithm? In: Shisha O (ed) Inequalities: III. Acad. Press, New York
10. Murty KG (1983) Linear programming. Wiley, New York
11. Paparrizos K (1989) Pivoting rules directing the simplex method through all feasible vertices of Klee–Minty examples. Opsearch 26(2):77–95
12. Paparrizos K (1993) An exterior point simplex algorithm for (general) linear programming problems. Ann Oper Res 47:497–508
13. Roos C (1990) An exponential example for Terlaky's pivoting rule for the criss - cross simplex method. Math Program 46:78–94
14. Terlaky T (1985) A convergent criss - cross method. Math Oper Statist Ser Optim 16:683–690

Linear Programming Models for Classification

Paul A. Rubin
The Eli Broad Graduate School of Management, Michigan State University, East Lansing, USA

MSC2000: 62H30, 68T10, 90C05

Article Outline

See also
References

Keywords

Classification; Discriminant analysis; Linear programming

Introduction

The *G-group classification problem (discriminant problem)* seeks to classify members of some population into one of G predefined groups based on the value of a *scoring function* f applied to a vector $\mathbf{x} \in \Re^p$ of observed attributes. The scoring function is constructed using *training samples* drawn from each group. Of several criteria available for selecting a scoring function, expected accuracy (measured either in terms of frequency of misclassification or average cost of misclassification) predominates. The scoring function f can be vector-valued, but when two groups are involved it is almost always scalar-valued, and scalar functions may be used even when there are more than two groups.

As discussed in [8], statistical methods for constructing scoring functions revolve around estimating, directly or indirectly, the density functions of the distributions of the various groups. In contrast, a number of approaches have been proposed that in essence ignore the underlying distributions and simply try to classify the training samples with maximal accuracy, hoping that this accuracy carries over to the larger population. The use of mathematical programming was suggested at least as early as 1965 by Mangasarian [11]; interest in it grew considerably with the publication of a pair of papers by Freed and Glover in 1981 [3,4], which led to parallel streams of research in algorithm development and algorithm analysis.

Though nonlinear scoring functions can be constructed, virtually all research into mathematical programming methods other than support vector machines [1] restricts attention to linear functions. This is motivated largely by tractability of the mathematical programming problems, but is bolstered by the fact that the Fisher linear discriminant function, the seminal statistically derived scoring function, is regarded as a good choice under a wide range of conditions. For the remainder of this article, we assume f to be linear. Directly maximizing accuracy on the training samples dictates the use of a mixed integer program to choose the scoring function (► Mixed Integer Classification Problems). The number of binary variables in such a formulation is proportional to the size of the training samples, and so computation time grows in a nonpolynomial manner as the sample sizes increase. It is therefore natural that attention turned to more computationally efficient linear programming classification models (LPCMs). Erenguc and Koehler [2] give a thorough survey of the spectrum of mathematical programming classification models as of 1989, and Stam [14] provides a somewhat more recent view of the field. Comparisons, using both "real-world" data and Monte Carlo experiments, of the accuracy of scoring functions produced by mathematical programming models with that of statistically-derived functions has produced mixed results [14], but there is evidence that LPCMs are more robust than statistical methods to large departures from normality in the population (such as populations with mixture distributions, discrete attributes, and outlier contamination).

Models

When $G = 2$ and f is linear and scalar-valued, classification of $\mathbf{x}$ is based without loss of generality on whether $f(\mathbf{x}) < 0$ or $f(\mathbf{x}) > 0$. (If $f(\mathbf{x}) = 0, \mathbf{x}$ can be assigned to either group with equal plausibility. This should be treated as a classification failure.) Barring the degenerate case $f \equiv 0$, the solution set to $f(\mathbf{x}) = 0$ forms a *separating hyperplane*. Ideally, though not often in practice, each group resides within one of the half-spaces defined by that hyperplane. An early precursor to linear programming models, the *perceptron algorithm* [12], constructs an appropriate linear classifier in finite time when the samples are separable, but can fail if the samples are not separable.

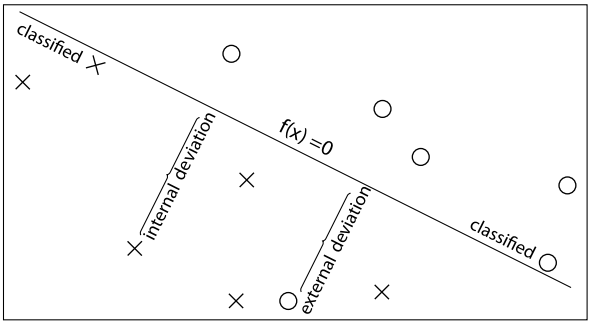
There being no way to count misclassifications in an optimization model without introducing integer variables, LPCMs must employ a surrogate criterion. A variety of criteria have been tried, all revolving around measurements of the displacement of the sample points from the separating hyperplane. Let $f(\mathbf{x}) = \mathbf{w}'\mathbf{x} + w_0$ for some non-null coefficient vector $\mathbf{w} \in \Re^p$ and some scalar w_0. The euclidean distance from $\mathbf{x}$ to the separating hyperplane is easily shown to be $|f(\mathbf{x})| \,/\, \|\mathbf{w}\|$. So the value of the scoring function at each training observa-

tion measures (to within a scalar multiple) how far the observation falls from the separating hyperplane. That distance is in turn identified as either an *internal deviation* or an *external deviation* depending on whether the observation falls in the correct or incorrect half-space. Figure 1 illustrates both types of deviation.

The "hybrid" model of Glover et al. [6] is sufficiently flexible to capture the key features of most two-group models. Let $\mathbf{X}_g$ be an $N_g \times p$ matrix of training observations from group g, and let $\mathbf{0}$ and $\mathbf{1}$ denote vectors of appropriate dimension, all of whose components are 0 and 1 respectively. The core of the hybrid model, to be expanded later, is:

$$\begin{aligned} \min & \sum_{g=1}^{2} \left(\alpha_g \cdot \mathbf{1}'\mathbf{e}_g - \beta_g \cdot \mathbf{1}'\mathbf{d}_g + \gamma_g e_{g0} - \delta_g d_{g0}\right) \\ \text{s.t. } & \mathbf{X}_1\mathbf{w} + w_0 \cdot \mathbf{1} + \mathbf{d}_1 - \mathbf{e}_1 + d_{10} \cdot \mathbf{1} - e_{10} \cdot \mathbf{1} \leq \mathbf{0} \\ & \mathbf{X}_2\mathbf{w} + w_0 \cdot \mathbf{1} - \mathbf{d}_2 + \mathbf{e}_2 - d_{20} \cdot \mathbf{1} + e_{20} \cdot \mathbf{1} \geq \mathbf{0} \\ & \mathbf{w}, w_0 \text{ free; } \mathbf{d}_g, \mathbf{e}_g, d_{g0}, e_{g0} \geq 0\,. \end{aligned}$$

Variables $\mathbf{d}_g$ and $\mathbf{e}_g$ are intended to capture the internal and external deviations respectively of individual observations from group g, while e_{g0} and d_{g0} are intended to capture the maximum (or minimum) external and internal deviations respectively across the sample from group g. (The original hybrid model had $d_{10} = d_{20}$ and $e_{10} = e_{20}$, which is unnecessarily restrictive.) The intent of Glover *et al.* in presenting the hybrid model was to subsume a number of previously proposed models, and so the hybrid model should be viewed as a framework. When applied, not all of the deviation variables need be present. For example, omission of $\mathbf{e}_g$ and $\mathbf{d}_g$ would yield a version of the "MMD" model [2], with e_{g0} the worst external deviation of any group g observation if any is misclassified (in which case $d_{g0} = 0$) and d_{g0} the minimum internal deviation of any group g observation if none is misclassified (in which case $e_{g0} = 0$). On the other hand, omission of e_{g0} and d_{g0} results in a variation of the "MSID" model [2], with the objective function penalizing individual external deviations ($\mathbf{e}_g$) and rewarding individual internal deviations ($\mathbf{d}_g$). The nonnegative objective coefficients α_g, β_g, γ_g, δ_g must be chosen so that the penalties for external deviations exceed the rewards for internal deviations; otherwise, the linear program becomes unbounded, as adding an equal amount to both e_{gn} and d_{gn} improves the objective value.



Linear Programming Models for Classification, Figure 1
Two-Group Problem with Linear Classifier

Pathologies

Due to their focus on minimizing error count, mixed integer classification models tend to be feasible (the trivial function $f \equiv 0$ is often a feasible solution) and bounded (one cannot do better than zero misclassifications). LPCMs, in contrast, tend to be "naturally" feasible but may require explicit bounding constraints. If the training samples are perfectly separable, a solution exists to the partial hybrid model with $e_{gn} = 0$ for all g and n and $d_{gn} > 0$ for some g and n; any positive scalar multiple of that solution is also feasible, and so the objective value is unbounded below. One way to correct this is to introduce bounds on the coefficients of the objective function, say

$$-\mathbf{1} \leq \mathbf{w} \leq +\mathbf{1}.$$

Another potential problem has to do with what is variously referred to as the "trivial" or "unacceptable" solution, namely $f \equiv 0$. Consider the partial hybrid model above. The trivial solution (all variables equal to zero) is certainly feasible, with objective value zero. Given the requirement that the objective coefficients of external deviation variables dominate those of internal deviation variables, any solution with a negative objective value must perfectly separate the training samples. Contrapositively, then, if the training samples cannot be separated, the objective value cannot be less than zero, in which case the trivial solution is in fact optimal. This is undesirable: the trivial function does not classify anything. The trick is to make the trivial solution suboptimal. Some authors try to accomplish this by fixing the

constant term w_0 of the classification function at some nonzero value (typically $w_0 = 1$). The trivial discriminant function $\mathbf{w} = \mathbf{0}$ with nonzero constant term now misclassifies one group completely, and is unlikely to be the model's optimal solution even when the training samples cannot be separated. There is the possibility, however, that the best linear classifier has $w_0 = 0$, in which case this approach dooms the model to finding an inferior solution.

Other approaches include various attempts to make $\mathbf{w} = \mathbf{0}$ infeasible, such as adding the constraint $\|\mathbf{w}\| = 1$. Unfortunately, trying to legislate the trivial solution out of existence results in a nonconvex feasible region, destroying the computational advantage of linear programming. Yet another strategy for weeding out trivial solutions is the introduction of a so-called *normalization* constraint. The normalization constraint proposed by Glover et al. for the hybrid model is

$$\sum_{g=1}^{2} \sum_{n=0}^{N_g} d_{gn} = 1.$$

Various pathologies have been connected to injudicious use of normalization constraints [9,10,13], including: unboundedness; trivial solutions; failure of the resulting discriminant function to adapt properly to rescaling or translation of the data (the optimal discriminant function after scaling or translating the data should be a scaled or translated version of the previously optimal discriminator, and the accuracy should be unchanged); and failure to find a discriminant function with perfect accuracy on the training samples when, in fact, they can be separated (which suggests that the discriminant function found will have suboptimal accuracy on the overall population). Indeed, Glover later changed the normalization of the hybrid model to [5]

$$-N_2 \cdot \mathbf{1}'\mathbf{X}_1\mathbf{w} + N_1 \cdot \mathbf{1}'\mathbf{X}_2\mathbf{w} = 1$$

to avoid some of these pathologies.

Multiple Group Problems

The use of a scalar-valued scoring function in an LPCM with $G > 2$ groups requires the a priori imposition of both a specific ordering and prescribed interval widths on the scores of the groups. This being impractical, attention turns to vector-valued functions. Whether using methods based on statistics or mixed integer programming, a common approach to the multiple group problem is to develop a separate scoring function for each group, and assign observations to the group whose scoring function yields the largest value at that observation. The linear programming analog would be to reward amounts by which the score $f_i(\mathbf{x})$ of an observation $\mathbf{x}$ from group i exceeds each $f_j(\mathbf{x}), j \neq i$ (or $\max_{j \neq i} f_j(\mathbf{x})$) and penalize differences in the opposite direction. This induces a proliferation of deviation variables (on the order of $(G-1)\sum_{g=1}^{G} N_g$). Other approaches may construct discriminant functions for all pairs of groups, or for each group versus all others, and then using a "voting" procedure to classify observations [15].

A good example of the use of a vector-valued scoring function is the work of Gochet et al. [7]. They begin with one scoring function per group, and in cases where two of those functions wind up identical, add additional functions to serve as tie-breakers. Their model is:

$$\begin{aligned}
\min \sum_{g=1}^{G} \sum_{g \neq h=1}^{G} & \mathbf{1}'\mathbf{e}_{gh} \\
\text{s.t. } & \mathbf{X}_g\left(\mathbf{w}_g - \mathbf{w}_h\right) + \left(w_{g0} - w_{h0}\right) \cdot \mathbf{1} + \mathbf{e}_{gh} - \mathbf{d}_{gh} = \mathbf{0} \\
& \sum_{g=1}^{G} \sum_{g \neq h=1}^{G} \mathbf{1}'\left(\mathbf{d}_{gh} - \mathbf{e}_{gh}\right) = q \\
& \mathbf{w}_g, w_{g0} \text{ free; } \mathbf{d}_{gh}, \mathbf{e}_{gh} \geq 0\,.
\end{aligned}$$

The scoring function corresponding to group g is $f_g(\mathbf{x}) = \mathbf{w}_g'\mathbf{x} + w_{g0}$. "Internal" and "external" deviations now represent amounts by which the scores of observations generated by the correct functions exceed or fall short of their scores from functions belonging to other groups. The first constraint is repeated for every pair of groups $g, h = 1, \ldots, G$, $g \neq h$. The second constraint, in which q is an arbitrary positive constant, is a normalization constraint intended to render infeasible both the trivial solution (all $\mathbf{w}_g$ identical) and solutions for which the total of the external deviations exceeds that of the internal deviations. If $\mathbf{w}_g = \mathbf{w}_h$ and $w_{g0} = w_{h0}$ for some $g \neq h$, the model is applied recursively to the subsamples from only those groups (possibly more than just g and h) that yielded identical scoring functions. The additional functions generated are used as tie-breakers.

Methods

The number of constraints in an LPCM approximately equals the number of training observations, while the number of variables can range from slightly more than the number of attributes to slightly more than the sum of the number of observations and the number of attributes, depending on which deviation variables are included in the model. In practice, the number of observations will exceed the number of attributes; indeed, if the difference is not substantial, the model runs the risk of overfitting the scoring function (in the statistical sense). When the number of deviation variables is small, then, the LPCM tends to have considerably more constraints than variables, and a number of authors have suggested solving its dual linear program instead, to reduce the amount of computation. Improvements in both hardware and software have lessened the need for this, but it may still be useful when sample sizes reach the tens or hundreds of thousands (which can happen, for example, when rating consumer credit, and in some medical applications).

See also

- ► Deterministic and Probabilistic Optimization Models for Data Classification
- ► Linear Programming
- ► Mixed Integer Classification Problems
- ► Statistical Classification: Optimization Approaches

References

1. Cortes C, Vapnik V (1995) Support-vector networks. Mach Learn 20(3):273–297
2. Erenguc SS, Koehler GJ (1990) Survey of mathematical programming models and experimental results for linear discriminant analysis. Manag Decis Econ 11:215–225
3. Freed N, Glover F (1981) A linear programming approach to the discriminant problem. Decis Sci 12:68–74
4. Freed N, Glover F (1981) Simple but powerful goal programming models for discriminant problems. Eur J Oper Res 7:44–60
5. Glover F (1990) Improved linear programming models for discriminant analysis. Decis Sci 21:771–785
6. Glover F, Keene SJ, Duea RW (1988) A new class of models for the discriminant problem. Decis Sci 19:269–280
7. Gochet W, Stam A, Srinivasan V, Chen S (1997) Multigroup discriminant analysis using linear programming. Oper Res 45:213–225
8. Hand DJ (1997) Construction and assessment of classification rules, Wiley, Chichester
9. Koehler GJ (1989) Unacceptable solutions and the hybrid discriminant model. Decis Sci 20:844–848
10. Koehler GJ (1990) Considerations for mathematical programming models in discriminant analysis. Manag Decis Econ 11:227–234
11. Mangasarian OL (1965) Linear and nonlinear separation of patterns by linear programming. Oper Res 13:444–452
12. Rosenblatt F (1958) The perceptron: A probabilistic model for information storage and organization in the brain. Psychol Rev 65(6):386–408
13. Rubin PA (1991) Separation failure in linear pgogramming discriminant models. Decis Sci 22:519–535
14. Stam A (1997) Nontraditional approaches to statistical classification: Some perspectives on Lp-norm methods. Ann Oper Res 74:1–36
15. Witten IH, Frank E (2005) Data mining: Practical machine learning tools and techniques, 2nd edn. Morgan Kaufmann, Amsterdam

Linear Space

LEONIDAS PITSOULIS
Princeton University, Princeton, USA

MSC2000: 15A03, 14R10, 51N20

Article Outline

Keywords

Linear algebra

Let F be a field, whose elements are referred to as *scalars*. A *linear space* V over F is a nonempty set on which the operations of addition and scalar multiplication are defined. That is, for any $x, y \in V$, we have $x + y \in V$, and for any $x \in V$ and $\alpha \in F$ we have $\alpha\, x \in V$. Furthermore, the following properties must be satisfied:

1) $x + y = y + x$, $\forall\, x, y \in V$.
2) $(x + y) + z = x + (y + z)$, $\forall\, x, y, z \in V$.
3) There exists an element $0 \in V$, such that $x + 0 = x$, $\forall x \in V$.
4) $\forall\, x \in V$, there exists $-x \in V$ such that $x + (-x) = 0$.
5) $\alpha\,(x + y) = \alpha\, x + \alpha\, y$, $\forall\, \alpha \in F$, $\forall\, x, y \in V$.
6) $(\alpha + \beta)x = \alpha x + \beta x$, $\forall\, \alpha, \beta \in F$, $\forall\, x \in V$.

7) $(\alpha\beta)x = \alpha(\beta x)$, $\forall\ \alpha, \beta \in F, \forall\ x \in V$.
8) $1x = x$, $\forall x \in V$.

The elements of V are called *vectors*, and V is also called a *vector space*.

See also

► Affine Sets and Functions
► Linear Programming

Lipschitzian Operators in Best Approximation by Bounded or Continuous Functions

VASANT A. UBHAYA
Department Computer Sci. and Operations Research, North Dakota State University, Fargo, USA

MSC2000: 65K10, 41A30, 47A99

Article Outline

Keywords

Approximation problem; Minimum distance problem; Best approximation; Best estimate; Selection; Continuous selection operator; Lipschitzian selection operator; Bounded function; Continuous function; Uniform norm; Quasiconvex function; Convex function; Isotone functions; Majorants and minorants

Stated in simplest terms, this article considers, in an abstract mathematical framework, a curve fitting or estimation problem where a given set of data points f is approximated or estimated by an element from a set K so that the estimate of f is least affected by perturbations in f.

Let X be a normed linear space with norm $\|\cdot\|$ and K be any (not necessarily convex) nonempty subset of X. For any f in X, let

$$d(f, K) = \inf\{\|f - h\| :\ h \in K\} \tag{1}$$

denote the shortest distance from f to K. Let also, for f in X,

$$P(f) = P_K(f) = \{h \in K:\ \|f - h\| = d(f, K)\}.$$

The set-valued mapping P on X is called the *metric projection* onto K. It is also called the *nearest point mapping*, *best approximation operator*, *proximity map*, etc. If $P(f) \neq \emptyset$, then each element in it is called a *best approximation* to (or a *best estimate* of) f from K. In practical curve fitting or estimation problems, f represents the given data and the set K is dictated by the underlying process that generates f. Because of random disturbance or noise, f is in general not in K, and it is required to estimate f by an element of K. See [7,12] and other references given there for a discussion of such problems and the use of various norms or distance functions in approximation. An approximation problem or a minimum distance problem such as (1) involves finding a best approximation, investigating its uniqueness and other properties, and developing algorithms for its computation. If $P(f) \neq \emptyset$ (respectively, $P(f)$ is a singleton) for each $f \in X$, then K is called *proximinal* (respectively, *Chebyshev*).

If K is proximinal, then we define a *selection operator*, or simply a *selection*, to be any (single valued) function T on X into K so that $T(f) \in P(f)$ for every $f \in X$. If K is Chebyshev, then clearly $T = P$ and T is unique. A *continuous selection operator* is a selection T which is continuous. There is a vast literature available on the existence and properties of continuous selections including some survey papers. See, e. g., [1,2,3,6,8] and other references given there. A more difficult problem is finding a *Lipschitzian selection operator* (LSO) i. e., a selection T which satisfies

$$\|T(f) - T(h)\| \leq c(T)\,\|f - h\|, \quad \text{all } f, h \in X, \tag{2}$$

where $c(T)$ (a positive constant depending upon T) is the smallest value satisfying (2). An LSO T is called an *optimal Lipschitzian selection operator* (OLSO) if $c(T) \leq c(T')$ for all LSO T'. If the operator T in (2) is OLSO, then (2) shows that the estimate $T(f)$ of f is least sensitive to changes in the given data f. Consequently, $T(f)$ is the most desirable estimate of f. The concept of an OLSO was introduced in [12] and the existence of an LSO and OLSO was investigated in [13,14,15,16,17]. If X is a Hilbert space and $K \subset X$ is nonempty, closed and

convex, then K is Chebyshev. Then T, which maps f to its unique best approximation, is an LSO, i. e., T satisfies (2) with $c(T) = 1$. For a proof see [5, p. 100]. Since T is unique, it is also trivially OLSO. For other spaces, the results are not so straightforward.

In this paper we present several results which identify LSOs and OLSOs in approximation problems on the space of bounded or continuous functions. We illustrate these results by examples.

Lipschitzian Selection Operators

Let S be any set and B denote the Banach space of real bounded functions f on S with the uniform norm $\|f\| = \sup\{|f(s)|: s \in S\}$. Similarly, when S is topological, denote by $C = C(S)$, the space of real bounded and continuous functions on S, again, with the uniform norm $\|\cdot\|$. Let $X = B$ or C in what follows. We let $f \in X$, $K \subset X$ and $d(f, K)$ as above. We let $d(f) = d(f, K)$ for convenience. For f in X, define $K_f = \{k \in K : k \leq f\}$ and $K'_f = \{k \in K : k \geq f\}$. Let

$$\overline{f}(s) = \sup\left\{k(s):\ k \in K_f\right\}, \quad s \in S,$$
$$\underline{f}(s) = \inf\left\{k(s):\ k \in K'_f\right\}, \quad s \in S.$$

We state three conditions below, they are identical for $X = B$ or C.

1) If $k \in K$, then $k + c \in K$ for all real c.
2) If $f \in X$, then $\overline{f} \in K$.
3) If $f \in X$, then $\underline{f} \in K$.

If $\overline{f}$ and $\underline{f}$ are in K, then they are called the *greatest K-minorant* and the *smallest K-majorant* of f, respectively. Note that condition 2) (respectively, 3)) implies that the pointwise maximum (respectively, minimum) of any two functions in K is also in K. This can be easily established by letting $f = \max\{f_1, f_2\}$ (respectively, $f = \min\{f_1, f_2\}$) where $f_1, f_2 \in K$. We call a $g \in K$ the *maximal* (respectively, *minimal*) *best approximation* to $f \in X$ if $g \geq g'$ (respectively, if $g \leq g'$) for all best approximations g' to f.

Theorem 1 *Consider (1) with $X = B$ or C, and any $K \subset X$.*

a) Assume K is not necessarily convex. Suppose that conditions 1) and 2) hold for K. Then $d(f) = \|f - \overline{f}\|/2$ and $f' = \overline{f} + d(f)$ is the maximal best approximation to f. Also $\|f' - h'\| \leq 2\|f - h\|$ for all $f, h \in X$. The operator T defined $T(f) = f'$ is an LSO with $c(T) = 2$.

b) Assume K is not necessarily convex. Suppose that conditions 1) and 3) hold for K. Then a) holds with $\overline{f}$ replaced by $\underline{f}$ and with $f' = \underline{f} - d(f)$, which is the minimal best approximation to f.

c) Assume K is convex. Suppose that conditions 1), 2) and 3) hold for K. Then a) and b) given above apply. In addition, $d(f) = (\|\underline{f} - \overline{f}\|)/2$. A g in K is a best approximation to f if and only if $\underline{f} - d(f) \leq g \leq \overline{f} + d(f)$. Moreover, if $f' = (\underline{f} + \overline{f})/2$, then f' is a best approximation to f and $\|f' - h'\| \leq \|f - h\|$ for all $f, h \in X$. The operator T defined by $T(f) = f'$ is an OLSO with $c(T) = 1$.

The following theorem shows that the existence of a maximal (respectively, minimal) best approximation to (1) implies condition 2) (respectively, 3)).

Theorem 2 *Consider (1) with $X = B$ or C, and any $K \subset X$. Assume condition 1) holds for K. Assume that the pointwise maximum (respectively, minimum) of two function in K is also in K. Then condition 2) (respectively, 3)) holds if the maximal (respectively, minimal) best approximation to f exists. This best approximation then equals $\overline{f} + d(f)$ (respectively, $\mathrm{f} - d(f)$).*

The above theorems and the next one appear in [14,15]. Their proofs are available there. We now define another approximation problem, closely related to (1). Let

$$\overline{d}(f) = d(f, K_f) = \inf\left\{\|f - h\|:\ h \in K_f\right\}. \qquad (3)$$

The problem is to find a $g \in \{h \in K_f : \|f - h\| = d(f, K_f)\}$, called a *best approximation* to f from K_f.

Theorem 3 *Consider (3) with $X = B$ or C, and any $K \subset X$ which is not necessarily convex.*

a) Suppose that conditions 1) and 2) hold for K. Then $\overline{f}$ is the maximal best approximation to f and $\overline{d}(f) = \left\|f - \overline{f}\right\| = 2d(f)$. The operator T defined by $T(f) = \overline{f}$ is the unique OLSO with $c(T) = 1$.

b) Assume condition 1) holds for K. Assume that the pointwise maximum of two functions in K is also in K. Then condition 2) holds if the maximum best approximation to f exists. This best approximation then equals $\overline{f}$.

Examples and Applications

Example 4 (Approximation by quasiconvex functions.) Let $S \subset \mathbf{R}^n$ be nonempty convex and consider $B = B(S)$. For $C = C(S)$ assume S is nonempty, compact and convex. A function $h \in B$ is called *quasiconvex* if

$$h(\lambda s + (1-\lambda)t) \le \max\{h(s), h(t)\}, \quad \text{for all } s, t \in S, \quad 0 \le \lambda \le 1. \tag{4}$$

Equivalently, h in B is quasiconvex if one of the following conditions holds [9,10]:

- $\{h \le c\}$ is convex for all real c;
- $\{h < c\}$ is convex for all real c.

Let K be the set of all quasiconvex functions in B. It is easy to show that K and $K \cap C$ are closed cones which are not convex and both satisfy condition 1) above (K is a cone if $\lambda h \in K$ whenever $h \in K$ and $\lambda \ge 0$.) The greatest K-minorant of f is called the *greatest quasiconvex minorant* of f. Using (4) it is easy to show that if $f \in B$ then such a minorant $\overline{f}$ exits in B. The next proposition shows that if $f \in C$ then $\overline{f} \in C$.

Let Π be the set of all convex subsets of S. Clearly, φ, $S \in \Pi$. For any $A \subset \mathbf{R}^n$, we denote by $\mathrm{co}(A)$ the *convex hull* of A, i. e., the smallest convex set containing A.

Proposition 5 *Let $f \in X$ and let*

$$f^0(P) = \inf\{f(t):\ t \in S \backslash P\}, \quad P \in \Pi,$$
$$\overline{f}(s) = \sup\left\{f^0(P):\ P \in \Pi,\ s \in S \backslash P\right\},\ s \in S.$$

Then the following holds:

- *If $f \in B$ (respectively, C) then $\overline{f} \in B$ (respectively, C) and is quasiconvex. It is the greatest quasiconvex minorant of f.*
- *An $h \in B$ is the greatest quasiconvex minorant of $f \in B$ if and only if*

$$\{h < c\} = \mathrm{co}\{f < c\} \quad \textit{for all real } c. \tag{5}$$

- *An $h \in B$ is the greatest quasiconvex minorant of $f \in C$ if and only if (5) holds or, equivalently, $\{h \le c\} = \mathrm{co}\{h \le c\}$ for all real c.*

This proposition and its proof appear in [15]. The proposition shows that condition 2) holds for K and $K \cap C$. Hence, Theorems 1a) and 3a) apply to $X = B$ and K, and also to $X = C$ and $K \cap C$. In particular, Theorem 1a) shows that in each of these two cases the operator T mapping f to $\overline{f}$ is LSO with $c(T) = 2$. Now the example given in [13, p. 332] shows that T is OLSO.

Example 6 (Approximation by convex functions.) Let $S \subset \mathbf{R}^n$ be nonempty convex and consider $B = B(S)$. A function $h \in B$ is called *convex* if $h(\lambda s + (1-\lambda)t) \le \lambda\, h(s) + (1-\lambda)\, h(t)$, for all $s, t \in S$ and all $0 \le \lambda \le 1$. Clearly, a convex function is quasiconvex. Let K be the set of all convex functions in B. It is easy to show that K is a closed convex cone and satisfies condition 1). The greatest K-minorant of f is called the *greatest convex minorant* of f. It follows at once from the definition of a convex function that if $f \in B$ then such a minorant $\overline{f}$ exists in B. Condition 2) therefore holds for K. Hence, Theorems 1a) and 3a) apply to $X = B$ and K. In particular, the LSO T of Theorem 1a) mapping f to $\overline{f}$ with $c(T) = 2$ can be shown to be an OLSO by using an example as in [13, p. 334].

Now consider approximation of a continuous function by continuous convex functions. For this case we let $S \subset \mathbf{R}^n$ be a polytope which is defined to be the convex hull of a finitely many points in $\mathbf{R}^n$. It is compact, convex and locally simplicial [11]. Let $K \subset C = C(S)$ be the set of continuous convex functions. It is easy to show that K is a closed convex cone. Again condition 1) holds for K. We assert that if $f \in C$, then $\overline{f}$ is convex and continuous. This will establish that $\overline{f}$ is the greatest convex minorant of f. To establish the assertion note that $\overline{f}$ is convex since it is the pointwise supremum of convex functions. Since S is locally simplicial, [11, Corol. 17.2.1; Thm. 10.2] show that $\overline{f}$ is continuous on S. Thus, condition 2) holds for K. Hence Theorems 1a) and 3a) apply to $X = C$ and K. In particular, the LSO T of Theorem 1a) mapping f to $\overline{f}$ with $c(T) = 2$ can be shown to be an OLSO by using the same example as in the bounded case above since the sequence used in that example consists of continuous functions [13].

Example 7 (Approximation by isotone functions.) Let S be any set with partial order $\le$. A *partial order* is a relation $\le$ on S satisfying [4, p. 4]:

- *reflexivity*, i. e., $s \le s$ for all $s \in S$; and
- *transitivity*, i. e., if $s, t, v \in S$, and $s \le t$ and $t \le v$, then $s \le v$.

A partial order is *antisymmetric* if $s \le t$ and $t \le s$ imply $s = t$. We do not include this antisymmetry condition in the partial order for sake of generality. We consider $B = B(S)$ as before, and define a function k in B to be *isotone*

if $k(s) \leq k(t)$ whenever $s, t \in S$ and $s \leq t$. Let $K \subset B$ be the set of all isotone functions. It is easy to see that K is a closed convex cone. It is nonempty since the zero function is in K. It is easy to verify that conditions 1), 2) and 3) apply to K. Thus the greatest isotone minorant $\overline{f}$ and the smallest isotone majorant $\underline{f}$ of an f in B exist. Theorem 1c) and 3a) apply and we conclude that the operator T of Theorem 1c), mapping f to $(\underline{f} + \overline{f})/2$, is OLSO with $c(T) = 1$ [15].

The next proposition gives explicit expressions for $\underline{f}$ and $\overline{f}$. We call a subset E of S a *lower* (respectively, *upper*) set if whenever $t \in E$ and $v \leq t$ (respectively, $t \leq v$), then $v \in E$. For s in S, let $L_s = \{t \in S,\ t \leq s\}$ and $U_s = \{t \in S,\ s \leq t\}$. Then, L_s (respectively, U_s) is the smallest lower (respectively, upper) set containing s, as may be easily seen.

Proposition 8

$$\underline{f}(s) = \sup\{f(t):\ t \in L_s\},$$
$$\overline{f}(s) = \inf\{f(t):\ t \in U_s\}.$$

For a proof, see [15].

Now we consider an application to C. Define $S = \times \{[a_i, b_i]: 1 \leq i \leq n\} \subset \mathbf{R}^n$, where $a_i < b_i$, and let $\leq$ be the usual partial order on vectors. Let $C = C(S)$ and let K be the set of isotone functions in C. It is easy to verify that K is a closed convex cone. Furthermore, if $f \in C$, then $\underline{f}, \overline{f} \in C$. We conclude, as before, that Theorems 1c) and 3a) apply. Various generalizations of this problem exist. See, for example, [12, Sect. 5], [15, Ex. 4.3], and [17].

As was observed in [16], the dual cone of K plays an important role in duality and approximation from K. Some properties of the cone K of isotone functions on a finite partially ordered set S and its dual cone are obtained in [18].

See also

► Convex Envelopes in Optimization Problems

References

1. Deutsch F (1983) A survey of metric selections. RC Sine (ed) Fixed points and Nonexpansive Mappings. In: Contemp. Math., vol 18. Amer Math Soc, Providence, pp 49–71
2. Deutsch F (1992) Selections for metric projections. SP Singh (ed) Approximation Theory, Spline Functions and Applications. Kluwer, Dordrecht, pp 123–137
3. Deutsch F, Li W, Park S-H (1989) Characterization of continuous and Lipschitz continuous metric selections in normed linear spaces. J Approx Theory 58:297–314
4. Dunford N, Schwartz JT (1958) Linear operators, Part I. Interscience, New York
5. Goldstein AA (1967) Constructive real analysis. Harper and Row, New York
6. Li W (1991) Continuous selections for metric projections and interpolating subspaces. In: Brosowski B, Deutsch F, Guddat J (eds) Approximation and Optimization, vol 1. P. Lang, Frankfurt, pp 1–108
7. Liu M-H, Ubhaya VA (1997) Integer isotone optimization. SIAM J Optim 7:1152–1159
8. Nurnberger G, Sommer M (1984) Continuous selections in Chebyshev approximation. In: Brosowski B, Deutsch F (eds) Parametric Optimization and Approximation. Internat Ser Numer Math, vol 72. Birkhäuser, Boston, pp 248–263
9. Ponstein J (1967) Seven kinds of convexity. SIAM Rev 9:115–119
10. Roberts AW, Varberg DE (1973) Convex functions. Acad. Press, New York
11. Rockafellar RT (1970) Convex analysis. Princeton Univ. Press, Princeton
12. Ubhaya VA (1985) Lipschitz condition in minimum norm problems on bounded functions. J Approx Theory 45:201–218
13. Ubhaya VA (1988) Uniform approximation by quasi-convex and convex functions. J Approx Theory 55:326–336
14. Ubhaya VA (1989) Lipschitzian selections in approximation from nonconvex sets of bounded functions. J Approx Theory 56:217–224
15. Ubhaya VA (1990) Lipschitzian selections in best approximation by continuous functions. J Approx Theory 61:40–52
16. Ubhaya VA (1991) Duality and Lispschitzian selections in best approximation from nonconvex cones. J Approx Theory 64:315–342
17. Ubhaya VA (1992) Uniform approximation by a nonconvex cone of continuous functions. J Approx Theory 68:83–112
18. Ubhaya VA (2001) Isotone functions, dual cones, and networks. Appl Math Lett 14:463–467

Load Balancing for Parallel Optimization Techniques

LBDOP

Ananth Grama[1], Vipin Kumar[2]

[1] Purdue University, West Lafayette, USA

[2] University Minnesota, Minneapolis, USA

MSC2000: 68W10, 90C27

Article Outline

Keywords

Parallel algorithm; Load balancing; Tree search; Graph search

Discrete optimization problems are solved using a variety of state space search techniques. The choice of technique is influenced by a variety of factors such as availability of heuristics and bounds, structure of state space, underlying machine architecture, availability of memory, and optimality of desired solution. The computational requirements of these techniques necessitates the use of large scale parallelism to solve realistic problem instances. In this chapter, we discuss parallel processing issues relating to state space search.

Parallel platforms have evolved significantly over the past two decades. Symmetric multiprocessors (SMPs), tightly coupled message passing machines, and clusters of workstations and SMPs have emerged as the dominant platforms. From an algorithmic standpoint, the key issues of locality of data reference and load balancing are key to effective utilization of all these platforms. However, message latencies, hardware support for shared address space and mutual exclusion, communication bandwidth, and granularity of parallelism all play important roles in determining suitable parallel formulations. A variety of metrics have also been developed to evaluate the performance of these formulations. Due to the nondeterministic nature of the computation, traditional metrics such as parallel runtime and speedup are difficult to quantify analytically. The scalability metric, Isoefficiency, has been used with excellent results for analytical modeling of parallel state space search.

The state spaces associated with typical optimization problems can be fashioned in the form of either a graph or a tree. Exploiting concurrency in graphs is more difficult compared to trees because of the need for replication checking. The availability of heuristics for best-first search imposes constraints on parallel exploration of states in the state space. For the purpose of parallel processing, we can categorize search techniques loosely into three classes: *depth-first tree search techniques* (a tree search procedure in which the deepest of the current nodes is expanded at each step), *best-first tree search techniques* (a tree search procedure in which nodes are expanded based on a global (heuristic) measure of how likely they are to lead to a solution), and *graph search techniques* (a search requiring additional computation for checking if a node has been encountered before, since a node can be reached from multiple paths). Many variants of these basic schemes fall into each of these categories as well.

Parallel Depth-First Tree Search

Search techniques in this class include ordered depth-first search, iterative deepening A∗ (IDA∗), and depth—first branch and bound (DFBB). In all of these techniques, the key ingredient is the depth-first search of a state space (cost-bounded in the case of IDA∗ and DFBB). DFS was among the first applications explored on early parallel computers. This is due to the fact that DFS is very amenable to parallel processing. Each subtree in the state space can be explored independently of other subtrees in the space. In simple DFS, there is no exchange of information required for exploring different subtrees. This implies that it is possible to device simple parallel formulations by assigning a distinct subtree to each processor. However, the space associated with a problem instance can be highly unstructured. Consequently, the work associated with subtrees rooted at different nodes can be very different. Therefore, a naive assignment of a subtree rooted at a distinct node to each processor can result in considerable idling overhead and poor parallel performance. The problem of designing efficient parallel DFS algorithms can be viewed in two steps: the partitioning problem and the assignment problem. The partitioning problem addresses the issue of breaking up a given search space into two subspaces. The assignment problem then maps subspaces to individual processors.

There are essentially two techniques for partitioning a given search space: node splitting and stack splitting. In node splitting, the root node of a subtree is expanded

to generate a set of successor nodes. Each of these nodes represents a distinct subspace. While node splitting is easy to understand and implement, it can result in search spaces of widely varying sizes. Since the objective of the assignment problem is to balance load while minimizing work transfers, widely varying subtask sizes are not desirable. An alternate technique called stack splitting attempts to partition a search space into two by assigning some nodes at all levels leading up to the specified node. Thus if the current node is at level 4, stack splitting will split the stack by assigning some nodes at levels 1, 2, and 3 to each partition. In general, stack splitting results in a more even partitioning of search spaces than node splitting.

We can now formally state the assignment problem for parallel DFS as a mapping of subtasks to processors such that:

- the work available at any processor can be partitioned into independent work pieces as long as it is more than some nondecomposable unit;
- the cost of splitting and transferring work to another processor is not excessive (i. e. the cost associated with transferring a piece of work is much less than the computation cost associated with it);
- a reasonable work splitting mechanism is available; i. e., if work w at one processor is partitioned in 2 parts ψw and $(1-\psi)w$, then $1-\alpha > \psi > \alpha$, where α is an arbitrarily small constant;
- it is not possible (or is very difficult) to estimate the size of total work at a given processor.

A number of mapping techniques have been proposed and analyzed in literature [5,7,8,9,11,16]. These mapping techniques are either initiated by a processor with work (*sender initiated*, the processor with work initiates the work transfer) or a processor looking for work (*receiver initiated*, an idle processor initiates the work transfer). In the *global round robin request* (GRR, idle processors in the global round robin scheme request processors for work in a round-robin fashion using a single (global) counter) receiver initiated scheme, a single counter specifies the processor that must receive the next request for work. This ensures that work requests are uniformly distributed across all processors. However, this scheme suffers from contention at the processor holding the counter. Consequently, the performance of this scheme is poor beyond a certain number of processors. A message combining variant of this scheme (*GRR-M*, a variant of the global round robin scheme in which requests for value of global counter are combined to alleviate contention overheads) relies on combining intermediate requests for the counter into single request. This alleviates the contention and performance bottleneck of the GRR scheme. The *asynchronous round robin balancing* (ARR, i. e. each processor selects a target for work request in a round robin manner using a local counter) uses one counter at each processor. Each processor uses its counter to determine the next processor to query for work. While this scheme balances work requests in a local sense, these requests may become clustered in a global sense. In the *random polling scheme* (RP, i. e. idle processors send work requests to a randomly selected target processor), each processor selects a random processor and requests work. In *near-neighbor load balancing scheme* (NN, i. e. an idle processor requests one of its immediate neighbors for work), processors request work from their immediate neighbors. This scheme has the drawback that localized hot-spots may take a long time to even out.

In sender initiated schemes a processor with work can give some of its work to a selected processor [6,16]. This class of schemes includes the master-slave (MS) and randomized allocation (RA) schemes. In the *MS scheme*, a processor, designated master, generates a fixed number of work pieces. These work-pieces are assigned to processors as they exhaust previously assigned work. The master may itself become the bottleneck when the number of processors is large. Multilevel master-slave algorithms have been used to alleviate this bottleneck. *Randomized allocation schemes* are sender initiated counterparts of RP schemes. In randomized allocation, a processor sends a part of its work to a randomly selected processor.

The performance and scalability of these techniques is often dependent on the underlying architecture. Many of these techniques are, in principle scalable, i. e., they result in linear speedup on increasing the number of processors p as long as the size of the search space grows fast enough with p. It is desirable that this required rate of growth of problem size (also referred to as the iso-efficiency metric [10]) be as small as possible since it allows the use of a larger number of processors effectively for solving a given problem instance. In Table 1, we summarize the iso-efficiency functions of various load balancing techniques.

Scalability results of receiver initiated load balancing schemes for various architectures

Arch Scheme	Shared	H-cube	Mesh (2D)	W/S Cluster
ARR	$p^2 \log p$	$p^2 \log^2 p$	$p^{2.5} \log p$	$p^3 \log p$
NN	$p^2 \log p$	$p^{\log \frac{1+1/\alpha}{2}}$	$k\sqrt{p}$	$p^3 \log p$
GRR	$p^2 \log p$	$p^2 \log p$	$p^2 \log p$	$p^2 \log p$
GRR-M	$p \log p$	$p \log^2 p$	$p^{1.5} \log p$	
RP	$p \log^2 p$	$p \log^2 p$	$p^{1.5} \log^2 p$	$p^2 \log^2 p$
Lower Bound	p	$p \log p$	$p^{1.5}$	p^2

IDA∗ and DFBB search techniques use this basic parallel DFS algorithm for searching state space. In *IDA∗* , each processor has a copy of the global cost bound. Processors perform parallel DFS with this cost bound. At the end of each phase, the cost is updated using a single global operation. Some schemes for allowing different processors to work with different cost bound have also been explored. In this case, a solution cannot be deemed optimal until search associated with all previous cost bounds has been completed. *DFBB* technique uses a global current best solution to bound parallel DFS. Whenever a processor finds a better solution, it updates this global current best solution (using a broadcast in message passing machines and a lock-set in shared memory machines). DFBB and IDA∗ using these parallel DFS algorithms has been shown to yield excellent performance for various optimization problems [3,13,19].

In many optimization problems, the successors of nodes tend to be strongly ordered. In such cases, naive parallel formulations that ignore this ordering information will perform poorly since they are likely to expand a much larger subspace than those that explore nodes in the right order. Parallel DFS formulations for such spaces associate priorities with nodes. Nodes with largest depth and highest likelihood of yielding a solution are assigned the highest priority. Parallel ordered DFS then expands these nodes in a prioritized fashion.

Parallel Best-First Tree Search

Best-first tree search algorithms rely on an *open list* (i. e. a list of unexplored configurations sorted on their quality) to sort available states on the basis of their heuristic solution estimate. If this heuristic solution estimate is guaranteed to be an underestimate (as is the case in the A∗ algorithm), it can be shown that the solution found by BFS is the optimal solution. The presence of a globally ordered open list makes it more difficult to parallelize BFS. In fact, at the first look, BFS may appear inherently serial since a node with higher estimated solution cost must be explored only after all nodes with lower costs have been explored. However, it is possible that there may be multiple nodes with the best heuristic cost. If the number of such nodes is less than the number of available processors, then some of the nodes with poorer costs may also be explored. Since it is possible that these nodes are never explored by the serial algorithm, this may result in excess work by the parallel formulation resulting in deceleration anomalies. These issues of speedup anomalies resulting from excess (or lesser) work done by the parallel formulations of state space search are discussed later.

A simple parallel formulation of BFS uses a global open list. Each processor locks the list, extracts the best available node and unlocks the list. The node is expanded and heuristic estimates are determined for each successor. The open list is locked again and all successors are inserted into the open list. Note that since the state space is a tree, no replication checking is required. The open list is typically maintained in the form of a global heap. The use of a global heap is a source of contention. If the time taken to lock, remove, and unlock the top element of the heap is t_{access} and time for expansion is t_{exp}, then the speedup of the formulation is bounded by $(t_{access} + t_{exp})/t_{access}$. A number of techniques have been developed to effectively reduce the access time [17]. These techniques support concurrent access to heaps stored in shared memory while maintaining strict insertion and deletion ordering. While these increase the upper bound on possible speedup, the performance of these schemes is still bounded.

The contention associated with the global data structure can be alleviated by distributing the open list across processors. Now, instead of p processors sharing a single list, they operate on k distinct open lists. In the limiting case, each processor has its own open list. A simple parallel formulation based on this framework starts off with the initial state in one heap. As additional states are generated, they are shipped off to

the other heaps. As nodes become available in other heaps, processors start exploring associated state space using local BFS. While it is easy to keep all processors busy using this framework, it is possible that some of the processors may expand nodes with poor heuristic estimates that are never expanded by the serial formulation. To avoid this, we must ensure that all open lists have a share of the best globally available nodes. This is also referred to as *quality equalization* (the process of ensuring that all processors are working on regions of state-space of high quality). Since the quality of nodes evolves with time, quality equalization must be performed periodically. Several triggering mechanisms have been developed to integrate quality equalization with load balancing [1,19]. A simple triggering mechanism tracks the best node in the system. The best node in the local heap is compared to the best node in the system and if it is considerably worse, an equalization process is initiated. Alternately, an equalization process may be initiated periodically. The movement of nodes between various heaps may itself be fashioned in a well defined topology. Lists may be organized into rings, shared blackboards, or hierarchical structures. These have been explored for several applications and architectures. Speedups in excess of 950 have been demonstrated on 1024 processor hypercubes in the context of TSPs formulated as best-first tree search problems [2].

Searching State Space Graphs

Searching state space graphs presents additional challenges since we must check for replicated states during search. The simplest strategy for dealing with graphs is to unroll them into trees. The overhead of unrolling a graph into a tree may range from a constant to an exponential factor. If the overhead is a small constant factor, the resulting tree may be searched using parallel DFS or BFS based techniques. However, for most graph search problems, this is not a feasible solution.

Graph search problems rely on a *closed list* (i. e. a list of all configurations that have been previously encountered) that keeps track of all nodes that have already been explored. Closed lists are typically maintained as hash tables for searching. In a shared memory context, insertion of nodes into the closed list requires locking of the list. If there is a single lock associated with the entire list, the list must be locked approximately as many times as the total number of nodes expanded. This represents a serial bottleneck. The bottleneck can be alleviated by associating multiple locks with the closed list. Processors lock only relevant parts of the closed list into which the node is being inserted.

Distributed memory versions of this parallel algorithm physically distribute the closed list across the processors. As nodes are generated, they are hashed to the appropriate processor that holds the respective part of the hash table. Search is performed locally at this processor and the node is explored further at this processor if required. This has two effects: if the hash function associated with the closed list is truly randomized, this has the effect of load balancing using randomized allocation. Furthermore, since nodes are randomly allocated to processors, there is a probabilistic quality equalization for heuristic search techniques. These schemes have been studied by many researchers [14,15]. Assuming a perfectly random hash function, it has been shown that if the number of nodes originating at each processor grows as $O(\log p)$, then each processor will have asymptotically equal number of nodes after the hash operation [15]. Since each node is associated with a communication, this puts constraints on the architecture bandwidth. Specifically, the bisection width of the underlying architecture must increase linearly with the number of processors for this formulation to be scalable.

A major drawback of graph search techniques such as BFS is that its memory requirement grows linearly with the search space. For large problems, this memory requirement becomes prohibitive. Many limited-memory variants of heuristic search have been developed. These techniques rely on retraction or delayed expansion of less promising nodes to reduce memory requirement. In the parallel processing context, retractions lead to additional communication and indexing for parent-child relationships [4].

Anomalies in Parallel Search

As we have seen above, it is possible for parallel formulations to do more or less work than the serial search algorithm. The ratio of nodes searched by the parallel and serial algorithms is called the *search overhead factor* (i. e. the ratio of excess work done by a parallel search formulation with respect to its serial formula-

tion). A search overhead factor of greater than one indicates a deceleration anomaly and less than one indicates an acceleration anomaly. An acceleration anomaly manifests itself in a speedup greater than p on p processors. It can be argued however that in these cases, the base sequential algorithm is suboptimal and a time-multiplexed serialization of the parallel algorithm is in fact a superior serial algorithm.

In DFS and related techniques, parallel formulations might detect solutions available close to the root on alternate branches, whereas serial formulations might search large parts of the tree to the left before reaching this node. Conversely, parallel formulations might also expand a larger number of nodes than the serial version. There are situations, in which parallel DFS can have a search overhead factor of less than 1 on the average, implying that the serial search algorithm in the situation is suboptimal. V. Kumar and V.N. Rao [18] show that if no heuristic information is available to order the successors of a node, then on the average, the speedup obtained by parallel DFS is superlinear if the distribution of solutions is nonuniform.

In BFS, the strength of the heuristic determines the search overhead factor. When strong heuristics are available, it is likely that expanding nodes with lower heuristic values will result in wasted effort. In general, it can be shown that for any given instance of BFS, there exists a number k such that expanding more than k nodes in parallel from a global open list leads to wasted computation [12]. This situation gets worse with distributed open lists since expanded nodes have locally minimum heuristics that are not the best nodes across all open lists. In contrast, the search overhead factor can be less than one if there are multiple nodes with identical heuristic estimates and one of the processors picks the right one.

Applications of Parallel Search Techniques

Parallel search techniques have been applied to a variety of problems such as integer and mixed integer programming, and quadratic assignment for applications ranging from path planning and resource location to VLSI packaging. Quadratic assignment problems from the Nugent–Eschermann test suites with up to 4.8×10^{10} nodes have been solved on parallel machines in days. Traveling salesman problems with thousands of cities and mixed integer programming problems with thousands of integer variable are within the reach of large scale parallel machines. While the use of parallelism increases the range of solvable problems, designing effective heuristic functions is critical. This has the effect of reducing effective branching factor and thus inter-node concurrency. However, the computation of the heuristic can itself be performed in parallel. The use of intra-node parallelism in addition to inter-node parallelism has also been explored. While significant amounts of progress has been made in effective use of parallelism in discrete optimization, with the development of new heuristic functions, opportunities for significant contributions abound.

See also

- ▶ Asynchronous Distributed Optimization Algorithms
- ▶ Automatic Differentiation: Parallel Computation
- ▶ Heuristic Search
- ▶ Interval Analysis: Parallel Methods for Global Optimization
- ▶ Parallel Computing: Complexity Classes
- ▶ Parallel Computing: Models
- ▶ Parallel Heuristic Search
- ▶ Stochastic Network Problems: Massively Parallel Solution

References

1. Cun BL, Roucairol C (1995) BOB: A unified platform for implementing branch-and-bound like algorithms. Techn. Report Univ. Versailles Saint Quentin 16
2. Dutt S, Mahapatra NR (1994) Scalable load-balancing strategies for parallel A* algorithms. J Parallel Distributed Comput 22(3):488–505, Special Issue on Scalability of Parallel Algorithms and Architectures (Sept. 1994)
3. Eckstein J (1997) Distributed versus centralized storage and control for parallel branch and bound: Mixed integer programming on the CM-5. Comput Optim Appl 7(2):199–220
4. Evett M, Hendler J, Mahanti A, Nau D (1990) PRA*: A memory-limited heuristic search procedure for the connection machine. Proc. Third Symp. Frontiers of Massively Parallel Computation, pp 145–149
5. Finkel RA, Manber U (Apr. 1987) DIB - A distributed implementation of backtracking. ACM Trans Program Languages and Systems 9(2):235–256
6. Furuichi M, Taki K, Ichiyoshi N (1990) A multi-level load balancing scheme for OR-parallel exhaustive search programs on the multi-PSI. Proc. Second ACM SIGPLAN Symp. Principles and Practice of Parallel Programming, pp 50–59

7. Janakiram VK, Agrawal DP, Mehrotra R (1988) A randomized parallel backtracking algorithm. IEEE Trans Comput C-37(12):1665–1676
8. Karp R, Zhang Y (1993) Randomized parallel algorithms for backtrack search and branch-and-bound computation. J ACM 40:765–789
9. Karypis G, Kumar V (oct. 1994) Unstructured tree search on SIMD parallel computers. IEEE Trans Parallel and Distributed Systems 5(10):1057–1072
10. Kumar V, Grama A, Gupta A, Karypis G (1994) Introduction to parallel computing: Algorithm design and analysis. Benjamin Cummings and Addison-Wesley, Redwod City, CA/Reading, MA
11. Kumar V, Grama A, Rao VN (July 1994) Scalable load balancing techniques for parallel computers. J Parallel Distributed Comput 22(1):60–79
12. Lai TH, Sahni S (1984) Anomalies in parallel branch and bound algorithms. Comm ACM 27(6):594–602
13. Lee EK, Mitchell JE (1997) Computational experience of an interior-point algorithm in a parallel branch-and-cut framework. Proc. SIAM Conf. Parallel Processing for Sci. Computing,).
14. Mahapatra NR, Dutt S (July 1997) Scalable global and local hashing strategies for duplicate pruning in parallel A* graph search. IEEE Trans Parallel and Distributed Systems 8(7):738–756
15. Manzini G, Somalvico M (1990) Probabilistic performance analysis of heuristic search using parallel hash tables. Proc. Internat. Symp. Artificial Intelligence and Math.,
16. Ranade AG (1991) Optimal speedup for backtrack search on a butterfly network. Proc. Third ACM Symp. Parallel Algorithms and Architectures,
17. Rao VN, Kumar V (1988) Concurrent access of priority queues. IEEE Trans Comput C-37(12):1657–1665
18. Rao VN, Kumar V (Apr 1993) On the efficicency of parallel backtracking. IEEE Trans Parallel and Distributed Systems 4(4):427–437. Also available as: Techn. Report 90–55, Dept. Computer Sci. Univ. Minnesota
19. Tschvke S, L-ling R, Monien B (1995) Solving the traveling salesman problem with a distributed branch-and-bound algorithm on a 1024 processor network. Proc. 9th Internat. Parallel Processing Symp. (April 1995), 182–189

Local Attractors for Gradient-related Descent Iterations

JOSEPH C. DUNN
Math. Department, North Carolina State University, Raleigh, USA

MSC2000: 49M29, 65K10, 90C06

Article Outline

Keywords

Unconstrained minimization; Gradient-related descent; Newtonian descent; Singular local attractors; Asymptotic convergence rates

In the classic unconstrained minimization problem, a continuously differentiable real-valued function f is given on a normed vector space $\mathbf{X}$ and the goal is to find points in $\mathbf{X}$ where the *infimum* of f is achieved or closely approximated. Descent methods for this problem start with some nonoptimal point x^0, search for a neighboring point x^1 where $f(x^1) < f(x^0)$, and so on ad infinitum. At each stage, the search is typically guided by a *local* model based on derivatives of f.

If f is convex and every local minimizer is therefore automatically a global minimizer, then well-designed descent methods can indeed generate *minimizing sequences*, i. e., sequences $\{x^k\}$ for which

$$\lim_{k\to\infty} f(x^k) = \inf_{x\in\mathbf{X}} f(x). \tag{1}$$

On the other hand, nonconvex cost functions can have multiple *local minimizers* and any of these may attract the iterates of the standard descent schemes. This behavior is examined here for a large class of gradient-related descent methods, and for local minimizers that need not satisfy the usual nonsingularity hypotheses. In addition, the analytical formulation adopted yields nontrivial local convergence theorems in infinite-dimensional normed vector spaces $\mathbf{X}$. Such theorems are not without computational significance since

they often help to explain emerging trends in algorithm behavior for increasingly refined finite-dimensional approximations to underlying infinite-dimensional optimization problems.

Differentials and Gradients

In a general normed vector space $\mathbf{X}$, the first (Fréchet) differential of f at a point x is a linear function $f'(x)$: $\mathbf{X} \to \mathbf{R}^1$ that satisfies the following conditions:

$$\left|\left\| f'(x) \right\|\right| \stackrel{\text{def}}{=} \sup_{\|u\|=1} \left| f'(x)u \right| < \infty, \tag{2}$$

$$\lim_{\|d\|\to 0} \frac{|f(x+d) - f(x) - f'(x)d|}{\|d\|} = 0. \tag{3}$$

Since $f'(x)d$ is linear in d, condition (2) holds if and only if $f'(x)d$ is continuous in d. Condition (2) is automatically satisfied in any finite-dimensional space $\mathbf{X}$. The remaining condition (3) asserts that $f(x)+ f'(x)d$ asymptotically approximates $f(x+d)$ with an $o(\|d\|)$ error as d approaches zero. At most one linear function can satisfy these conditions in some norm on $\mathbf{X}$. If conditions (2) and (3) do hold in the norm $\|\cdot\|$, then f is said to be (Fréchet) *differentiable* at x (relative to the norm $\|\cdot\|$). If f is differentiable near $x \in \mathbf{X}$ and if

$$\lim_{\|y-x\|\to 0} \left|\left\| f'(y) - f'(x) \right\|\right| = 0, \tag{4}$$

then f is continuously differentiable at x. Note that in finite-dimensional spaces, all norms are equivalent and conditions (2)–(4) hold in any norm if they hold in some norm. However, two norms on the same infinite-dimensional space need not be equivalent, and continuity and differentiability are therefore *norm-dependent properties* at this level of generality.

In the Euclidean space $\mathbf{X} = \mathbf{R}^n$, f is continuously differentiable if and only if the partial derivatives of f are continuous; moreover, when f has continuous partial derivatives, $f'(x)$ is specified by the familiar formula,

$$f'(x)d = \langle \nabla f(x), d \rangle, \tag{5}$$

where $\langle \cdot, \cdot \rangle$ is the standard Euclidean inner product and $\nabla f(x)$ is the corresponding *gradient* of f at x, i. e.,

$$\langle x, y \rangle = \sum_{i=1}^{n} x_i y_i$$

and

$$\nabla f(x) = \left(\frac{\partial f}{\partial x_1}(x), \ldots, \frac{\partial f}{\partial x_n}(x) \right).$$

When $\nabla f(\cdot)$ is continuous, conditions (2)–(4) can be proved for the linear function in (5) with a straightforward application of the chain rule, Cauchy's inequality and the one-dimensional mean value theorem. In addition, it can be shown that $d = \nabla f(x)$ is the unique solution of the equations,

$$\|d\| = \left|\left\| f'(x) \right\|\right| \tag{6}$$

and

$$f'(x)d = \left|\left\| f'(x) \right\|\right| \, \|d\|, \tag{7}$$

where $\|\cdot\|$ and $|||\cdot|||$ are induced by the Euclidean inner product on $\mathbf{R}^n$.

The circumstances in the Euclidean space $\mathbf{R}^n$ suggest a natural extension of the gradient concept in general normed vector spaces $\mathbf{X}$. Let f be differentiable at $x \in \mathbf{X}$. Then any vector $d \in \mathbf{X}$ that satisfies conditions (6)–(7) will be called a *gradient vector* for f at x. Note that the symbols $\|\cdot\|$ and $|||\cdot|||$ in (6)–(7) now signify the norm provided on $\mathbf{X}$ and the corresponding operator norm in (2). Depending on the space $\mathbf{X}$, its norm $\|\cdot\|$ and the point x, conditions (6)–(7) may have no solutions for d, or a unique solution, or infinitely many solutions.

In any finite-dimensional space $\mathbf{X}$, linear functions are continuous, the unit sphere $\{u \in \mathbf{X}: \|u\| = 1\}$ is compact, the supremum in (2) is therefore attained at some unit vector u, and the existence of solutions d for (6)–(7) is consequently guaranteed. On the other hand, f may have infinitely many distinct gradients at a point x if the norm on $\mathbf{X}$ is not strictly convex. For example, if $\mathbf{X} = \mathbf{R}^n$ and $\|x\| = \max_{1 \le i \le n} |x_i|$, then $f'(x)$ is prescribed by (5), and

$$\left|\left\| f'(x) \right\|\right| = \sum_{i=1}^{n} \left| \frac{\partial f}{\partial x_i}(x) \right|.$$

Moreover, d is a gradient vector for f at x if and only if

$$d = \left|\left\| f'(x) \right\|\right| u$$

and

$$u_i \in \operatorname{sgn}\left(\frac{\partial f}{\partial x_i}(x) \right),$$

where $\mathrm{sgn}(t) = \{-1\}$ or $[-1, 1]$ or $\{1\}$ for $t < 0$, $t = 0$ and $t > 0$, respectively.

The existence of gradients can also be proved in reflexive infinite-dimensional spaces $\mathbf{X}$ where bounded linear functions are weakly continuous and closed unit balls are weakly compact. In nonreflexive spaces, conditions (6)–(7) may not have solutions d; however, in any normed vector space and for any fixed arbitrarily small $\nu \in (0, 1)$, the relaxed conditions,

$$\|d\| = \left|\left\|f'(x)\right\|\right| \tag{8}$$

and

$$f'(x)d \geq (1-\nu)\left|\left\|f'(x)\right\|\right| \ \|d\|, \tag{9}$$

always have solutions d. This follows easily from (2) and the meaning of sup. The solutions of (8)–(9) will be called *ν-approximate gradients* of f at x. They occupy a central position in the present formulation of the subject algorithms.

Gradient-Related Descent Methods

If $f'(x) = 0$, then x is called a *stationary point* of f. If $f'(x) \neq 0$, then x is not stationary and the set $\{d \in \mathbf{X}: f'(x)d < 0\}$ is a nonempty open half-space. An element d in this half-space is called a *descent vector* since condition (3) immediately implies that $f(x + td) < f(x)$ when t is positive and sufficiently small. If d is a ν-approximate gradient at a nonstationary point x, then according to (8)–(9),

$$f'(x)(-d) \leq -(1-\nu)\left|\left\|f'\right\|\right|^2 < 0.$$

Hence $-d$ is a descent vector. In particular, if d is a gradient at a nonstationary point x, then $-d$ is a *steepest descent vector* in the sense that

$$f'(x)(-d) \leq f'(x)v, \tag{10}$$

for all $v \in \mathbf{X}$ such that $\|v\| = \|d\|$.

Suppose that ν, μ_1, and μ_2 are fixed positive numbers, with $\nu \in (0, 1)$ and $\mu_2 \geq \mu_1 > 0$. At each $x \in \mathbf{X}$, let $G^\nu(x)$ denote the nonempty set of ν-approximate gradients for f at x and let $G(x)$ be a nonempty subset of the set of all multiples μd with $\mu \in [\mu_1, \mu_2]$ and $d \in G^\nu(x)$, i. e.,

$$\emptyset \neq G(x) \subset \bigcup_{\mu \in [\mu_1, \mu_2]} \mu G^\nu(x). \tag{11}$$

The corresponding set-valued mapping $G(\cdot)$ is referred to here as a *gradient-related set function* with parameters ν, μ_1 and μ_2. In the present development, a gradient-related iterative descent method consists of a gradient-related set function $G(\cdot)$, and a rule that selects a vector $d^k \in G(x^k)$ at each iterate x^k, and another rule that determines the *steplength* parameter $s^k \in (0, 1]$ in the recursion,

$$x^{k+1} = x^k - s^k d^k, \tag{12}$$

once d^k has been chosen. The sequences $\{x^k\}$ generated by this recursion are called gradient-related successive approximations. (For related formulations, see [3,4,10].) The convergence theorems described later in this article depend only on basic properties of gradient-related set functions and the steplengths s^k, hence the precise nature of the rule for selecting d^k in $G(x^k)$ is not important here. This rule may refer to prior iterates $\{x^i\}_{i \leq k}$, or may even be random in nature. There are also many alternative steplength rules that achieve sufficient reductions in f at each iteration in (12) and move the successive approximations x^k toward regions in the domain of f that are interesting in at least a local sense [3,4,10].

Descent Method Prototypes

When gradients of f exist and f attains its infima on lines in $\mathbf{X}$, the steepest descent and exact line minimization rules for d and s yield the prototype *steepest descent method*,

$$x^{k+1} = x^k - s^k d^k, \tag{13}$$

where

$$s^k \in \arg\min_{t>0} f(x^k - td^k) \tag{14}$$

and d^k is any solution of (6)–(7) for $x = x^k$. Note that the actual reduction in f achievable on a steepest descent half-line $\{y \in \mathbf{X}: \exists t > 0,\ y = x - td\}$ may be *smaller* than that attainable on other half-lines, since (10) merely refers to norm-dependent local directional rates of change for f at x. Thus the name of this method is somewhat misleading.

Newtonian descent algorithms also amount to special gradient-related descent methods near a certain type of *nonsingular local minimizer* x^*. These schemes

employ variants of the restricted line minimization steplength rule,

$$s^k \in \arg\min_{t\in(0,1]} f(x^k - td^k), \tag{15}$$

and replace the gradients d^k in a steepest descent iteration by descent vectors that approximate the Newton increment,

$$d^N(x^k) = f''(x^k)^{-1} f'(x^k). \tag{16}$$

Gradient-related descent vector approximations to $d^N(x^k)$ are generated in some neighborhood of x^* by various quasi-Newton auxiliary recursions, provided that the following (interdependent) nonsingularity conditions hold:

i) f is twice continuously (Fréchet) differentiable at x^*;
ii) $f''(x^*)$ satisfies the coercivity condition

$$(f''(x^*)v)v \geq c\,\|v\|^2$$

for some $c > 0$ and all $v \in \mathbf{X}$;
iii) A bounded inverse map $f''(x)^{-1}$ exists for all x sufficiently near x^*;
iv) $f''(\cdot)^{-1}$ is continuous at x^*.

Near a nonsingular local minimizer, the local convergence rates for Newtonian descent methods are generally much faster than the steepest descent convergence rate [8,10]. On the other hand, near singular local minimizers the Newton increments $d^N(x^k)$ and their quasi-Newton approximations are typically not confined to the image sets $G(x^k)$ of some gradient-related set function $G(\cdot)$, and may actually be *undefined* on continuous manifolds in $\mathbf{X}$ containing x^*. Under these circumstances, the unmodified Newtonian scaling principles can degrade or even destroy local convergence. In any case, the convergence properties of Newtonian descent methods near singular local minimizers x^* are not well-understood, and are likely to depend on the higher order structure of the singularity at x^*.

The Armijo Steplength Rule

The line minimization steplength rules in (14) and (15) can be very effective in special circumstances; however, they are more often difficult or impossible to implement, and are not intrinsically 'optimal' in any general sense when coupled with standard descent direction rules based on local models of f. By their very nature, such schemes do not anticipate the effect of current search direction and steplength decisions on the reductions achievable in f in later stages of the calculation. Therefore, over many iterations, the exact line minimization rule may well produce *smaller* total reductions in f than other much simpler steplength rules that merely aim for local reductions in f that are 'large enough' compared with $\|\|f'(x)\|\|$ at each iteration. A. Goldstein and L. Armijo proposed the first practical steplength rules of this kind in [1,8,9] for steepest descent and Newtonian descent methods in $\mathbf{R}^n$. These rules and other related schemes described in [10] and [4] are easily adapted to general gradient-related iterations. The present development focusses on the local convergence properties of the simple Armijo rule described below; however, with minor modifications, the theorems set forth here extend readily to the Goldstein rule and other similar line search formulations.

Let $G(\cdot)$ be a gradient-related set function with parameters ν, μ_1 and μ_2. Fix β in (0, 1) and δ in (0, 1), and for each x in $\mathbf{X}$ and d in $G(x)$ construct $s(x, d) \in (0, 1]$ with the *Armijo steplength rule*,

$$s(x, d) = \max t \tag{17}$$

subject to

$$t \in \{1, \beta, \beta^2, \ldots\}$$

and

$$f(x) - f(x - td) \geq \delta t f'(x)d.$$

When x is not stationary and $-d$ is any descent vector, the rule (17) admits *precisely one* associated steplength $s(x, d) \in (0, 1]$. This is true because β^k converges to zero as $k \to \infty$ and

$$\begin{aligned} &f(x) - f(x - td) \\ &= \delta f'(x)td + (1 - \delta)f'(x)td + o(t) \\ &\geq \delta f'(x)td \end{aligned}$$

for t positive and sufficiently small, in view of (3). When x is stationary, (17) yields $s(x, d) = 1$ trivially for every vector d.

Fixed Points

Descent methods based on gradient-related set functions and Armijo's rule generate sequences $\{x^k\}$ that sat-

isfy

$$x^{k+1} \in T(x^k), \quad k = 0, 1, \ldots, \tag{18}$$

where

$$T(x) \stackrel{\text{def}}{=} \{y\colon\ \exists d \in G(x),\ y = x - s(x,d)d\}\,. \tag{19}$$

The convergence theory outlined in the following sections addresses the behavior of all such Armijo gradient-related sequences near fixed points of the set-valued map $T(\cdot)\colon \mathbf{X} \to 2^{\mathbf{X}}$. The roots of this theory lie in Bertsekas' convergence proof for steepest descent iterates near nonsingular local minimizers in $\mathbf{R}^n$ [2], and subsequent modifications of this proof strategy for gradient projection methods and singular local minimizers in finite-dimensional or infinite-dimensional vector spaces with inner products [6,7]. For related nonlocal theories, see [10] and [4].

By definition, x^* is a *fixed point* of $T(\cdot)$ if and only if

$$x^* \in T(x^*).$$

Since Armijo's rule produces nonzero steplengths $s(x, d)$, it follows that x^* is a fixed point of $T(\cdot)$ if and only if x^* is a stationary point of f. More precisely,

Proposition 1 *Let $T(\cdot)$ be an Armijo gradient-related iteration map in (19). Then for all $x \in \boldsymbol{X}$,*

$$x \in T(x) \Leftrightarrow T(x) = \{x\} \Leftrightarrow G(x) = \{0\} \Leftrightarrow f'(x) = 0. \tag{20}$$

According to Proposition 1, any Armijo gradient-related sequence $\{x^k\}$ that intercepts a fixed point x^* of $T(\cdot)$ must terminate in x^*. Conversely, if $\{x^k\}$terminates in a vector x^*, then x^* is a fixed point of $T(\cdot)$, and hence a stationary point of f. On the other hand, Armijo gradient-related sequences that merely pass *near* some stationary point x^* may or may not converge to x^*.

Local Attractors: Necessary Conditions

A vector x^* is said to be a *local attractor* for an Armijo gradient-related iteration (18) if and only if there is a nonempty open ball,

$$B(x^*, \rho) = \{x \in \mathbf{X}\colon\ \|x - x^*\| < \rho\}$$

with center x^* and radius $\rho > 0$ such that every sequence $\{x^k\}$ which satisfies (18) and enters the ball $B(x^*, \rho)$ must converge to x^*, i. e.,

$$\exists l,\ x^l \in B(x^*, \rho) \Rightarrow \lim_{k\to\infty} \left\| x^k - x^* \right\| = 0. \tag{21}$$

With Proposition 1 and another rudimentary result for gradient-related set functions and Armijo steplengths, it is readily shown that a local attractor must be a *strict local minimizer* of f and an *isolated stationary point* of f.

Proposition 2 *Let $\nu \in (0, 1)$, $\mu_1 > 0$, and $\delta \in (0, 1)$ be fixed parameter values in the gradient-related set function $G(\cdot)$ and Armijo rule (17), and put $c_1 = \delta\,(1 - \nu)\mu_1 > 0$. Then for all $x \in \boldsymbol{X}$ and $d \in G(x)$,*

$$f(x) - f(x - s(x,d)d) \geq c_1 s(x,d) \left\| \left\| f'(x) \right\| \right\|^2. \tag{22}$$

Corollary 3 *Let $T(\cdot)$ be the Armijo gradient-related iteration map in (19). If $\{x^k\}$ is generated by the corresponding gradient-related iteration (18), then for all $k = 0, 1, \ldots,$*

$$f(x^{k+1}) \leq f(x^k) \tag{23}$$

and

$$f'(x^k) \neq 0 \Rightarrow f(x^{k+1}) < f(x^k). \tag{24}$$

Since f is continuous, the claimed necessary conditions for local attractors are now immediate consequences of Proposition 1 and Corollary 3.

Theorem 4 *A vector x^* is a local attractor for an Armijo gradient-related iteration (18) only if x^* is an isolated stationary point and a strict local minimizer of f, i. e., only if there is a nonempty open ball $B(x^*, \rho^*)$ that excludes every other stationary point $x \neq x^*$, and also excludes points $x \neq x^*$ at which $f(x) \leq f(x^*)$.*

The conclusion in Theorem 4 actually applies more generally to set-valued iteration maps $T(\cdot)$ prescribed by any steplength rule that guarantees the fixed-point characterization (20) and the descent property (23)–(24). On the other hand, related converse assertions are tied more closely to special properties of the Armijo rule and its variants, and to certain local uniform growth conditions on f and $\|\|f'(\cdot)\|\|$. If $\mathbf{X}$ is a finite-dimensional space, and x^* is a strict local minimizer

and an isolated stationary point, then the requisite uniform growth conditions automatically hold near x^* and the full converse of Theorem 4 can be proved. If $\mathbf{X}$ is an infinite-dimensional space, the growth conditions become hypotheses in a weaker but still nontrivial partial converse of Theorem 4. This is explained in greater detail below.

Local Attractors: Sufficient Conditions

If x^* is a strict local minimizer of f, then for some $\rho^* > 0$ and all x in the closed ball,

$$\overline{B}(x^*, \rho^*) = \{x \in \mathbf{X}\colon\ \|x - x^*\| \le \rho^*\},$$

the quantity $f(x) - f(x^*)$ is strictly positive when $x \neq x^*$. In finite-dimensional spaces, it is possible to say more. If $\dim \mathbf{X} < \infty$, then for each $t \in (0, \rho^*]$ the corresponding closed annulus,

$$A(t, \rho^*) = \{x\colon\ t \le \|x - x^*\| \le \rho^*\}, \tag{25}$$

is *compact*. Since the function $f(\cdot) - f(x^*)$ is continuous and positive in $A(t, \rho^*)$, it must attain a *positive minimum value* in this set, i. e.,

$$\alpha(t) \stackrel{\text{def}}{=} \min_{x \in A(t,\rho^*)} f(x) - f(x^*) > 0, \tag{26}$$

for each $t \in (0, \rho^*]$. Put $\alpha(0) = 0$ and note that for all t_1, t_2, $0 < t_1 < t_2 \le \rho^* \Rightarrow A(t_1, \rho^*) \supset A(t_2, \rho^*) \Rightarrow \alpha(t_1) \le \alpha(t_2)$. This establishes the following uniform growth property for strict local minimizers in finite-dimensional spaces.

Lemma 5 *Let $\mathbf{X}$ be a finite-dimensional normed vector space. If x^* is a strict local minimizer for f, then there is a positive number ρ^* and a positive definite nondecreasing real-valued function $\alpha(\cdot)$ on $[0, \rho^*]$ such that,*

$$f(x) - f(x^*) \ge \alpha(\|x - x^*\|), \tag{27}$$

for all $x \in \overline{B}(x^*, \rho^*)$.

In infinite-dimensional spaces, the uniform growth condition (27) need not hold at every strict local minimizer; however, when this condition is satisfied, the minimizer x^* has a crucial stability property for gradient-related descent methods. More specifically, suppose that (27) holds and $T(\cdot)$ is an Armijo iteration map (19) with associated parameter $\mu_2 > 0$. Since descent directions can not exist at a local minimizer, the vector x^* must be a stationary point. Fix $\epsilon \in (0, \rho^*]$ and note that since $f'(\cdot)$ is continuous and $f'(x^*) = 0$, there is a $\tau_\epsilon \in (0, \epsilon]$ for which

$$\|x - x^*\| + \mu_2 \left|\left\|f'(x)\right\|\right| < \epsilon \tag{28}$$

for all $x \in B(x^*, \tau_\epsilon)$. Now construct the corresponding set,

$$I(\epsilon) = \{x \in B(x^*, \epsilon)\colon\ f(x) - f(x^*) < \alpha(\tau_\epsilon)\}. \tag{29}$$

By Proposition 2, the simple descent property,

$$f(x - s(x, d)d) \le f(x) \tag{30}$$

holds for all x and all $d \in G(x)$, hence the restriction (28) and the properties of $\alpha(\cdot)$ insure that $I(\epsilon)$ is an *invariant set* for $T(\cdot)$, i. e., $T(x) \subset I(\epsilon)$ for all $x \in I(\epsilon)$. Moreover, since f is continuous, the minimizer x^* is clearly an *interior point* of the set $I(\epsilon)$, and this proves the following stability lemma for Armijo gradient-related iterations (or indeed, any gradient-related method with the descent property (30)).

Lemma 6 *Suppose that the uniform growth condition (27) holds near a local minimizer x^* for f. Let $T(\cdot)$ be an Armijo gradient-related iteration map in (19). Then for every $\epsilon > 0$ there is a corresponding $\rho \in (0, \epsilon]$ such that for all sequences $\{x^k\}$ satisfying (18), and all indices l,*

$$x^l \in B(x^*, \rho) \Rightarrow \forall k \ge l\ x^k \in B(x^*, \epsilon). \tag{31}$$

According to Lemma 6, the uniform growth condition (27) guarantees that an Armijo gradient-related sequence $\{x^k\}$ will remain in any specified arbitrarily small open ball $B(x^*, \epsilon)$ provided $\{x^k\}$ enters a sufficiently small sub-ball of $B(x^*, \epsilon)$. This property alone does not imply that $\{x^k\}$ *converges* to x^*; however, it is an essential ingredient in the local convergence proof outlined below. This proof requires two additional technical estimates for the Armijo rule and gradient-related set functions, a local uniform growth condition for $\left|\left\|f'(\cdot)\right\|\right|$ analogous to (27), and a local uniform continuity hypothesis on $f'(\cdot)$. The first pair of estimates are straightforward consequences of the Armijo rule and the one-dimensional mean value theorem. The last two requirements are automatically satisfied in finite-dimensional spaces, once again because closed bounded sets are compact in these spaces.

Proposition 7 *Let $\nu \in (0, 1)$, $\mu_2 > 0$, $\beta \in (0, 1)$, and $\delta \in (0, 1)$ be fixed parameter values in the gradient-related set function $G(\cdot)$ and Armijo rule (17), and put $c_2 = \delta(1-\nu)\mu_2^{-1} > 0$. Then for all $x \in \boldsymbol{X}$ and $d \in G(x)$,*

$$f(x) - f(x - s(x, d)d) \geq c_2 s(x, d)^2 \|d\|^2 . \tag{32}$$

Moreover, if $s(x, d) < 1$ and $c_3 = (1 - \delta)(1 - \nu)$, then there is a vector ξ in the line segment joining x to $x - \beta^{-1}s(x, d)d$ such that

$$\left|\left\|f'(\xi) - f'(x)\right\|\right| \geq c_3 \left|\left\|f'(x)\right\|\right| \tag{33}$$

and

$$\|\xi - x\| \leq \beta^{-1} s(x, d) \|d\| . \tag{34}$$

Lemma 8 *Let $\boldsymbol{X}$ be a finite-dimensional normed vector space. If x^* is an isolated stationary point for f, then there is a positive number ρ^* and a positive definite nondecreasing real-valued function $\beta(\cdot)$ on $[0, \rho^*]$ such that,*

$$\left|\left\|f'(x)\right\|\right| \geq \beta(\|x - x^*\|), \tag{35}$$

for all $x \in \overline{B}(x^*, \rho^*)$.

The proof of Lemma 8 is similar to the proof of Lemma 5.

Now suppose that the growth conditions (27) and (35) both hold in the ball $\overline{B}(x^*, \rho^*)$, and that $f'(\cdot)$ is uniformly continuous in this ball. By Lemma 6, there is a positive number $\rho \in (0, \rho^*/2]$ such that every sequence $\{x^k\}$ which satisfies (18) and enters the ball $B(x^*, \rho)$, thereafter remains in the larger ball $B(x^*, \rho^*/2)$. But if $\{x^k\}$ is eventually confined to the ball $B(x^*, \rho^*/2)$, then the mean value theorem insures that the nonincreasing real sequence $\{f(x^k)\}$ is bounded below and therefore converges to some finite limit. In this case, the differences $f(x^k) - f(x^{k+1})$ converge to zero and Propositions 2 and 7 therefore yield,

$$\lim_{k\to\infty} s(x^k, d^k) \left|\left\|f'(x^k)\right\|\right|^2 = 0, \tag{36}$$

and

$$\lim_{k\to\infty} s(x^k, d^k) \left\|d^k\right\| = 0, \tag{37}$$

where $d^k \in G(x^k)$ and $s(x^k, d^k)\, d^k = x^{k+1} - x^k$ for all k. It follows easily from the remainder of Proposition 7 and the growth condition (35) that

$$\lim_{k\to\infty} \left|\left\|f'(x^k)\right\|\right| = 0 \tag{38}$$

and therefore

$$\lim_{k\to\infty} \left\|x^k - x^*\right\| = 0. \tag{39}$$

To see that (38) must hold, construct the index sets, $\psi = \{k\colon s(x^k, d^k) = 1\}$ and $\phi = \{k\colon s(x^k, d^k) < 1\}$. If ψ is an infinite set, then,

$$\lim_{\substack{k\in\psi \\ k\to\infty}} \left|\left\|f'(x^k)\right\|\right| = 0,$$

by (36). On the other hand, if ϕ is an infinite set, then

$$\lim_{\substack{k\in\psi \\ k\to\infty}} \left|\left\|f'(x^k)\right\|\right| = 0,$$

by (37), (33), (34), and the local uniform continuity of $f'(\cdot)$. This establishes (38) and proves the following local convergence results.

Theorem 9 *If the uniform growth conditions (27) and (35) hold simultaneously in the closed ball $\overline{B}(x^*, \rho^*)$ for some $\rho^* > 0$, and if $f'(\cdot)$ is uniformly continuous in $\overline{B}(x^*, \rho^*)$ then x^* is a local attractor for Armijo gradient-related iterations (18).*

Corollary 10 *If $\boldsymbol{X}$ is a finite-dimensional normed vector space and x^* is a strict local minimizer and an isolated stationary point for f, then x^* is a local attractor for Armijo gradient-related iterations (18).*

Nonsingular Attractors

The nonsingularity conditions i) and ii) and Taylor's formula imply that in some neighborhood of x^*, the objective function f is convex and satisfies the local growth condition (27) with

$$\alpha(t) = a\, t^2 \tag{40}$$

for some $a > 0$. But if f is locally convex near x^*, then

$$\begin{aligned} f(x) - f(x^*) &\leq f'(x)(x - x^*) \\ &\leq \left|\left\|f'(x)\right\|\right| \|x - x^*\| \end{aligned} \tag{41}$$

near x^*, and therefore (27) and (40) imply (35) with

$$\beta(t) = a\, t. \tag{42}$$

These observations and Theorem 9 immediately yield the following extension of the convergence result in [2] for steepest descent processes in $\mathbf{R}^n$.

Corollary 11 *Every nonsingular local minimizer x^* is a local attractor for Armijo gradient-related iterations (18).*

Singular Attractors and Local Convexity

The growth condition (27) alone does not imply local convexity of f, or condition (35), or the local attractor property. In fact, (27) can hold even if x^* is the limit of some infinite sequence of local minimizers for f. This is readily demonstrated by the following simple function $F: \mathbf{R}^1 \to \mathbf{R}^1$:

$$F(x) = x^2\left[\sqrt{2} - \sin\left(\frac{5\pi}{6} - \sqrt{3}\ln x^2\right)\right]. \qquad (43)$$

This function has a strict absolute minimizer at $x^* = 0$, with

$$(\sqrt{2}-1)x^2 \le F(x) \le (\sqrt{2}+1)x^2$$

for all $x \in \mathbf{R}^1$. However, F also has infinitely many (nonsingular) local minimizers,

$$x_m^{\pm} = \pm\exp\left[\frac{(1-8m)\pi}{8\sqrt{3}}\right]$$

for $m = 1, 2, \ldots$, and these local minimizers *accumulate* at 0. Since each $x_m^{\pm}$ is a stationary point and not an absolute minimizer, it follows that F is not convex in any neighborhood of the absolute minimizer at $x^* = 0$, that (35) cannot hold at x^*, and that x^* is not a local attractor for gradient-related descent processes. Evidently, $x^* = 0$ is a singular minimizer for F; in fact, $F''(x)$ does not exist at $x = 0$. (Apart from a minor alteration in one of its constants, (43) is taken directly from [6, Example 1.1]. The erroneous constant in [6] was kindly called to the author's attention by D. Bertsekas.)

The growth conditions (27) and (35) *together* still do not imply convexity of f near x^*, and indeed f may not be convex in any neighborhood of a singular local attractor. This is shown by another function $F: \mathbf{R}^2 \to \mathbf{R}^1$ from [6, Example 1.2], viz.

$$F(x) = x_1^2 - 1.98x_1\|x\|^2 + \|x\|^4, \qquad (44)$$

where $x = (x_1, x_2)$ and $\|\cdot\|$ is the Euclidean norm in $\mathbf{R}^2$. This function has a singular absolute minimizer at $x^* = 0$, and $F(x)$ and $\|\|F'(x)\|\|$ grow like $\|x\|^4$ and $\|x\|^3$, respectively, near 0. On the other hand, since every neighborhood of 0 contains points x where $F'(x)(x-0)$ is *negative*, it follows that F is not convex (or even pseudoconvex) near 0. Nevertheless, $x^* = 0$ is a local attractor for Armijo gradient-related iterations, according to Corollary 10.

Although f need not be convex near a singular local attractor x^*, there are many instances where some sort of local convexity property is observed. (The function $f(x) = x^4$ provides a simple illustration.) If the local pseudoconvexity condition,

$$\kappa(f(x) - f(x^*)) \le f'(x)(x - x^*), \qquad (45)$$

is satisfied for some $\kappa > 0$ and all x in the ball $\overline{B}(x^*, \rho^*)$, then

$$\kappa(f(x) - f(x^*)) \le \|\|f'(x)\|\| \, \|x - x^*\|$$

near x^*, and condition (35) follows at once from (27), with

$$\beta(t) = \kappa(\rho^*)^{-1}\alpha(t)$$

for all $t \in [0, \rho^*]$. These considerations immediately yield two additional corollaries of Theorem 9.

Corollary 12 *Suppose that the uniform growth condition (27) holds in the closed ball $\overline{B}(x^*, \rho^*)$ for some $\rho^* > 0$. In addition, suppose that in $\overline{B}(x^*, \rho^*)$, $f'(\cdot)$ is uniformly continuous and f satisfies the pseudoconvexity condition (45). Then x^* is a local attractor for Armijo gradient-related iterations (18).*

Corollary 13 *If $\mathbf{X}$ is a finite-dimensional normed vector space, if x^* is a strict local minimizer for f, and if f satisfies the pseudoconvexity condition (45), then x^* is a local attractor for Armijo gradient-related iterations (18).*

Local Convexity and Convergence Rates

A local version of the convergence rate proof strategy in [5] also works in the present setting when $f'(\cdot)$ is locally Lipschitz continuous and f satisfies the pseudoconvexity condition (45) and the growth condition (27) near x^*. Under these circumstances, the worst-case convergence rate estimate,

$$f(x^k) - f(x^*) = O(k^{-1}), \qquad (46)$$

can be proved for Armijo gradient-related sequences $\{x^k\}$ that pass sufficiently near x^*. More refined order estimates are possible if the first two hypotheses hold and

$$f(x) - f(x^*) \ge a\|x - x^*\|^r \qquad (47)$$

for some $a > 0$ and $r \in (1, \infty)$, and all $x \in \overline{B}(x^*, \rho^*)$. In such cases, it it can be shown that

$$f(x^k) - f(x^*) = O(k^{-\frac{r}{(r-2)}}) \tag{48}$$

for $r \in (2, \infty)$, and

$$\exists \lambda \in [0, 1) \quad f(x^k) - f(x^*) = O(\lambda^k) \tag{49}$$

for $r \in (1, 2]$. (The latter estimate is comparable to the basic geometric convergence rate theorem for steepest descent iterates near nonsingular local minimizers [2].) The proof strategy in [5] can also produce still more precise local convergence rate estimates that relate the constants implicit in the order estimates (48) and (49) to local Lipschitz constants for $f'(\cdot)$ and parameters in the gradient-related set functions $G(\cdot)$, the growth condition (47), the pseudoconvexity condition (45), and the Armijo steplength rule (17).

In the absence of local convexity assumptions, it is harder to establish analogous asymptotic convergence rate theorems; however, the analysis in [6] and [7] has established $O(k^{-2})$ rate estimates for Hilbert space steepest descent iterations and a class of nonlinear functions f that contains the example (44).

Concluding Remarks

In a finite-dimensional space any two norms are equivalent and it can be seen that the gradient-related property and the local attractor property are therefore norm-invariant qualitative features of set-valued maps $G(\cdot)$: $\mathbf{X} \to 2^{\mathbf{X}}$ and local minimizers x^*. On the other hand, even in finite-dimensional spaces, the Lipschitz constants, growth rate constants, and gradient-related set function parameters in the present formulation are *not* norm-invariant, and this is reflected in norm-dependent convergence rates and norm-dependent size and shape parameters for the domains that are sent to a local attractor x^* by gradient-related iterations. These facts have potentially important computational manifestations when gradient-related methods are applied to large scale finite-dimensional problems that approximate some limiting problem in an infinite-dimensional space. Note that infinite-dimensional spaces can support multiple nonequivalent norms, and a set-valued function $G(\cdot)$ that is gradient-related in one norm need not be gradient-related relative to some other nonequivalent norm. Similarly, the local attractor property for a minimizer x^*, and indeed local optimality itself, are also typically norm-dependent at this level of generality.

See also

- ▶ Conjugate-gradient Methods
- ▶ Large Scale Trust Region Problems
- ▶ Nonlinear Least Squares: Newton-type Methods
- ▶ Nonlinear Least Squares: Trust Region Methods

References

1. Armijo L (1966) Minimization of functions having continuous partial derivatives. Pacific J Math 16:1–3
2. Bertsekas DP (1982) Constrained optimization and Lagrange multiplier methods. Acad. Press, New York
3. Bertsekas DP (1995) Nonlinear programming. Athena Sci., Belmont, MA
4. Daniel JW (1971) Approximate minimization of functionals. Prentice-Hall, Englewood Cliffs, NJ
5. Dunn JC (1981) Global and asymptotic convergence rate estimates for a class of projected gradient processes. SIAM J Control Optim 12:659–674
6. Dunn JC (1987) Asymptotic decay rates from the growth properties of Liapunov functions near singular attractors. J Math Anal Appl 125:6–21
7. Dunn JC (1987) On the convergence of projected gradient processes to singular critical points. J Optim Th Appl 55:203–216
8. Goldstein A (1965) On Newton's method. Numer Math 7:391–393
9. Goldstein A (1965) On steepest descent. SIAM J Control 3:147–151
10. Ortega JM, Rheinboldt WC (1970) Iterative solution of nonlinear equations in several variables. Acad. Press, New York

Location Routing Problem

YANNIS MARINAKIS
Department of Production Engineering and Management, Decision Support Systems Laboratory, Technical University of Crete, Chania, Greece

MSC2000: 90B06, 90B80

Article Outline

Introduction

In the last few years, the need for an integrated logistic system has become a primary objective of every company manager. Managers recognize that there is a strong relation between the location of facilities, the allocation of suppliers, vehicles, and customers to the facilities, and the design of routes around the facilities. In a **location routing problem (LRP)**, the optimal number, the capacity, and the location of facilities are determined, and the optimal set of vehicle routes from each facility is also sought.

In most location models, it is assumed that the customers are served directly from the facilities being located. Each customer is served on his or her own route. In many cases, however, customers are not served individually from the facilities. Rather, customers are consolidated into routes that may contain many customers. One of the reasons for the added difficulty in solving these problems is that there are far more decisions that need to be made by the model. These decisions include:

- How many facilities to locate,
- Where the facilities should be,
- Which customers to assign to which depots,
- Which customers to assign to which routes,
- In what order customers should be served on each route.

In the LRP, a number of facilities are located among candidate sites and delivery routes are established for a set of users in such a way that the total system cost is minimized. As Perl and Daskin [51] pointed out, LRPs involve three interrelated, fundamental decisions: where to locate facilities, how to allocate customers to facilities, and how to route vehicles to serve customers.

The difference between the LRP and the classic vehicle routing problem is that not only routing must be designed but the optimal depot location must be simultaneously determined as well. The main difference between the LRP and the classical location-allocation problem is that, once the facility is located, the former requires a visitation of customers through tours while the latter assumes that the customer will be visited from the vehicle directly, and then the vehicle will return to the facility without serving any other customer ([47]). In general terms, the combined location routing model solves the joint problem of determining the optimal number, capacity, and location of facilities serving more than one customer and finding the optimal set of vehicle routes. In the LRP, the distribution cost is decreased due to the assignment of the customers to vehicles while the main objective is the design of the appropriate routes of the vehicles.

Variants of the Location Routing Problem

Laporte et al. [39] considered three variants of LRPs, including (1) capacity-constrained vehicle routing problems, (2) cost-constrained vehicle routing problems, and (3) cost-constrained location routing problems. The authors examined multidepot, asymmetrical problems and developed an optimal solution procedure that enables them to solve problems with up to 80 nodes. Chan et al. [11] solved a multidepot, multivehicle location routing problem with stochastically processed demands, which are defined as demands that are generated upon completing site-specific service on their predecessors. Min et al. [47] synthesized the past research and suggested some future research directions for the LRP. An extended recent literature review is included in the survey paper published by Nagy and Salhi [48]. They proposed a classification scheme and looked at a number of problem variants. The most important exact and heuristic algorithms were presented and analyzed in this survey paper.

Exact Algorithms for the Solution of the Location Routing Problem

A number of exact algorithms for the problem was presented by Laporte et al. [38]. Applications and formulations and exact and approximation algorithms for LRPs under capacity and maximum cost restrictions are studied in the survey of Laporte [34]. Nonlinear programming exact algorithms for the solution of the LRP have been proposed in [20,61]. Dynamic programming exact algorithms for the solution of the LRP have been proposed in [5]. Integer programming exact al-

gorithms for the solution of the LRP have been proposed in [35,37,46]. Mixed integer goal programming exact algorithms for the solution of the LRP have been proposed in [65]. Two branching strategies have been proposed in [36]. An iterative exact procedure has been proposed in [9]. A branch-and-bound technique on the LP relaxation has been proposed in [17].

Heuristic Algorithms for the Solution of the Location Routing Problem

The LRP is very difficult to solve using exact algorithms, especially if the number of customers or the candidate for location facilities is very large due to the fact that this problem belongs to the category of ***NP-hard problems***, i. e. there are no known polynomial-time algorithms that can be used to solve them. Madsen [43] presented a survey of heuristic methods. Christofides and Eilon [16] were the first to consider the problem of locating a depot from which customers are served by tours rather than individual trips. They proposed an approximation algorithm for the solution of the problem. Watson-Gandy and Dohrn [63] proposed an algorithm where the problem is solved by transforming its location part into an ordinary location problem using the Christofides–Eilon approximation algorithm. The routing part of the algorithm is solved using the Clarke and Wright algorithm. Jacobsen and Madsen [31] proposed three algorithms. The first is called a tree-tour heuristic. The second is called ALA–SAV and is a three-phase heuristic, where in the first phase a location–allocation problem is solved and in the second and third phases a Clarke and Wright heuristic is applied for solving the problem. Finally, the third proposed algorithm is called SAV–DROP and is a heuristic algorithm that combines the Clarke–Wright method and the DROP algorithm. A two-phase heuristic is presented in [4], where in the first phase the set of open plants is determined and a priori routes are considered, while in the second phase the routes are optimized. Other two-phase heuristics have been proposed in [7,12,13,30,33,42,49,50,58]. Cluster analysis algorithms are presented in [6,18,60]. Iterative approaches have been proposed by [27,59]. Min ([46]) considered a two-level location–allocation problem of terminals to customer clusters and supply sources using a hierarchical approach consisting of both exact and heuristic procedures. Insertion methods have been proposed in [15]. A partitioning heuristic algorithm is proposed in [35], and a sweep heuristic is proposed in [21].

Metaheuristic Algorithms for the Solution of the Location Routing Problem

Several metaheuristic algorithms have been proposed for the solution of the LRP. In what follows, an analytical presentation of these algorithms is given.

- **Tabu search (TS)** was introduced by Glover [22,23] as a general iterative metaheuristic for solving combinatorial optimization problems. Computational experience has shown that TS is a well-established approximation technique that can compete with almost all known techniques and that, by its flexibility, can beat many classic procedures. It is a form of local neighbor search. Each solution S has an associated set of neighbors $N(S)$. A solution $S' \in N(S)$ can be reached from S by an operation called a *move*. TS can be viewed as an iterative technique that explores a set of problem solutions by repeatedly making moves from one solution S to another solution S' located in the neighborhood $N(S)$ of S [24]. TS moves from a solution to its best admissible neighbor, even if this causes the objective function to deteriorate. To avoid cycling, solutions that have been recently explored are declared *forbidden or tabu* for a number of iterations. The tabu status of a solution is overridden when certain criteria (*aspiration criteria*) are satisfied. Sometimes, *intensification* and *diversification* strategies are used to improve the search. In the first case, the search is accentuated in the promising regions of the feasible domain. In the second case, an attempt is made to consider solutions in a broad area of the search space. Tuzun and Burke [62] proposed a two-phase tabu search architecture for the solution of the LRP. TS algorithms for the LRP are also presented in [10,14,41,45,57].
- **Simulated annealing (SA)** [1,3,32] plays a special role within local search for two reasons. First, SA appears to be quite successful when applied to a broad range of practical problems. Second, some threshold accepting algorithms such as SA have a stochastic component, which facilitates a theoretical analysis of their asymptotic convergence. SA [2] algorithms are stochastic algorithms that allow random

uphill jumps in a controlled fashion in order to provide possible escapes from poor local optima. Gradually the probability allowing the objective function value to increase is lowered until no more transformations are possible. SA owes its name to an analogy with the annealing process in condensed-matter physics, where a solid is heated to a maximum temperature at which all particles of the solid randomly arrange themselves in the liquid phase, followed by cooling through careful and slow reduction of the temperature until the liquid is frozen with the particles arranged in a highly structured lattice and minimal system energy. This ground state is reachable only if the maximum temperature is sufficiently high and the cooling sufficiently slow. Otherwise a metastable state is reached. The metastable state is also reached with a process known as quenching, in which the temperature is instantaneously lowered. Its predecessor is the so-called Metropolis filter. Wu et al. [64] proposed an algorithm that divides the original problem into two subproblems, i. e., the location–allocation problem and the general vehicle routing problem, respectively. Each subproblem is, then, solved in a sequential and iterative manner by the SA algorithm embedded in the general framework for the problem-solving procedure. SA algorithms for the LRP are presented in [8,40,41].

- **Greedy randomized adaptive search procedure (GRASP)** [56] is an iterative two-phase search method that has gained considerable popularity in combinatorial optimization. Each iteration consists of two phases, a construction phase and a local search procedure. In the construction phase, a randomized greedy function is used to build up an initial solution. This randomized technique provides a feasible solution within each iteration. This solution is then exposed for improvement attempts in the local search phase. The final result is simply the best solution found over all iterations. Prins et al. [52] proposed a GRASP with a path-relinking phase for the solution of the capacitated location routing problem.
- **Genetic algorithms (GAs)** are search procedures based on the mechanics of natural selection and natural genetics. The first GA was developed by John H. Holland in the 1960s to allow computers to evolve solutions to difficult search and combinatorial problems such as function optimization and machine learning [28]. Genetic algorithms offer a particularly attractive approach to problems like location routing problems since they are generally quite effective for the rapid global search of large, nonlinear, and poorly understood spaces. Moreover, GAs are very effective in solving large-scale problems. GAs [25] mimic the evolution process in nature. They are based on an imitation of the biological process in which new and better populations among different species are developed during evolution. Thus, unlike most standard heuristics, GAs use information about a population of solutions, called individuals, when they search for better solutions. A GA is a stochastic iterative procedure that maintains the population size constant in each iteration, called a generation. Their basic operation is the mating of two solutions to form a new solution. To form a new population, a binary operator called a crossover and a unary operator called a mutation are applied [54,55]. Crossover takes two individuals, called parents, and produces two new individuals, called offspring, by swapping parts of the parents. Marinakis and Marinaki [44] proposed a bilevel GA for a real-life LRP. A new formulation based on bilevel programming was proposed. Based on the fact that in the LRP decisions are made at a strategic level and at an operational level, we formulate the problem in such a way that in the first level, the decisions of the strategic level are made, namely, the top manager finds the optimal location of the facilities, while in the second level, the operational-level decisions are made, namely, the operational manager finds the optimal routing of vehicles. Other evolutionary approaches for the solution of the LRP have been proposed in [29,53].
- **Variable neighborhood search (VNS)** is a metaheuristic for solving combinatorial optimization problems whose basic idea is systematic change of a neighborhood within a local search [26]. VNS algorithms for the LRP are presented in [45].
- The **ant colony optimization (ACO)** metaheuristic is a relatively new technique for solving combinatorial optimization problems (COPs). Based strongly on the ant system (AS) metaheuristic developed by Dorigo, Maniezzo, and Colorni [19], ACO is derived

from the foraging behavior of real ants in nature. The main idea of ACO is to model the problem as the search for a minimum cost path in a graph. Artificial ants walk through this graph looking for good paths. Each ant has a rather simple behavior so that it will typically only find rather poor-quality paths on its own. Better paths are found as the emergent result of the global cooperation among ants in the colony. An ACO algorithm consists of a number of cycles (iterations) of solution construction. During each iteration a number of ants (which is a parameter) construct complete solutions using heuristic information and the collected experiences of previous groups of ants. These collected experiences are represented by a digital analog of trail pheromone that is deposited on the constituent elements of a solution. Small quantities are deposited during the construction phase while larger amounts are deposited at the end of each iteration in proportion to solution quality. Pheromone can be deposited on the components and/or the connections used in a solution depending on the problem. ACO algorithms for the LRP are presented in [8].

References

1. Aarts E, Korst J (1989) Simulated Annealing and Boltzmann Machines – A Stochastic Approach to Combinatorial Optimization and Neural Computing. John Wiley and Sons, Chichester
2. Aarts E, Korst J, Van Laarhoven P (1997) Simulated Annealing. In: Aarts E, Lenstra JK (eds) Local Search in Combinatorial Optimization. Wiley, Chichester, pp 91–120
3. Aarts E, Ten Eikelder HMM (2002) Simulated Annealing. In: Pardalos PM, Resende MGC (eds) Handbook of Applied Optimization. Oxford University Press, New York, pp 209–221
4. Albareda-Sambola M, Diaz JA, Fernandez E (2005) A Compact Model and Tight Bounds for a Combined Location-Routing Problem. Comput Oper Res 32(3):407–428
5. Averbakh I, Berman O (1994) Routing and Location-Routing p-Delivery Men Problems on a Path. Transp Sci 28(2):162–166
6. Barreto S, Ferreira C, Paixao J, Santos BS (2007) Using Clustering Analysis in a Capacitated Location-Routing Problem. Eur J Oper Res 179(3):968–977
7. Bookbinder JH, Reece KE (1988) Vehicle Routing Considerations in Distribution System Design. Eur J Oper Res, 37:204-213
8. Bouhafs L, Hajjam A, Koukam A (2006) A Combination of Simulated Annealing and Ant Colony System for the Capacitated Location-Routing Problem. Knowl-Based Intelligent Inf Eng Syst, LNCS 4251:409–416
9. Burness RC, White JA (1976) The Traveling Salesman Location Problem. Transp Sci 10(4):348–360
10. Caballero R, Gonzalez M, Guerrero FM, Molina J, Paralera C (2007) Solving a Multiobjective Location Routing Problem with a Metaheuristic Based on Tabu Search. Application to a Real Case in Andalusia. Eur J Oper Res 177(3):1751–1763
11. Chan Y, Carter WB, Burnes MB (2001) A Multiple-Depot, Multiple-Vehicle, Location-Routing Problem with Stochastically Processed Demands. Comput Oper Res 28:803–826
12. Cappanera P, Gallo G, Maffioli F (2003) Discrete Facility Location and Routing of Obnoxious Activities. Discret Appl Math 133(1–3):3–28
13. Chan Y, Baker SF (2005) The Multiple Depot, Multiple Traveling Salesmen Facility-Location Problem: Vehicle Range, Service Frequency, Heuristic Implementations. Math Comput Model 41(8-9):1035–1053
14. Chiang WC, Russell RA (2004) Integrating Purchasing and Routing in a Propane Gas Supply Chain. Eur J Oper Res 154(3):710–729
15. Chien TW (1993) Heuristic Procedures for Practical-sized Uncapacitated Location-Capacitated Routing Problems. Decis Sci 24(5):995–1021
16. Christofides N, Eilon S (1969) Expected Distances for Distribution Problems. Oper Res Q 20:437–443
17. Daskin MS (1987) Location, Dispatching, Routing Models for Emergency Services with Stochastic Travel Times. In: Ghosh A, Rushton G (eds) Spatial Analysis and Location-Allocation Models. Von Nostrand Reinhold Company, NY, pp 224–265
18. Dondo R, Cerda J (2007) A Cluster-Based Optimization Approach for the Multi-Depot Heterogeneous Fleet Vehicle Routing Problem with Time Windows. Eur J Oper Res 176(3):1478–1507
19. Dorigo M, Stutzle T (2004) Ant Colony Optimization, A Bradford Book. MIT Press Cambridge, MA, London
20. Ghosh JK, Sinha SB, Acharya D (1981) A Generalized Reduced Gradient Based Approach to Round-trip Location Problem. In: Jaiswal NK (eds) Scientific Management of Transport Systems. Amsterdam, Holland, pp 209–213
21. Gillett B, Johnson J (1976) Multi-Terminal Vehicle-Dispatch Algorithm. Omega 4(6):711–718
22. Glover F (1989) Tabu Search I. ORSA J Compu 1(3):190–206
23. Glover F (1990) Tabu Search II. ORSA J Compu 2(1):4–32
24. Glover F, Laguna M, Taillard E, de Werra D (eds) (1993) Tabu Search. JC Baltzer AG, Science Publishers, Basel
25. Goldberg DE (1989) Genetic Algorithms in Search, Optimization, Machine Learning. Addison-Wesley, Reading Massachussets
26. Hansen P, Mladenovic N (2001) Variable Neighborhood Search: Principles and Applications. Eur J Oper Res 130:449–467

27. Hansen PH, Hegedahl B, Hjortkajaer S, Obel B (1994) A Heuristic Solution to the Warehouse Location-Routing Problem. Eur J Oper Res 76:111–127
28. Holland JH (1975) Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor, MI
29. Hwang HS (2002) Design of Supply-Chain Logistics System Considering Service Level. Comput Ind Eng 43(1–2):283–297
30. Jacobsen SK, Madsen OBG (1978) On the Location of Transfer Points in a Two-Level Newspaper Delivery System – A Case Study. Presented at The International Symposium on Locational Decisions. The Institute of Mathematical Statistics and Operations Research, The Technical University of Denmark, Lyngbyn Denmark, pp 24–28
31. Jacobsen SK, Madsen OBG (1980) A Comparative Study of Heuristics for Two Level Routing Location Problem. Eur J Oper Res 5:378–387
32. Kirkpatrick S, Gelatt CD, Vecchi MP (1982) Optimization by Simulated Annealing. Science 220:671–680
33. Laoutaris N, Zissimopoulos V, Stavrakakis I (2005) On the Optimization of Storage Capacity Allocation for Content Distribution. Comput Netw 47(3):409–428
34. Laporte G (1988) Location Routing Problems. In: Golden BL et al (eds) Vehicle Routing: Methods and Studies. North-Holland, Amsterdam, pp 163–198
35. Laporte G, Dejax PJ (1989) Dynamic Location-Routing Problems. J Oper Res Soc 40(5):471–482
36. Laporte G, Nobert Y (1981) An Exact Algorithm for Minimizing Routing and Operating Costs in Depot Location. Eur J Oper Res 6:224–226
37. Laporte G, Nobert Y, Arpin D (1986) An Exact Algorithm for Solving a Capacitated Location-Routing Problem. Ann Oper Res 6:293–310
38. Laporte G, Nobert Y, Pelletier P (1983) Hamiltonian Location Problems. Eur J Oper Res 12(1):82–89
39. Laporte G, Nobert Y, Taillefer S (1988) Solving a Family of Multi-depot Vehicle Routing and Location Routing Problems. Transp Sci 22:161–172
40. Lin CKY, Chow CK, Chen A (2002) A Location-Routing-Loading Problem for Bill Delivery Services. Comput Ind Eng 43(1–2):5–25
41. Lin CKY, Kwok RCW (2006) Multi-Objective Metaheuristics for a Location-Routing Problem with Multiple Use of Vehicles on Real Data and Simulated Data. Eur J Oper Res 175(3):1833–1849
42. Liu SC, Lee SB (2003) A Two-Phase Heuristic Method for the Multi-Depot Location Routing Problem Taking Inventory Control Decisions Into Consideration. Int J Adv Manuf Technol 22(11-12):941–950
43. Madsen OBG (1983) Methods for Solving Combined Two Level Location Routing Problems of Realistic Dimension. Eur J Oper Res 12(3):295–301
44. Marinakis Y, Marinaki M (2008) A Bilevel Genetic Algorithm for a Real Life Location Routing Problem. Int J Logist 11(1):49–65
45. Melechovsky J, Prins C, Calvo RW (2005) A Metaheuristic to Solve a Location-Routing Problem with Non-Linear Costs. J Heurist 11(5-6):375–391
46. Min H (1996) Consolidation Terminal Location-Allocation and Consolidated Routing Problems. J Bus Logist 17(2): 235–263
47. Min H, Jayaraman V, Srivastava R (1998) Combined Location-Routing Problems: A Synthesis and Future Research Directions. Eur J Oper Res 108:1–15
48. Nagy G, Salhi S (2007) Location-Routing: Issues, Models and Methods. Eur J Oper Res 177:649–672
49. Nambiar JM, Gelders LF, Van Wassenhove LN (1981) A Large Scale Location-Allocation Problem in the Natural Rubber Industry. Eur J Oper Res 6:183–189
50. Perl J, Daskin MS (1984) A Unified Warehouse Location-Routing Methodology. J Bus Logist 5(1):92–111
51. Perl J, Daskin MS (1985) A Warehouse Location Routing Model. Transp Res B 19:381–396
52. Prins C, Prodhon C, Calvo RW (2006) Solving the Capacitated Location-Routing Problem by a GRASP Complemented by a Learning Process and a Path Relinking, 4OR 4:221–238
53. Prins C, Prodhon C, Calvo RW (2006) A Memetic Algorithm with Population Management (MA|PM) for the Capacitated Location-Routing Problem. Evol Comput Combinatorial Optim, LNCS 3906:183–194
54. Reeves CR (1995) Genetic Algorithms. In: Reeves CR (eds) Modern Heuristic Techniques for Combinatorial Problems. McGraw-Hill, London, pp 151–196
55. Reeves CR (2003) Genetic Algorithms. In: Glover F, Kochenberger GA (eds) Handbooks of Metaheuristics. Kluwer, Dordrecht, pp 55–82
56. Resende MGC, Ribeiro CC (2003) Greedy Randomized Adaptive Search Procedures. In: Glover F, Kochenberger GA (eds) Handbook of Metaheuristics. Kluwer, Boston, pp 219–249
57. Russell R, Chiang WC, Zepeda D (2006) Integrating Multi-Product Production and Distribution in Newspaper Logistics. Comput Oper Res 35(5): 1576–1588
58. Simchi-Levi D, Berman O (1988) A Heuristic Algorithm for the Traveling Salesman Location Problem on Networks. Eur J Oper Res 36:478–484
59. Srisvastava R (1993) Alternate Solution Procedures for the Location-Routing Problem. Omega 21(4):497–506
60. Srivastava R, Benton WC (1990) The Location-Routing Problem: Consideration in Physical Distribution System Design. Comput Oper Res 6:427–435
61. Stowers CL, Palekar US (1993) Location Models with Routing Considerations for a Single Obnoxious Facility. Transp Sci 27(4):350–362
62. Tuzun D, Burke LI (1999) A Two-Phase Tabu Search Approach to the Location Routing Problem. Eur J Oper Res 116:87–99
63. Watson-Gandy CTD, Dohrn PJ (1973) Depot Location with Van Salesman – A Practical Approach. Omega 1:321–329

64. Wu TH, Low C, Bai JW (2002) Heuristic Solutions to Multi-Depot Location-Routing Problems. Comput Oper Res 29:1393–1415
65. Zografos KG, Samara S (1989) Combined Location-Routing Model for Hazardous Waste Transportation and Disposal. Transp Res Record 1245:52–59

Logconcave Measures, Logconvexity

ANDRÁS PRÉKOPA
RUTCOR, Rutgers Center for Operations Research, Piscataway, USA

MSC2000: 90C15

Article Outline

Keywords

Logconcave function; Logconcave measure; Logconvex function; Logconvex measure; α-concave function; α-concave measure; Quasiconcave function; Quasiconcave measure

A nonnegative function $f\colon \mathbf{R}^n \to \mathbf{R}^1$ is called a *logconcave* (point) function if for every $\mathbf{x}, \mathbf{y} \in \mathbf{R}^n$ and $0 < \lambda < 1$ we have the inequality

$$f(\lambda\mathbf{x} + (1-\lambda)\mathbf{y}) \geq \left[f(\mathbf{x})\right]^{\lambda}\left[f(\mathbf{y})\right]^{1-\lambda}.$$

A probability measure P defined on the Borel sets of $\mathbf{R}^n$ is called *logconcave* if for any Borel sets $A, B \subset \mathbf{R}^n$ and $0 < \lambda < 1$ we have the inequality

$$\mathsf{P}(\lambda A + (1-\lambda)B) \geq \left[\mathsf{P}(A)\right]^{\lambda}\left[\mathsf{P}(B)\right]^{1-\lambda},$$

provided that $\lambda A + (1-\lambda)B$ is also a Borel set. If P is a logconcave measure in $\mathbf{R}^n$ and $A \subset \mathbf{R}^n$ is a convex set, then $\mathsf{P}(A + \mathbf{z})$ is a logconcave point function in $\mathbf{R}^n$. In particular, the probability distribution function $F(\mathbf{z}) = \mathsf{P}(\{\mathbf{x}\colon \mathbf{x} \leq \mathbf{z}\}) = \mathsf{P}(\{\mathbf{x}\colon \mathbf{x} \leq \mathbf{0}\} + \mathbf{z})$, of the probability measure P, is a logconcave point function. If $n = 1$, then also $1 - F(\mathbf{z})$ is logconcave.

The basic theorem concerning logconcave measures [5,6] states that if the probability measure P is generated by a logconcave probability density function f, i. e.,

$$\mathsf{P}(C) = \int_C f(\mathbf{x})\, d\mathbf{x}$$

for every Borel set $C \subset \mathbf{R}^n$, then P is a logconcave measure.

Examples for logconcave probability distributions are the multivariate normal, the uniform (on a convex set) and for special parameter values the Wishart, the beta, the univariate and some multivariate gamma distributions.

A closely related theorem [5] states that if $f\colon \mathbf{R}^{n+m} \to \mathbf{R}^1$ is a logconcave function, then

$$\int_{\mathbb{R}^m} f(\mathbf{x}, \mathbf{y})\, d\mathbf{y}$$

is a logconcave function in $\mathbf{R}^n$. This implies that the convolution of two logconcave functions is also logconcave [3,5].

Logconcave probability distributions play important role in probabilistic constrained stochastic programming problems. If the problem is:

$$\begin{cases} \min & \mathbf{c}^{\top}\mathbf{x} \\ \text{s.t.} & \mathsf{P}(T\mathbf{x} \geq \boldsymbol{\xi}) \geq p, \\ & A\mathbf{x} = \mathbf{b}, \quad \mathbf{x} \geq \mathbf{0}, \end{cases}$$

and the random vector $\boldsymbol{\xi}$ has continuous distribution with logconcave probability density function, then the set of feasible solutions is convex (for more general results see [6]). On the other hand, if the problem is solved by a barrier function method with logarithmic penalty function, then the function, to be minimized in each step, is convex.

The basic theorem of logconcave measures has the following generalization [1,2]: If $-\infty \leq \alpha \leq \infty$, $0 < \lambda < 1$, and the probability density function $f\colon \mathbf{R}^n \to \mathbf{R}^1$ satisfies ($\mathbf{x}, \mathbf{y} \in \mathbf{R}^n$):

$$f(\lambda\mathbf{x} + (1-\lambda)\mathbf{y}) \geq \left[\lambda f^{\alpha}(\mathbf{x}) + (1-\lambda)f^{\alpha}(\mathbf{y})\right]^{\frac{1}{\alpha}},$$

then for any Borel sets $A, B \subset \mathbf{R}^n$ such that $\lambda A + (1-\lambda)B$ is also a Borel set, we have

$$\mathsf{P}(\lambda A + (1-\lambda)B) \qquad (1)$$

$$\geq \left\{\lambda \left[\mathsf{P}(A)\right]^{\gamma} + (1-\lambda)\left[\mathsf{P}(B)\right]^{\gamma}\right\}^{\frac{1}{\gamma}}, \qquad (2)$$

where $\gamma = \alpha/(1 + n\alpha)$. The cases $\alpha, \gamma = -\infty, 0, \infty$ are interpreted by continuity. Logconcavity corresponds to the case $\alpha = \gamma = 0$. If f, P satisfy the above inequalities, then f is called an *α-concave function* and P a *γ-concave probability measure*. If $\alpha, \gamma = -\infty$, then f and P are called *quasiconcave*.

A nonnegative function $f: \mathbf{R}^n \to \mathbf{R}^1$ is called *logconvex* in the convex set $D \in \mathbf{R}^n$ if for every $\mathbf{x}, \mathbf{y} \in D$ and $0 < \lambda < 1$ we have the inequality

$$f(\lambda\mathbf{x} + (1-\lambda)\mathbf{y}) \leq \left[f(\mathbf{x})\right]^{\lambda}\left[f(\mathbf{y})\right]^{1-\lambda}.$$

Similarly, the probability measure P defined on the Borel subsets of the convex set $D \subset \mathbf{R}^n$ is called *logconvex* if for any Borel sets $A, B \subset D$ we have the inequality

$$\mathsf{P}(\lambda A + (1-\lambda)B) \leq \left[\mathsf{P}(A)\right]^{\lambda}\left[\mathsf{P}(B)\right]^{1-\lambda}.$$

It follows, by Hölder's inequality, that the sum of logconvex functions is also logconvex. This fact, in turn, implies that if f is logconvex in D, then the function of the variable $\mathbf{t} \in \mathbf{R}^n$

$$g(\mathbf{t}) = \int_{C+\mathbf{t}} f(\mathbf{x})\, d\mathbf{x}$$

is logconvex for any fixed Borel set $C \subset \mathbf{R}^n$ in the sense that $g(\lambda\mathbf{t}_1 + (1-\lambda)\,\mathbf{t}_2) \leq [g(\mathbf{t}_1)]^{\lambda}\,[g(\mathbf{t}_2)]^{1-\lambda}$ provided that $C + \mathbf{t}_1 \subset D$, $C + \mathbf{t}_2 \subset D$ and $0 < \lambda < 1$.

See also

- ▶ Approximation of Extremum Problems with Probability Functionals
- ▶ Approximation of Multivariate Probability Integrals
- ▶ Discretely Distributed Stochastic Programs: Descent Directions and Efficient Points
- ▶ Extremum Problems with Probability Functions: Kernel Type Solution Methods
- ▶ General Moment Optimization Problems
- ▶ Logconcavity of Discrete Distributions
- ▶ L-shaped Method for Two-stage Stochastic Programs with Recourse
- ▶ Multistage Stochastic Programming: Barycentric Approximation
- ▶ Preprocessing in Stochastic Programming
- ▶ Probabilistic Constrained Linear Programming: Duality Theory
- ▶ Probabilistic Constrained Problems: Convexity Theory
- ▶ Simple Recourse Problem: Dual Method
- ▶ Simple Recourse Problem: Primal Method
- ▶ Stabilization of Cutting Plane Algorithms for Stochastic Linear Programming Problems
- ▶ Static Stochastic Programming Models
- ▶ Static Stochastic Programming Models: Conditional Expectations
- ▶ Stochastic Integer Programming: Continuity, Stability, Rates of Convergence
- ▶ Stochastic Integer Programs
- ▶ Stochastic Linear Programming: Decomposition and Cutting Planes
- ▶ Stochastic Linear Programs with Recourse and Arbitrary Multivariate Distributions
- ▶ Stochastic Network Problems: Massively Parallel Solution
- ▶ Stochastic Programming: Minimax Approach
- ▶ Stochastic Programming Models: Random Objective
- ▶ Stochastic Programming: Nonanticipativity and Lagrange Multipliers
- ▶ Stochastic Programming with Simple Integer Recourse
- ▶ Stochastic Programs with Recourse: Upper Bounds
- ▶ Stochastic Quasigradient Methods in Minimax Problems
- ▶ Stochastic Vehicle Routing Problems
- ▶ Two-stage Stochastic Programming: Quasigradient Method
- ▶ Two-stage Stochastic Programs with Recourse

References

1. Borell C (1975) Convex set functions in d-space. Periodica Math Hungarica 6:111–136
2. Brascamp HJ, Lieb EH (1976) On extensions of the Brunn–Minkowski and Prékopa–Leindler theorems, including inequalities for log-concave functions, and with an application to the diffusion equations. J Funct Anal 22:366–389
3. Davidovich YS, Korenblum BL, Hacet BI (1969) A property of logarithmically concave functions. Soviet Math Dokl 10:477–480
4. Prékopa A (1971) Logarithmic concave measures with applications to stochastic programming. Acta Sci Math (Szeged) 32:301–316
5. Prékopa A (1973) On logarithmic concave measures and functions. Acta Sci Math (Szeged) 34:335–343
6. Prékopa A (1995) Stochastic programming. Kluwer, Dordrecht

Logconcavity of Discrete Distributions

ANDRÁS PRÉKOPA
RUTCOR, Rutgers Center for Operations Research, Piscataway, USA

MSC2000: 90C15

Article Outline

Keywords

Discrete logconcave distributions; Poisson distribution; Binomial distribution; Hypergeometric distribution; Geometric distribution; Trinomial distribution

The univariate discrete probability distribution $\{p_k: k \in \mathbf{Z}\}$ is called *logconcave* if for every k we have the inequality $p_k^2 \geq p_{k-1}\, p_{k+1}$. This inequality implies that if $k = \lambda i + (1-\lambda)j$, where i, j, k are integers and $0 < \lambda < 1$, then we have the inequality $p_k \geq p_i^{\lambda}\, p_j^{(1-\lambda)}$. Examples are the binomial, Poisson, hypergeometric, geometric distributions.

A classical theorem by M. Fekete [3] states that the convolution of two logconcave univariate discrete distributions is also logconcave.

The multivariate discrete logconcavity [2] is not a direct generalization of its univariate counterpart. The discrete probability distribution $\{\mathsf{P}(\mathbf{x}): \mathbf{x} \in \mathbf{Z}^m\}$ is said to be *logconcave* if there exists a convex function $g: \mathbf{R}^m \to \mathbf{R}$ such that

$$-\log \mathsf{P}(\mathbf{x}) = g(\mathbf{x}) \quad \text{for } \mathbf{x} \in \mathbb{Z}^m.$$

If $\mathsf{P}(\mathbf{x}) = 0$, then by definition $-\log \mathsf{P}(\mathbf{x}) = +\infty$.

The convolution theorem, mentioned above in connection with logconcave univariate distributions, does not carry over to the multivariate case. We know, however, that the trinomial distribution:

$$\mathsf{P}(k_1, k_2) = \frac{n!}{k_1!k_2!(n-k_1-k_2)!} \times p_1^{k_1} p_2^{k_2} (1-p_1-p_2)^{n-k_1-k_2}$$

is logconcave and the convolution of trinomial distributions is also logconcave [5]. For the use of discrete logconcavity in stochastic programming consult [6].

Other definitions and results, concerning multivariate discrete logconcavity, can be found in [1,4].

See also

- Approximation of Extremum Problems with Probability Functionals
- Approximation of Multivariate Probability Integrals
- Discretely Distributed Stochastic Programs: Descent Directions and Efficient Points
- Extremum Problems with Probability Functions: Kernel Type Solution Methods
- General Moment Optimization Problems
- Logconcave Measures, Logconvexity
- L-shaped Method for Two-stage Stochastic Programs with Recourse
- Multistage Stochastic Programming: Barycentric Approximation
- Preprocessing in Stochastic Programming
- Probabilistic Constrained Linear Programming: Duality Theory
- Probabilistic Constrained Problems: Convexity Theory
- Simple Recourse Problem: Dual Method
- Simple Recourse Problem: Primal Method
- Stabilization of Cutting Plane Algorithms for Stochastic Linear Programming Problems
- Static Stochastic Programming Models
- Static Stochastic Programming Models: Conditional Expectations
- Stochastic Integer Programming: Continuity, Stability, Rates of Convergence
- Stochastic Integer Programs
- Stochastic Linear Programming: Decomposition and Cutting Planes
- Stochastic Linear Programs with Recourse and Arbitrary Multivariate Distributions
- Stochastic Network Problems: Massively Parallel Solution
- Stochastic Programming: Minimax Approach
- Stochastic Programming Models: Random Objective
- Stochastic Programming: Nonanticipativity and Lagrange Multipliers
- Stochastic Programming with Simple Integer Recourse

- ▶ Stochastic Programs with Recourse: Upper Bounds
- ▶ Stochastic Quasigradient Methods in Minimax Problems
- ▶ Stochastic Vehicle Routing Problems
- ▶ Two-stage Stochastic Programming: Quasigradient Method
- ▶ Two-stage Stochastic Programs with Recourse

References

1. Bapat RB (1988) Discrete multivariate distributions and generalized log-concavity. Sankhyā Ser. A 50:98–100
2. Barndorff-Nielsen O (1978) Information and exponential families in statistical theroy. Wiley, New York
3. Fekete M, Polya G (1912) Uber ein Problem von Laguerre. Rend Circ Mat Palermo 23:89–120
4. Karlin S, Rinott Y (1981) Entropy inequalities for classes of probability distributions II. The multivariate case. Adv Appl Probab 13:325–351
5. Pedersen JG (1975) On strong unimodality and Mancillarity with applications to contingency tables. Scandinavian J Statist 2:127–137
6. Prékopa A (1995) Stochastic programming. Kluwer, Dordrecht

Logic-Based Outer Approximation

IGNACIO E. GROSSMANN
Department of Chemical Engineering,
Carnegie Mellon University, Pittsburgh, USA

Article Outline

Keywords

Generalized Disjunctive Programming; Disjunctive Programming; Mixed-Integer Programming; Outer-Approximation Method; Logic-based Optimization

Introduction

Turkay and Grossmann [7] proposed a logic version of the outer-approximation algorithm for MINLP by Duran and Grossmann [2] for solving a special class of generalized disjunctive programming (GDP) problems. The problem arises in the optimization of process networks and involves two-term disjunctions in which the first term is activated when a unit or node is selected, while the second term enforces zero values to a subset of the continuous variables. The specific form of the GDP problem is as follows:

$$
\begin{aligned}
&\min Z = \sum_{k \in K} c_k + f(x) \\
&s.t. \quad r(x) \le 0 \\
&\begin{bmatrix} Y_k \\ g_k(x) \le 0 \\ c_k = \gamma_k \end{bmatrix} \vee \begin{bmatrix} \neg Y_k \\ B^k x = 0 \\ c_k = 0 \end{bmatrix} \quad k \in K \qquad \text{(GDP)} \\
&\Omega(Y) = \text{True} \\
&x \in R^n, \quad c \in R^m, \quad Y \in \{\text{true,false}\}^m ,
\end{aligned}
$$

where Y_k are the Boolean variables that decide whether the first term or second term in a disjunction $k \in K$ is true or false, and x are the continuous variables. The objective function involves the term $f(x)$ for the continuous variables and the charges c_k that depend on the discrete choices in each disjunction $k \in K$. The constraints $r(x) \le 0$ must hold regardless of the discrete choices. In contrast, $g_k(x) \le 0$ are conditional constraints that must hold when Y_k is true in the kth disjunction; otherwise $(\neg Y_i)$ a subset of the x variables is set to zero with the proper definition of the matrix B^i. In particular, we define $B^i = \left[b^{\mathrm{T}}\right]$ such that $b_j^{\mathrm{T}} = e^{\mathrm{T}}$ if $x_j = 0$, and $b_j^{\mathrm{T}} = 0^{\mathrm{T}}$ if $x_j \neq 0$. In this way only a subset of the variables x is forced to zero (typically flows). The cost variables c_k correspond to the fixed charges, and their value equals γ_k if the Boolean variable Y_k is true; otherwise they are zero. $\Omega(Y) = \text{True}$ are logical relations for the Boolean variables expressed as propositional logic. It is assumed for the derivation of basic methods that the functions are convex, although in practical applications these often correspond to nonconvex functions.

NLP and Master Subproblems

Following the original algorithm [2], the logic-based outer-approximation algorithm consists of solving NLP subproblems and disjunctive or MILP master problems.

As described in Turkay and Grossmann [7], for fixed values of the Boolean variables, Y_k = true and Y_k = false, the corresponding NLP subproblem is as follows:

$$\begin{aligned}
&\min Z = \sum_{k\in K} c_k + f(x)\\
&s.t. \quad r(x) \le 0\\
&\qquad \left.\begin{array}{l} g_k(x) \le 0 \\ c_k = \gamma_k \end{array}\right\} \text{for } Y_k = \text{true} \quad k \in K\\
&\qquad \left.\begin{array}{l} B^k x = 0 \\ c_k = 0 \end{array}\right\} \text{for } Y_k = \text{false} \quad k \in K\\
&\quad x \in R^n, \quad c_i \in R^m,
\end{aligned} \tag{NLPD}$$

Note that for every disjunction $k \in K$ only constraints corresponding to the Boolean variable Y_k that is true are imposed, thus leading to a reduction in the size of the problem. Also, fixed charges γ_k are only applied to these terms. Assuming that *NF* subproblems (NLPD) are solved in which sets of linearizations $l = 1, \ldots NF$ are generated for subsets of disjunction terms $L_k = \{l | Y_k^l = \text{true}\}$, one can define the following disjunctive OA master problem:

$$\begin{aligned}
&\text{Min} Z = \sum_k c_k + \alpha\\
&s.t. \quad \left.\begin{array}{l} \alpha \ge f(x^l) + \nabla f(x^l)^{\mathrm{T}}(x - x^l) \\ r(x^l) + \nabla r(x^l)^{\mathrm{T}}(x - x^l) \le 0 \end{array}\right\} l = 1, \ldots, L\\
&\left[\begin{array}{l} Y_k \\ g_k(x^l) + \nabla g_k(x^l)^{\mathrm{T}}(x - x^l) \le 0 \\ l \in L_k \\ c_k = \gamma_k \end{array}\right] \vee \left[\begin{array}{l} \neg Y_k \\ B^k x = 0 \\ c_k = 0 \end{array}\right]\\
&\quad k \in K\\
&\quad \Omega(Y) = \text{True}\\
&\alpha \in R, \quad x \in R^n, \quad c \in R^m, \quad Y \in \{\text{true, false}\}^m
\end{aligned} \tag{MGDP}$$

It should be noted that before applying the above master problem it is necessary to solve various subproblems (NLPD) so as to produce at least one linear approximation of each of the terms in the disjunctions. As shown by Turkay and Grossmann [7], selecting the smallest number of subproblems amounts to solving a set covering problem, which is of small size and easy to solve. In the context of a process flowsheet synthesis problem, another way of generating the linearizations in (MGDP) is by starting with an initial flowsheet and suboptimizing the remaining subsystems.

The above problem (MGDP) can be solved by the methods described by Beaumont [1], Raman and Grossmann [6], and Hooker [4]. Turkay and Grossmann [4] have shown that if the convex hull representation of the disjunctions is used in (MGDP), then converting the logic relations $\Omega(Y)$ into the inequalities $Ay \le a$ leads to the following MILP problem:

$$\begin{aligned}
&\text{Min} Z = \sum_k y_k y_k + \alpha\\
&s.t. \quad \left.\begin{array}{l} \alpha \ge f(x^l) + \nabla f(x^l)^{\mathrm{T}}(x - x^l) \\ r(x^l) + \nabla r(x^l)^{\mathrm{T}}(x - x^l) \le 0 \end{array}\right\} l = 1, \ldots L\\
&\nabla_{x_{Z_k}} g_k(x^l)^{\mathrm{T}} x_{Z_k} + \nabla_{x_{N_k}} g_k(x^l)^{\mathrm{T}} x_{N_k}^1\\
&\quad \le \left[-g_k(x^l) + \nabla_x g_k(x^l)^{\mathrm{T}} x^l\right] y_k\\
&l \in L_k, \quad k \in K\\
&x_{N_k} = x_{N_k}^1 + x_{N_k}^2\\
&0 \le x_{N_k}^1 \le x_{N_k}^U y_k\\
&0 \le x_{N_k}^2 \le x_{N_k}^U (1 - y_k)\\
&Ay \le a\\
&x \in R^n, \quad x_{N_k}^1 \ge 0, \quad x_{N_k}^2 \ge 0, \quad y \in \{0,1\}^m
\end{aligned} \tag{MIPDF}$$

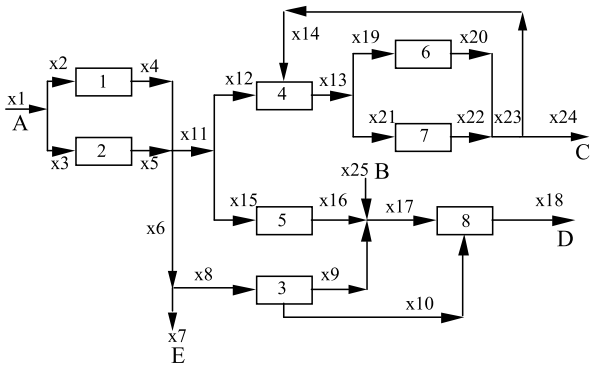
where the vector x is partitioned into the variables (x_{z_k}, x_{N_k}) for each disjunction k according to the definition of the matrix B^i (i. e., x_z refers to nonzero rows of this matrix). It is interesting to note that the logic-based outer-approximation algorithm represents a generalization of the modeling/decomposition strategy of Kocis and Grossmann [5] for the synthesis of process flowsheets.

Steps of Algorithm

Assuming feasible NLP subproblems, the steps of the proposed logic-based auter-approximation method are as follows:

Step 1: Model the problem in generalized disjunctive form as in (GDP).

Step 2: Identify the NF subproblems to be solved either from inspection or from set covering problems.



Logic-Based Outer Approximation, Figure 1
Process network example

Step 3: Solve NLP subproblems (NLPD) for the NF subproblems determined in step 2. The lowest-cost solution of these NLPs yields an upper bound, Z_U, for the problem.
Step 4: Linearize the objective function and constraints of the current NLP subproblem(s) and set up the MILP master problem (MIPDF). The solution of this problem gives the lower bound, Z_L, for the problem.
Step 5: If $|Z_U - Z_L| \leq \varepsilon$, where ε is a tolerance, then stop. The solution with the current Z_U is the optimal solution. Otherwise go to step 6.
Step 6: Solve NLP subproblem (NLPD) by fixing the Boolean variables predicted by the master problem. The objective function value of the solution is Z_{NLP}. If $Z_{NLP} < Z_U$, then set $Z_U = Z_{NLP}$.
Step 7: Compare the upper bound Z_U with the lower bound Z_L. If $|Z_U - Z_L| \leq \varepsilon$, then stop; the solution with the current Z_U is the optimal solution. Otherwise go to step 4.

It should be noted that one can also derive a logic-based version of Generalized Benders Decomposition as described in [7]. The logic outer-approximation algorithm described above has been implemented in the computer code LOGMIP by Vecchietti and Grossmann [8], which can be accessed from http://www.logmip.ceride.gov.ar

Example

Consider the following (GDP) problem from [7] that deals with a simplified version of the synthesis of a process network shown in Fig. 1.

The GDP model is as follows:

1. Objective function:

$$\begin{aligned}\min Z = {} & c_1 + c_2 + c_3 + c_4 + c_5 + c_6 + c_7 + c_8 + x_2 \\ & - 10x_3 + x_4 - 15x_5 - 40x_9 + 15x_{10} + 15x_{14} \\ & + 80x_{17} - 65x_{18} + 25x_{19} - 60x_{20} \\ & + 35x_{21} - 80x_{22} - 35x_{25} + 122\end{aligned}$$

2. Material balances at mixing/splitting points:

$$\begin{aligned} x_1 - x_2 - x_3 &= 0 \\ x_4 + x_5 - x_6 - x_{11} &= 0 \\ x_{13} - x_{19} - x_{21} &= 0 \\ x_{17} - x_9 - x_{16} - x_{25} &= 0 \\ x_{11} - x_{12} - x_{15} &= 0 \\ x_6 - x_7 - x_8 &= 0 \\ x_{23} - x_{20} - x_{22} &= 0 \\ x_{23} - x_{14} - x_{24} &= 0 \end{aligned}$$

3. Specifications on the flows:

$$\begin{aligned} x_{10} - 0.8x_{17} &\leq 0 \\ x_{10} - 0.4x_{17} &\geq 0 \\ x_{12} - 5x_{14} &\leq 0 \\ x_{12} - 2x_{14} &\geq 0 \end{aligned}$$

4. Disjunctions:

$$\text{Unit 1:} \begin{bmatrix} Y_1 \\ \exp(x_4) - 1 - x_2 \leq 0 \\ c_1 = 5 \end{bmatrix} \vee \begin{bmatrix} \neg Y_1 \\ x_2 = x_4 = 0 \\ c_1 = 0 \end{bmatrix}$$

$$\text{Unit 2:} \begin{bmatrix} Y_2 \\ \exp(x_5/1.2) - 1 - x_3 \leq 0 \\ c_2 = 8 \end{bmatrix} \vee \begin{bmatrix} \neg Y_2 \\ x_4 = x_3 = 0 \\ c_2 = 0 \end{bmatrix}$$

$$\text{Unit 3:} \begin{bmatrix} Y_3 \\ 1.5x_9 - x_8 + x_{10} = 0 \\ c_3 = 6 \end{bmatrix} \vee \begin{bmatrix} \neg Y_3 \\ x_8 = x_9 = x_{10} = 0 \\ c_3 = 0 \end{bmatrix}$$

$$\text{Unit 4:} \begin{bmatrix} Y_4 \\ 1.5(x_{12}+x_{14})-x_{13}=0 \\ c_4=10 \end{bmatrix} \vee \begin{bmatrix} \neg Y_4 \\ x_{12}=x_{13}=x_{14}=0 \\ c_4=0 \end{bmatrix}$$

$$\text{Unit 5:} \begin{bmatrix} Y_5 \\ x_{15}-2x_{16}=0 \\ c_5=6 \end{bmatrix} \vee \begin{bmatrix} \neg Y_5 \\ x_{15}=x_{16}=0 \\ c_5=0 \end{bmatrix}$$

$$\text{Unit 6:} \begin{bmatrix} Y_6 \\ \exp(x_{20}/1.5)-1-x_{19}\le 0 \\ c_6=7 \end{bmatrix} \vee \begin{bmatrix} \neg Y_6 \\ x_{19}=x_{20}=0 \\ c_6=0 \end{bmatrix}$$

$$\text{Unit 7:} \begin{bmatrix} Y_7 \\ \exp(x_{22})-1-x_{21}\le 0 \\ c_7=4 \end{bmatrix} \vee \begin{bmatrix} \neg Y_7 \\ x_{21}=x_{22}=0 \\ c_7=0 \end{bmatrix}$$

$$\text{Unit 8:} \begin{bmatrix} Y_8 \\ \exp(x_{18})-1-x_{10}-x_{17}\le 0 \\ c_8=5 \end{bmatrix} \vee \begin{bmatrix} \neg Y_8 \\ x_{10}=x_{17}=x_{18}=0 \\ c_8=0 \end{bmatrix}$$

5. Propositional Logic $[\Omega = (Y_i)]$:

$$Y_1 \Rightarrow Y_3 \vee Y_4 \vee Y_5$$
$$Y_2 \Rightarrow Y_{13} \vee Y_4 \vee Y_5$$
$$Y_3 \Rightarrow Y_1 \vee Y_2, \quad Y_3 \Rightarrow Y_8$$
$$Y_4 \Rightarrow Y_1 \vee Y_2, \quad Y_4 \Rightarrow Y_6 \vee Y_7$$
$$Y_5 \Rightarrow Y_1 \vee Y_2, \quad Y_5 \Rightarrow Y_8$$
$$Y_6 \Rightarrow Y_4$$
$$Y_7 \Rightarrow Y_4$$
$$Y_8 \Rightarrow Y_3 \vee Y_5 \vee (\neg Y_3 \wedge \neg Y_5)$$

Logic-Based Outer Approximation, Table 1
Progress of iterations

Subproblem	Objective value
NLPD1	73.277
NLPD2	103.584
NLPD3	113.789
MGDP	67.948
NLPD4	68.009

6. Specifications:

$$Y_1 \veebar Y_2$$
$$Y_4 \veebar Y_5$$
$$Y_6 \veebar Y_7$$

7. Variables:

$$x_j, \quad c_i \ge 0, \quad Y_i = \{\text{True,False}\}$$
$$i = 1, 2, \ldots, 8, \; j = 1, 2, \ldots, 25$$

Applying LOGMIP to solve this problem, and starting with three NLP subproblems at

$$\text{NLPD1}: Y_2 = Y_3 = Y_4 = Y_5 = Y_8 = \text{True}$$
$$\text{NLPD2}: Y_1 = Y_3 = Y_4 = Y_7 = Y_8 = \text{True}$$
$$\text{NLPD2}: Y_2 = Y_4 = Y_6 = Y_7 = \text{True}$$

the predicted optimum solution is given by $Z = 68.009$. Table 1 shows the progress of the iterations.

References

1. Beaumont N (1991) An Algorithm for Disjunctive Programs. Eur J Oper Res 48:362–371
2. Duran MA, Grossmann IE (1986) An Outer-Approximation Algorithm for a Class of Mixed-integer Nonlinear Programs. Math Program 36:307
3. Geoffrion AM (1972) Generalized Benders Decomposition. J Optim Theory Appl 10(4):237-260
4. Hooker JN (1999) Logic-Based Methods for Optimization. Wiley, New York
5. Kocis GR, Grossmann IE (1989) A Modeling and Decomposition Strategy for the MINLP Optimization of Process Flowsheets. Comput Chem Eng 13:797
6. Raman R, Grossmann IE (1994) Modelling and Computational Techniques for Logic Based Integer Programming. Comput Chem Eng 18(7):563–578
7. Turkay M, Grossmann IE (1996) Logic-based MINLP Algorithms for the Optimal Synthesis of Process Networks. Comput Chem Eng 20(8):959–978
8. Vecchietti A, Grossmann IE (1999) LOGMIP: A Disjunctive 0-1 Nonlinear Optimizer for Process Systems Models. Comput Chem Eng 23:555–565

Lovász Number

STANISLAV BUSYGIN
Department of Industrial and Systems Engineering, University of Florida, Gainesville, USA

MSC2000: 05C69, 05C15, 05C17, 05C35, 90C35, 90C22

Article Outline

Synonyms

ϑ-function

Introduction

Let $G(V,E)$ be a simple undirected graph, $V = \{1,2,\ldots,n\}$. The *adjacency matrix* of G is a matrix $A_G = (a_{ij})_{n\times n}$, where $a_{ij} = 1$ if $(i,j) \in E$ and $a_{ij} = 0$ otherwise. The set of vertices *adjacent* to a vertex $i \in V$ will be denoted by $N(i) = \{j \in V : (i,j) \in E\}$ and called the *neighborhood* of the vertex i. We will also consider the *complementary graph* $\bar{G}(V,\bar{E})$ having the same set of vertices V, but an edge $(i,j) \in \bar{E}$ if and only if i and j are not adjacent in G.

An *independent set* S is a subset of V such that no two vertices of S are adjacent, i. e., $\forall i \in S N(i) \cap S = \varnothing$. The set S is called a *maximal* independent set if any vertex $i \in V \setminus S$ has at least one adjacent vertex in S, i. e., $\forall i \in V \setminus S\ N(i) \cap S \neq \varnothing$. Finally, the set S is called a *maximum* independent set if it has the largest cardinality among all independent sets of the graph. This cardinality will be denoted by $\alpha(G)$ and called the *independence* (or *stability*) *number* of the graph G.

In addition to the maximum cardinality stable sets, we will consider the *maximum weight independent sets*. Let there be a given vector $w = (w_1, w_2, \ldots, w_n)^{\mathrm{T}}$ of nonnegative *vertex weights*. A maximum weight independent set is such an independent set $S \subseteq V$ that has the largest weight $\alpha(G,w) = \max_S \sum_{i\in S} w_i$.

Similarly, a *clique* Q of the graph G is a subset of V such that any two vertices in it are adjacent, i. e., $\forall i \in Q\ N(i) \cap Q = Q \setminus \{i\}$. The clique Q is called *maximal* if for any vertex $i \in V \setminus Q$ there is at least one vertex in Q non-adjacent to i, i. e., $\forall i \in V \setminus Q\ N(i) \cap Q \neq Q$. If Q has the largest cardinality among all cliques of the graph, it is called a *maximum* clique. The cardinality of a maximum clique will be denoted by $\omega(G)$ and called the *clique number* of the graph G. A *maximum weight clique* is a clique having the largest weight $\omega(G,w) = \max_Q \sum_{i\in Q} w_i$.

It is easy to see that independent sets of the graph G correspond to cliques of $\bar{G}$, and vise versa.

We will denote by $\chi(G)$ the *chromatic number* of the graph G (i. e. the minimum number of colors to which the graph vertices can be colored without using one color for any two adjacent vertices.) The number $\chi(\bar{G})$, giving the minimum number of cliques of G to which the vertex set V can be partitioned, will be also denoted by $\bar{\chi}(G)$ and called the *clique partition number* of the graph G.

Next, for two graphs $G_1(V_1,E_1)$ and $G_2(V_2,E_2)$ we define their *strong product* $G_1 \cdot G_2$ as the graph, whose vertex set is the Cartesian product $V_1 \times V_2$ and in which a vertex (i,j) is adjacent to a vertex (i',j') if and only if $(i,i') \in E_1$ and $(j,j') \in E_2$. The strong product of k copies of G will be denoted by G^k.

Formulation

Lovász Number as an Upper Bound of Shannon Capacity

Let us consider the set $V = \{1,2,\ldots,n\}$ to be an alphabet in which the adjacency means that the two letters can be confused. Then any set of one-letter messages that cannot be confused with each other corresponds to an independent set of the graph and vise versa. Furthermore, the maximum number of one-letter messages that cannot be confused with each other is equal to $\alpha(G)$, and the maximum number of k-letter messages that cannot be confused with each other is equal to $\alpha(G^k)$. It is easy to see that there are at least $\alpha(G)^k$ k-letter messages that cannot be confused with each other,

so $\alpha(G^k) \geq \alpha(G)^k$. So,

$$\Theta(G) = \sup_k \sqrt[k]{\alpha(G^k)} = \lim_{k\to\infty} \sqrt[k]{\alpha(G^k)} \geq \alpha(G)\,. \quad (1)$$

The value $\Theta(G)$ is called the *Shannon zero-error capacity* of the graph G [14]. Generally, it is extremely hard to compute, and nowadays $\Theta(G)$ is not even known for the graph C_7 (cycle of 7 vertices).

Thus, the independence number $\alpha(G)$ gives a *lower* bound on $\Theta(G)$. In 1979, L. Lovász defined a new nontrivial *upper* bound on the Shannon zero-error capacity of a graph in his seminal paper [11]. This function was named later *Lovász number* (or ϑ*-function*) of a graph.

First, define an *orthonormal representation* of the graph G as a system $(u_1, u_2, \ldots, u_n)$ of unit vectors in a Euclidean space such that whenever two vertices i and j are not adjacent, the vector u_i is orthogonal to the vector u_j. It is easy to see that such systems of vectors do exist, e. g., any n orthonormal vectors from the space $\mathbb{R}^n$. The ϑ-function is defined as the following minimax value:

$$\vartheta(G) = \min_{\{c,(u_i)\}} \max_{i\in V} \frac{1}{(c^{\mathrm{T}}u_i)^2}\,, \quad (2)$$

where c ranges over unit vectors of the same dimensionality that the vectors u_i are. The vector c was called by Lovász the *handle* of the representation.

It can be shown that for a strong product of graphs, $\vartheta(G\cdot H) \leq \vartheta(G)\vartheta(H)$. To show that $\alpha(G) \leq \vartheta(G)$ one needs to observe that if S is a maximum independent set of G, then $1 = c^2 \geq \sum_{i\in S}(c^{\mathrm{T}}u_i)^2 \geq \alpha(G)/\vartheta(G)$. From here it is obvious that $\Theta(G) \leq \vartheta(G)$ as $\alpha(G^k) \leq \vartheta(G^k) \leq \vartheta(G)^k$.

Similarly, we introduce the weighted ϑ-function:

$$\vartheta(G,w) = \min_{\{c,(u_i)\}} \max_{i\in V} \frac{w_i}{(c^{\mathrm{T}}u_i)^2}\,, \quad (3)$$

which gives an upper bound for $\alpha(G,w) \leq \vartheta(G,w)$.

In contrast to $\Theta(G)$ and $\alpha(G,w)$, which are hard to compute, $\vartheta(G,w)$ can be computed with an arbitrary precision in a polynomial time by either the ellipsoid method or an interior point method due to its semidefinite programming formulation considered below (see also [7,9,13]). This makes ϑ-function attractive for estimating these intractable numbers.

The Sandwich Theorem

Other equivalent formulations implying a number of interesting properties of $\vartheta(G)$ were established in [7] (see also the extensive survey [9]). To introduce them, let us define three specific convex sets in $\mathbb{R}^n$ associated with the graph:

$$\begin{aligned}
\mathcal{STAB}(G) &= \mathrm{hull}(\{x\in\{0,1\}^n | x_j + x_k \leq 1,\\
&\qquad \forall (j,k)\in E\}),\\
\mathcal{TH}(G) &= \{x\geq 0 | \sum_{j\in V}(c^{\mathrm{T}}u_j)^2 x_j \leq 1,\\
&\qquad \forall \text{ ort. lab. } (u_j) \text{ of } G,\ \|c\| = 1\},\\
\mathcal{QSTAB}(G) &= \{x\geq 0 \mid \sum_{j\in Q} x_j \leq 1,\\
&\qquad \forall \text{ cliques } Q \text{ of } G\}\,.
\end{aligned}$$

Let $x^S \in \{0,1\}^n$ be the *incidence vector* of an independent set S, that is, $x_i^S = 1$ if $i\in S$, and $x_i = 0$ otherwise. Then, obviously, for any orthonormal representation (u_i) and a unit vector c,

$$\sum_{j\in V}(c^{\mathrm{T}}u_j)^2 x_j^S = \sum_{j\in S}(c^{\mathrm{T}}u_j)^2 \leq 1\,.$$

So, any $x\in\mathcal{STAB}(G)$ satisfy the constraints of $\mathcal{TH}(G)$. Let Q be any clique of G. Then we can construct an orthonormal representation as follows. Let all vectors $(u_i)_{i\in V\setminus Q}$ be mutually orthogonal, and also each of them be orthogonal to another unit vector c. We set all $(u_i)_{i\in Q}$ to be equal to c. If we consider the constraint $\sum_{j\in V}(c^{\mathrm{T}}u_j)^2 x_j \leq 1$ over only such orthonormal representations, we obtain the clique constraints defining the set $\mathcal{QSTAB}(G)$. Hence, we have

$$\mathcal{STAB}(G) \subseteq \mathcal{TH}(G) \subseteq \mathcal{QSTAB}(G)\,. \quad (4)$$

Obviously,

$$\alpha(G,w) = \max_x\{w^{\mathrm{T}}x \mid x\in\mathcal{STAB}(G)\}. \quad (5)$$

Let us also denote

$$\kappa(G,w) = \max_x\{w^{\mathrm{T}}x \mid x\in\mathcal{QSTAB}(G)\}. \quad (6)$$

We will prove that

$$\vartheta(G,w) = \max_x\{w^{\mathrm{T}}x \mid x\in\mathcal{TH}(G)\} \quad (7)$$

and henceforth conclude that

$$\alpha(G,w) \le \vartheta(G,w) \le \kappa(G,w). \tag{8}$$

The double inequality (8) constitutes the famous *sandwich theorem*.

Let us denote by S_n the set of all $n \times n$ symmetric matrices, and by S_n^+ the set of all *positive semidefinite* $n \times n$ matrices:

$$S_n^+ = \{A \in S_n |\, x^{\mathrm{T}} A x \ge 0 \ \forall x \in \mathbb{R}^n\}.$$

We also denote by $z = (\sqrt{w_1}, \sqrt{w_2}, \ldots, \sqrt{w_n})^{\mathrm{T}}$ the vector of square roots of the vertex weights. Consider the following functions of the graph and vertex weights:

$$\begin{aligned}\vartheta_2(G,w) &= \min_{A \in S_n} \lambda_{\max}(A),\\ \text{s.t. } a_{ij} &= \sqrt{w_i w_j}, \quad \forall (i,j) \notin E,\end{aligned}$$

where $\lambda_{\max}(A)$ denotes the largest eigenvalue of A;

$$\begin{aligned}\vartheta_3(G,w) &= \max_{X \in S_n^+} z^{\mathrm{T}} X z,\\ \text{s.t. } x_{ij} &= 0, \quad \forall (i,j) \in E, \quad \mathrm{tr}(X) = 1,\end{aligned}$$

where $\mathrm{tr}(X) = \sum_{i=1}^n x_{ii}$ denotes the *trace* of the matrix X;

$$\vartheta_4(G,w) = \max_{\{d,(v_i)\}} \sum_{i \in V} (d^{\mathrm{T}} v_i)^2 w_i,$$

where $(v_i)_{i \in V}$ range over all orthonormal representations of the complementary graph $\bar{G}$ and $\|d\| = 1$.

Theorem 1

$$\begin{aligned}\vartheta(G,w) &= \vartheta_2(G,w) = \vartheta_3(G,w) = \vartheta_4(G,w)\\ &= \max_x \{w^T x \mid x \in \mathcal{TH}(G)\}.\end{aligned}$$

Proof First we show that $\vartheta(G,w) \le \vartheta_2(G,w)$. Consider a matrix $A \in S_n$ such that $a_{ij} = \sqrt{w_i w_j}$, $\forall (i,j) \notin E$, and let $t = \lambda_{\max}(A)$. Then $tI - A \in S_n^+$, and hence there exists $X \in \mathbb{R}^{n \times n}$ such that $tI - A = X^{\mathrm{T}} X$. Let $x_i \in \mathbb{R}^n$ be the i-th column of X. Then

$$x_i^{\mathrm{T}} x_i = t - w_i, \quad \forall i \in V$$

and

$$x_i^{\mathrm{T}} x_j^{\mathrm{T}} = -\sqrt{w_i w_j}, \quad \forall i, j \text{ nonadjacent in } G.$$

Note that $\mathrm{rank}(X) < n$ since the matrix $tI - A$ has a zero eigenvalue. This implies that there exists a unit vector $c \in \mathbb{R}^n$ orthogonal to all x_i, $i \in V$. Consider the vectors

$$u_i = (\sqrt{w_i} c + x_i)/\sqrt{t}, \quad i \in V.$$

It is easy to see that

$$u_i^{\mathrm{T}} u_i = \frac{(w_i c^{\mathrm{T}} c + x_i^{\mathrm{T}} x_i)}{t} = 1$$

and for any two nonadjacent vertices $i, j \in V$,

$$u_i^{\mathrm{T}} u_j = \frac{(\sqrt{w_i w_j} c^{\mathrm{T}} c + x_i^{\mathrm{T}} x_j)}{t} = 0.$$

Hence, the vectors (u_i) form an orthonormal representation of G and

$$\vartheta(G,w) \le \max_{i \in V} \frac{w_i}{(c^{\mathrm{T}} u_i)^2} = \max_{i \in V} \frac{w_i}{w_i/t} = t = \lambda_{\max}(A).$$

Now, we show that $\vartheta_2(G,w) \le \vartheta_3(G,w)$. We have $z^{\mathrm{T}} X z \le \vartheta_3 \cdot \mathrm{tr}(X)$ for any $X \in S_n^+$ such that $x_{ij} = 0\ \forall (i,j) \in E$. This inequality is equivalent to $(W - \vartheta_3 I) \bullet X \le 0$, where $W = (\sqrt{w_i w_j})_{n \times n}$ and "$\bullet$" denotes the Euclidian inner product in $\mathbb{R}^{n \times n}$, i. e., $A \bullet B = \sum_{i,j} a_{ij} b_{ij}$. From here it can be inferred that the matrix $\vartheta_3 I - W$ is a sum of some positive semidefinite matrix $D \in S_n^+$ and another symmetric matrix $A = (a_{ij}) \in S_n$ such that if $(i,j) \notin E$, then $a_{ij} = 0$. This implies that $\vartheta_3 I - W - A \in S_n^+$ and hence $\vartheta_3 \ge \lambda_{\max}(W + A) \ge \vartheta_2$.

Next, we show that $\vartheta_3(G,w) \le \vartheta_4(G,w)$. Let $X = (x_{ij}) \in \mathbb{R}^{n \times n}$ be an optimum matrix for the program defining ϑ_3. Since $X \in S_n^+$, there exists a matrix $Y \in \mathbb{R}^{n \times n}$ such that $X = Y^{\mathrm{T}} Y$. Let x_i denote the i-th column of X and y_i denote the i-th column of Y. Construct an orthonormal system of vectors $(u_i)_{i \in V}$ in $\mathbb{R}^n$ such that there is the vector $u_i = y_i / \|y_i\|$ whenever $y_i \ne 0$. Since $y_i^{\mathrm{T}} y_j = x_{ij} = 0 \forall (i,j) \in E$, the system (u_i) is an orthonormal representation of $\bar{G}$. Furthermore, $z^{\mathrm{T}} Y^{\mathrm{T}} Y z = z^{\mathrm{T}} X z = \vartheta_3$ and hence $d = Yz/\sqrt{\vartheta_3}$ is a unit vector. Whenever $y_i \ne 0$,

$$d^{\mathrm{T}} v_i = \frac{z^{\mathrm{T}} Y^{\mathrm{T}} y_i}{(\sqrt{\vartheta_3} \|y_i\|)} = \frac{z^{\mathrm{T}} x_i}{(\sqrt{\vartheta_3} \|y_i\|)}.$$

Thus, $\|y_i\| d^{\mathrm{T}} v_i = z^{\mathrm{T}} x_i / \sqrt{\vartheta_3}$, $\forall i \in V$, and

$$\begin{aligned}\sum_{i \in V} \|y_i\| \sqrt{w_i} d^{\mathrm{T}} v_i &= \frac{1}{\sqrt{\vartheta_3}} \sum_{i \in V} z^{\mathrm{T}} x_i \sqrt{w_i}\\ &= \frac{1}{\sqrt{\vartheta_3}} z^{\mathrm{T}} X z = \sqrt{\vartheta_3}.\end{aligned}$$

Using the Cauchy–Schwarz inequality,

$$\begin{aligned}\vartheta_3 &= \left(\sum_{i\in V}\|y_i\|\sqrt{w_i}d^{\mathrm{T}}v_i\right)^2\\ &\le \left(\sum_{i\in V}\|y_i\|^2\right)\left(\sum_{i\in V}w_i(d^{\mathrm{T}}v_i)^2\right)\\ &= \left(\sum_{i\in V}x_{ii}\right)\left(\sum_{i\in V}w_i(d^{\mathrm{T}}v_i)^2\right)\\ &= \sum_{i\in V}w_i(d^{\mathrm{T}}v_i)^2 \le \vartheta_4(G,w)\,.\end{aligned}$$

Next, we prove that $\vartheta_4(G,w) \le \max_x\{w^{\mathrm{T}}x \mid x \in \mathcal{TH}(G)\}$. Let $(v_i)_{i\in V}$ and d be correspondingly an optimum orthogonal representation of $\bar{G}$ and its handle for the program defining ϑ_4. We show that the vector $\left((d^{\mathrm{T}}v_i)^2\right)_{i\in V}$ belongs to $\mathcal{TH}(G)$. Consider some orthonormal representation of G, $(u_i)_{i\in V}$, $u_i \in \mathbb{R}^n$, and let $c \in \mathbb{R}^n$ be some unit vector. The matrices $u_iv_i^{\mathrm{T}} \in \mathbb{R}^{n\times n}$ are mutually orthogonal and have the unit norm with respect to the inner product "$\bullet$", i. e.,

$$(u_iv_i^{\mathrm{T}})\bullet(u_jv_j^{\mathrm{T}}) = (u_i^{\mathrm{T}}u_j)(v_i^{\mathrm{T}}v_j) = \begin{cases}1, \text{if } i=j,\\ 0, \text{if } i\ne j.\end{cases}$$

Now we may conclude that

$$\begin{aligned}\sum_{i\in V}(c^{\mathrm{T}}u_i)^2(d^{\mathrm{T}}v_i)^2 &= \sum_{i\in V}((cd^{\mathrm{T}})\bullet(u_iv_i^{\mathrm{T}}))^2\\ &\le (cd^{\mathrm{T}})\bullet(cd^{\mathrm{T}}) = 1\,.\end{aligned}$$

Hence $\left((d^{\mathrm{T}}v_i)^2\right)_{i\in V} \in \mathcal{TH}(G)$ and

$$\begin{aligned}\vartheta_4(G,w) &= \sum_{i\in V}w_i(d^{\mathrm{T}}v_i)^2\\ &\le \max_x\{w^{\mathrm{T}}x \mid x\in\mathcal{TH}(G)\}\,.\end{aligned}$$

The final step is to show that $\max_x\{w^{\mathrm{T}}x \mid x \in \mathcal{TH}(G)\}$
$\le \vartheta(G,w)$. Let x^* be a vector maximizing w^Tx over $\mathcal{TH}(G)$. Choosing any orthonormal representation $(u_i)_{i\in V}$ and a unit vector c, we have

$$\begin{aligned}w^{\mathrm{T}}x^* &\le \left(\max_i\frac{w_i}{(c^{\mathrm{T}}u_i)^2}\right)\sum_{i\in V}(c^{\mathrm{T}}u_i)^2x_i^*\\ &\le \max_i\frac{w_i}{(c^{\mathrm{T}}u_i)^2} \le \vartheta(G,w)\,.\end{aligned}$$

The four inequalities established above can hold if and only if all the ϑ's are equal. QED. □

Let us denote by $\kappa(G)$ the value of $\kappa(G,w)$ when all vertex weights are 1's. It is easy to see that $\kappa(G) \le \bar{\chi}(G)$. Indeed, if we take a minimum clique partition $\{Q_1, Q_2, \ldots, Q_{\bar{\chi}}\}$ of G, then $\bar{\chi}(G)$ is equal to the optimum value of the program:

$$\begin{aligned}&\max_{x\in\mathbb{R}^n}\sum_{i\in V}x_i,\\ &\text{s.t.}\sum_{i\in Q_k}x_i \le 1, \quad k=1,2,\ldots,\bar{\chi}, \quad x\ge 0\,.\end{aligned}$$

This program has the same objective as the program for $\kappa(G)$, but its set of constraints is a subset of constraints of $\mathcal{QSTAB}(G)$. So, we may extend the sandwich inequality to

$$\alpha(G) \le \vartheta(G) \le \kappa(G) \le \bar{\chi}(G)\,.$$

Omitting $\kappa(G)$ and applying the inequalities to the complementary graph, we obtain

$$\omega(G) \le \vartheta(\bar{G}) \le \chi(G)\,, \tag{9}$$

which expresses another famous form of the sandwich theorem stating that a polynomial-time computable number $\vartheta(\bar{G})$ lies in between the two *NP*-hard numbers: the clique number and the chromatic number.

Lovász Number as a Dual Bound of Quadratic Maximization

Consider the following quadratic formulation of the maximum weight independent set problem:

$$\begin{aligned}\alpha(G,w) &= \max w^{\mathrm{T}}x\\ &\text{s.t. } x_ix_j = 0, \quad \forall(i,j)\in E,\\ &\quad x_i^2 - x_i = 0,\ i=1, \quad \ldots,n\,.\end{aligned} \tag{10}$$

It has been shown by N.Z. Shor that the optimal Lagrangian dual bound of program (10) is equal to $\vartheta(G,w)$ [15]. One can compute this bound for a maximization problem with a quadratic objective subject to a set of linear and quadratic constraints minimizing a convex non-differentiable function defined over a parametric (linearly dependent on Lagrangian multipliers) set of negative semidefinite symmetric matrices.

Indeed, the Lagrangian of program (10) is

$$L(x, \Lambda) = w^{\mathrm{T}}x + \sum_{i=1}^{n} \lambda_{ii}(x_i^2 - x_i) + \sum_{(i,j)\in E} \lambda_{ij} x_i x_j \,.$$

The considered optimization problem is equivalent to

$$\max_{x} \min_{\Lambda} L(x, \Lambda) \,,$$

while the optimal Lagrangian dual bound is derived as

$$\min_{\Lambda} \max_{x} L(x, \Lambda) \,.$$

From here it follows that in the dual problem Λ should always be chosen in such a way that the quadratic form of $L(x, \Lambda)$ is negative semidefinite with respect to x (otherwise the inner maximization with respect to x will deliver infinity), while minimization with respect to Λ turns out to be convex non-differentiable [15].

Applications

Perfect Graphs

A graph is called *perfect* if, for all its vertex-induced subgraphs, the clique number is equal to the chromatic number. In this case the inequalities (9) become the equalities:

$$\omega(G) = \vartheta(\bar{G}) = \chi(G) \,.$$

So, both the clique number and the chromatic number can be computed in a polynomial time by means of the ϑ-function. It is also easy to see how the ϑ-function can be used to actually find a maximum clique of a perfect graph. Indeed, a vertex i of a perfect graph G belongs to some maximum clique if and only if ϑ-function of the subgraph induced by the neighborhood $N(i)$ is equal to $\vartheta(G) - 1$. Hence, we can obtain a maximum clique of the graph successively selecting a vertex satisfying this condition and repeating the procedure with the subgraph induced by its neighborhood. Moreover, this simple algorithm can be improved [1,17]. Coloring a perfect graph can be also performed in a polynomial time (see, e. g., [7]).

A graph is perfect if and only if its complementary graph is perfect [7,10]. This means that for any vertex-induced subgraph of a perfect graph there is also the equality between the independence number and the clique partition number. The *strong perfect graph theorem*, proved recently [5], states that a graph is perfect if and only if it does not include an odd hole or an odd antihole as a vertex-induced subgraph. A polynomial-time algorithm for recognizing perfect graphs was also derived on the basis of the strong perfect graph theorem [4].

Improving Upper Bounds for Independence Number

It is worth to consider how well does $\vartheta(G)$ approximate the independence number $\alpha(G)$ for general graphs and whether this approximation can be improved without breaking the polynomial-time computability. It turns out that for random graphs $\vartheta(G)/\alpha(G)$ grows as $O(\sqrt{n}/\log n)$ [2,9]. So, $\vartheta(G)$ does not allow for a fixed approximation guarantee for $\alpha(G)$. Moreover, the maximum independent set problem is known to be hard to approximate (see, e. g., [8]). However, there is a number of approaches to formulate increasingly tight approximations of $\alpha(G)$ based on the ϑ-function. The first one is the "lift-and-project" method by Lovász and Shrijver [12]. The second approach is to express $\alpha(G)$ as a *copositive* program and to use its approximations via semidefinite programming [6]. Finally, we may try to improve the dual bound of (10) and make it closer to the optimum generating *superfluous* quadratic constraints [15,16].

In first two cases one obtains a sequence of semidefinite programs increasing in size, but having non-increasing optimum values, and at some point the optimum value becomes equal to $\alpha(G)$. It comes as no surprise that before achieving the value $\alpha(G)$, in the general case, the size of the program increases exponentially (otherwise it would imply $P = NP$). What is more surprising is that any provable polynomial-time improvement of $\vartheta(G)$ (i. e. a polynomial-time computable function of a graph having a value less than $\vartheta(G)$ whenever $\alpha(G) < \vartheta(G)$) would also imply $P = NP$ [3]. Hence, unless $P = NP$, neither method can deliver, in general case, a value closer to $\alpha(G)$ than $\vartheta(G)$ before the size of the program becomes exponential.

See also

► Copositive Programming

References

1. Alizadeh F (1991) A sublinear-time randomized parallel algorithm for the maximum clique problem in perfect

graphs. In: Proc 2nd ACM-SIAM SODA, San Francisco, CA, pp 188–194
2. Bollobás B (1985) Random graphs. Academic Press, London
3. Busygin S, Pasechnik DV (2006) On NP-hardness of the clique partition – independence number gap recognition and related problems. Discret Math 304(4):460–463
4. Chudnovsky M, Cornuéjols G, Liu X, Seymour P, Vušković K (2006) Recognizing Berge graphs. Combinatorica 25(2):143–186
5. Chudnovsky M, Robertson N, Seymour P, Thomas R (2006) The strong perfect graph theorem. Ann Math 164(1):51–229
6. de Klerk E, Pasechnik D (2002) Approximation of the stability number of a graph via copositive programming. SIAM J Optim 12(4):875–892
7. Grötschel M, Lovász L, Schrijver A (1988) Geometric algorithms and combinatorial optimization. Springer, Berlin
8. Khot S (2001) Improved inapproximability results for max-clique, chromatic number and approximate graph coloring. In: Proc 42nd Annual IEEE Symposium on the Foundations of Computer Science (FOCS), pp 600–609
9. Knuth DE (1994) The sandwich theorem. Elec J Comb 1: 1–48
10. Lovász L (1972) Normal hypergraphs and the perfect graph conjecture. Discret Math 2:253–267
11. Lovász L (1979) On the Shannon capacity of a graph. IEEE Trans Inform Theory IT-25(1):1–7
12. Lovász L, Schrijver A (1991) Cones of matrices and set-functions and 0-1 optimization. SIAM J Optim 1(2):166–190
13. Nesterov Y, Nemirovskii A (1994) Interior point polynomial methods in convex programming. SIAM, Philadelphia
14. Shannon CE (1956) The zero-error capacity of a noisy channel. IRE Trans Inform Theory IT-2(3):8–19
15. Shor NZ (1998) Nondifferentiable optimization and polynomial problems. Kluwer, Dordrecht
16. Shor NZ, Stetsyuk PI (2002) Lagrangian bounds in multi-extremal polynomial and discrete optimization problems. J Glob Optim 23:1–41
17. Yildirim EA, Fan-Orzechowski X (2006) On extracting maximum stable sets in perfect graphs using Lovász's theta function. Comput Optim Appl 33(2–3):229–247

LP Strategy for Interval-Newton Method in Deterministic Global Optimization

YOUDONG LIN, MARK A. STADTHERR
Department of Chemical and Biomolecular Engineering, University of Notre Dame, Notre Dame, USA

Article Outline

Background

The interval-Newton method is a tool for solving a system of nonlinear algebraic equations. It provides the capability to enclose *all* solutions of the equation system that occur within a specified search interval, and to do so with *mathematical and computational certainty*. In the context of optimization, it is generally applied to the deterministic global optimization of bound-constrained problems:

$$\min_{\boldsymbol{x}} \phi(\boldsymbol{x}) \tag{1}$$

$$\boldsymbol{x} \in \boldsymbol{X}^{(0)}. \tag{2}$$

The objective $\phi(\boldsymbol{x})$ is in general a nonconvex function that may have multiple local minima. The interval vector (box) $\boldsymbol{X}^{(0)}$ provides upper and lower bounds on each component of the decision-variable vector $\boldsymbol{x}$. It is assumed here that these bounds are sufficiently wide that the global minimum of $\phi(\boldsymbol{x})$ will occur in the interior of $\boldsymbol{X}^{(0)}$. This means that the stationarity condition $\nabla\phi(\boldsymbol{x}) = \boldsymbol{0}$ can be used in the search for the global minimum. The interval-Newton method can then be applied to enclose all stationary points, one of which is the global minimizer. If only the global minimizer is sought, then interval-Newton is typically combined with some branch-and-bound scheme, so that all stationary points need not actually be found. However, in other applications, such as transition state analysis [22,38] and computation of phase equilibrium [13,37], it may be useful to know all of the stationary points, and the interval-Newton approach provides this capability. For situations in which it is possible that the global minimum will lie on a boundary

of $\boldsymbol{X}^{(0)}$, then the "peeling" process described by Kearfott [19], in which interval-Newton is applied to each of the lower dimensional subspaces that constitute the boundary of $\boldsymbol{X}^{(0)}$, can be used. For more general constrained problems, the interval-Newton method can be applied to the solution of the Karush–Kuhn–Tucker (KKT) conditions or the Fritz-John conditions. A thorough discussion of the use of the interval-Newton approach in global optimization has been given by Hansen and Walster [11]. In recent years, this approach has been used in a number of applications, including computation of fluid phase equilibrium from activity coefficient models [27,36,37,40], cubic equation-of state models [5,12,13,14,40] and statistical associating fluid theory [39], computation of solid-fluid equilibrium [35,41], parameter estimation using standard least squares [7,25] and error-in-variables [8,9], calculation of adsorption in nanoscale pores from a density function theory model [26], transition state analysis [22] and determination of molecular structures [24].

A drawback to the interval-Newton approach, as well as to other approaches for deterministic global optimization, is the potentially high computational cost. One way to improve the efficiency of the interval-Newton method is to more tightly bound the solution set of the linear interval equation system that is at the core of this approach. In this article, we discuss the solution of this linear interval system and describe a bounding strategy [21,23] based on the use of linear programming (LP) techniques. Using this approach it is possible to exactly (within round out) determine the desired bounds on the solution set of the linear interval system. By providing tight interval bounds on the solution set, the goal is to more quickly contract intervals that may contain stationary points, as well as to more quickly identify intervals that contain a unique stationary point or no stationary point, thus leading to an overall improvement in computational efficiency.

Methods

Interval-Newton

Several good introductions to interval computations are available [11,17,19,28]. A real interval X is defined as the set of real numbers lying between (and including) given upper and lower bounds; that is, $X = [a, b] = \{x \in \Re \mid a \le x \le b\}$. A real interval vector $\boldsymbol{X} = (X_1, X_2, \ldots, X_n)^{\mathrm{T}}$ has n real interval components and can be interpreted geometrically as an n-dimensional rectangle or box. Note that in this context uppercase quantities are intervals, and lowercase quantities are real numbers. Basic arithmetic operations with intervals are defined by $X \,\mathrm{op}\, Y = \{x \,\mathrm{op}\, y \mid x \in X, y \in Y\}$, where $\mathrm{op} \in \{+, -, \times, \div\}$. Interval versions of the elementary functions can be similarly defined. It should be emphasized that, when machine computations with interval arithmetic operations are done, as in the procedures outlined below, the endpoints of an interval are computed with a directed (outward) rounding. That is, the lower endpoint is rounded down to the next machine-representable number and the upper endpoint is rounded up to the next machine-representable number. In this way, through the use of interval, as opposed to floating-point arithmetic, any potential rounding error problems are avoided and rigorous enclosures are maintained. Implementations of interval arithmetic and elementary functions are readily available, and recent compilers from Sun Microsystems directly support interval arithmetic and an interval data type.

For an arbitrary function $f(\boldsymbol{x})$, the *interval extension*, $F(\boldsymbol{X})$, encloses all values of $f(\boldsymbol{x})$ for $\boldsymbol{x} \in \boldsymbol{X}$; that is, it encloses the range of $f(\boldsymbol{x})$ over $\boldsymbol{X}$. It is often computed by substituting the given interval $\boldsymbol{X}$ into the function $f(\boldsymbol{x})$ and then evaluating the function using interval arithmetic. This so-called "natural" interval extension may be wider than the actual range of function values, though it always includes the actual range. For the case in which the function is a single-use expression, that is, an expression in which each variable occurs only once, natural interval arithmetic will always yield the true function range. For cases in which such rearrangements are not possible, there are a variety of other approaches that can be used to try to tighten interval extensions [11,17,19,28,29].

Of interest here is the interval-Newton method and its application to the stationarity condition $\nabla\phi(\boldsymbol{x}) = \boldsymbol{0}$. Given an $n \times n$ nonlinear equation system $\boldsymbol{f}(\boldsymbol{x}) = \nabla\phi(\boldsymbol{x}) = \boldsymbol{0}$ with a finite number of real roots in some initial interval, this technique provides the capability to find tight enclosures of *all* the roots of the system that lie within the given initial interval. An outline of the interval-Newton methodology is given here. More details are available elsewhere [11,19,34]. It should be emphasized that this technique is *not* equivalent to simply

implementing the routine "point" Newton method in interval arithmetic.

Given some initial interval $\boldsymbol{X}^{(0)}$, the interval-Newton algorithm is applied to a sequence of subintervals, which arises due to a bisection process. Consider a subinterval $\boldsymbol{X}^{(k)}$ in the sequence. Before application of interval-Newton, measures are usually taken first to try to eliminate, or at least shrink, this subinterval. For example, one may apply a function range test. An interval extension $\boldsymbol{F}(\boldsymbol{X}^{(k)})$ of the function $\boldsymbol{f}(\boldsymbol{x})$ is calculated. If there is any component of the interval extension $\boldsymbol{F}(\boldsymbol{X}^{(k)})$ that does not include zero, then the interval can be discarded, since no solution of $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{0}$ can exist in this interval. The next subinterval in the sequence may then be considered. Otherwise, testing of $\boldsymbol{X}^{(k)}$ continues. A variety of other interval-based techniques (e. g., constraint propagation) may also be applied to try to shrink $\boldsymbol{X}^{(k)}$ before proceeding to the interval-Newton procedure.

In the interval-Newton procedure, the linear interval equation system

$$\boldsymbol{F}'(\boldsymbol{X}^{(k)})(\boldsymbol{N}^{(k)} - \boldsymbol{x}^{(k)}) = -\boldsymbol{f}(\boldsymbol{x}^{(k)}), \tag{3}$$

is solved for a new interval $\boldsymbol{N}^{(k)}$, where $\boldsymbol{F}'(\boldsymbol{X}^{(k)})$ is an interval extension of the Jacobian of $\boldsymbol{f}(\boldsymbol{x})$, and $\boldsymbol{x}^{(k)}$ is an arbitrary point in $\boldsymbol{X}^{(k)}$. It has been shown [11,19,28] that any root contained in $\boldsymbol{X}^{(k)}$ is also contained in the *image* $\boldsymbol{N}^{(k)}$. This implies that if the intersection between $\boldsymbol{X}^{(k)}$ and $\boldsymbol{N}^{(k)}$ is empty, then no root exists in $\boldsymbol{X}^{(k)}$, and also suggests the iteration scheme $\boldsymbol{X}^{(k+1)} = \boldsymbol{X}^{(k)} \cap \boldsymbol{N}^{(k)}$. In addition, it has also been shown [11,19,28] that, if $\boldsymbol{N}^{(k)}$ is in the interior of $\boldsymbol{X}^{(k)}$, then there is a *unique* root contained in $\boldsymbol{X}^{(k)}$ and thus in $\boldsymbol{N}^{(k)}$. Thus, after computation of $\boldsymbol{N}^{(k)}$ from Eq. (3), there are three possibilities: (1) $\boldsymbol{X}^{(k)} \cap \boldsymbol{N}^{(k)} = \varnothing$, meaning there is no root in the current interval $\boldsymbol{X}^{(k)}$ and it can be discarded; (2) $\boldsymbol{N}^{(k)}$ is in the interior of $\boldsymbol{X}^{(k)}$, meaning that there is *exactly* one root in the current interval $\boldsymbol{X}^{(k)}$; (3) neither of the above, meaning that no conclusion can be drawn. In the last case, if $\boldsymbol{X}^{(k)} \cap \boldsymbol{N}^{(k)}$ is sufficiently smaller than $\boldsymbol{X}^{(k)}$, then the interval-Newton test can be reapplied to the resulting intersection, $\boldsymbol{X}^{(k+1)} = \boldsymbol{X}^{(k)} \cap \boldsymbol{N}^{(k)}$. Otherwise, the intersection $\boldsymbol{X}^{(k)} \cap \boldsymbol{N}^{(k)}$ is bisected, and the resulting two subintervals are added to the sequence of subintervals to be tested. If an interval containing a unique root has been identified, then this root can be tightly enclosed by continuing the interval-Newton iteration, which will converge quadratically to a desired tolerance (on the enclosure diameter).

At termination, when the subintervals in the sequence have all been tested, either all the real roots of $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{0}$ have been tightly enclosed, or it is determined that no root exists. Applied to nonlinear equation solving problems, this can be regarded as a type of branch-and-prune scheme on a binary tree. It should be emphasized that the enclosure, existence, and uniqueness properties discussed above, which are the basis of the method, can be derived without making any strong assumptions about the function $\boldsymbol{f}(\boldsymbol{x})$ for which roots are sought. The function must have a *finite* number of roots over the search interval of interest; however, no special properties such as convexity or monotonicity are required, and $\boldsymbol{f}(\boldsymbol{x})$ may have transcendental terms.

Solution of Linear Interval Equation Systems

Clearly, the solution of the linear interval system given by Eq. (3) is essential to the interval-Newton approach. To see the issues involved in solving such a system, consider the general linear interval system $A\boldsymbol{z} = \boldsymbol{B}$, where the matrix A and the right hand side vector $\boldsymbol{B}$ are interval-valued. The solution set S of this system is defined by $S = \left\{\boldsymbol{z} \,\middle|\, \tilde{A}\boldsymbol{z} = \boldsymbol{b}, \tilde{A} \in A, \boldsymbol{b} \in \boldsymbol{B}\right\}$. However, in general this set is not an interval and may have a very complex polygonal geometry. Thus to "solve" the linear interval system, one instead seeks an interval $\boldsymbol{Z}$ containing S. Computing the interval hull (the tightest interval containing S) is NP-hard [33], but there are several methods for determining an interval $\boldsymbol{Z}$ that contains but overestimates S. Various interval-Newton methods differ in how they solve Eq. (3) for $\boldsymbol{N}^{(k)}$ and thus in the tightness with which the solution set is enclosed. By obtaining bounds that are as tight as possible, the overall performance of the interval-Newton approach can be improved, since with a smaller $\boldsymbol{N}^{(k)}$ the volume of $\boldsymbol{X}^{(k)} \cap \boldsymbol{N}^{(k)}$ is reduced, and it is also more likely either that $\boldsymbol{X}^{(k)} \cap \boldsymbol{N}^{(k)} = \varnothing$ that $\boldsymbol{N}^{(k)}$ is in the interior of $\boldsymbol{X}^{(k)}$ will be satisfied. Thus, intervals that may contain solutions of the nonlinear equation system are more quickly contracted, and intervals that contain no solution or that contain a unique solution may be more quickly identified, all of which leads to a likely reduction in the number of bisections needed.

Frequently, $\boldsymbol{N}^{(k)}$ is computed component-wise using an interval Gauss–Seidel approach, preconditioned with an inverse-midpoint matrix. Though the inverse-midpoint preconditioner is a good general-purpose preconditioner, it is not always the most effective approach [18,19]. A hybrid preconditioning approach (HP/RP) [10], which combines a simple pivoting preconditioner with the standard inverse-midpoint scheme, has been shown to be significantly more efficient than the inverse-midpoint preconditioner alone on some applications. However, it still may not yield the tightest enclosure of the solution set, which, as noted above, is in general an NP-hard problem. Nevertheless, it is possible, using an LP-based strategy, to compute exact component-wise bounds on the solution set required in the context of the interval-Newton method, while avoiding exponential time complexity. This method is described next.

LP Strategy for Interval-Newton Method

Many types of methods have been proposed for bounding the solution set of a system of linear interval equations. One such method is based on the use of LP techniques [1,3,15,17]. Consider again the linear interval system $A\boldsymbol{z} = \boldsymbol{B}$. Oettli and Prager [31] showed that the solution set S is determined by the constraints:

$$\left|\hat{A}\boldsymbol{z} - \hat{\boldsymbol{B}}\right| \leq \Delta A\,|\boldsymbol{z}| + \Delta\boldsymbol{B}, \tag{4}$$

where $\hat{A}$ is the component-wise midpoint matrix of the interval matrix A, ΔA is the component-wise half-width (radius) matrix of A, and similarly $\hat{\boldsymbol{B}}$ and $\Delta\boldsymbol{B}$ are the midpoint and radius of $\boldsymbol{B}$. Eq. (4) is not directly useful for computing bounds on the solution set because of the absolute value operation on the right-hand side. In general, the solution may lie in all 2^n orthants for an n-dimensional problem. In each orthant, each component of $\boldsymbol{z}$ keeps a constant sign, and thus the absolute value can be dropped. For a given orthant, define the diagonal matrix D_α by

$$(D_\alpha)_{jj} = \begin{cases} 1 & z_j \geq 0 \\ -1 & z_j \leq 0 \end{cases} \quad j = 1, 2, \ldots, n. \tag{5}$$

Thus $|\boldsymbol{z}| = D_\alpha \boldsymbol{z}$ and $\boldsymbol{z} = D_\alpha\,|\boldsymbol{z}|$. Eq. (4) becomes:

$$\left|\hat{A}\boldsymbol{z} - \hat{\boldsymbol{B}}\right| \leq \Delta A D_\alpha \boldsymbol{z} + \Delta\boldsymbol{B}. \tag{6}$$

This can be rearranged to the set of linear inequalities

$$\begin{pmatrix} \hat{A} - \Delta A D_\alpha \\ -\hat{A} - \Delta A D_\alpha \end{pmatrix} \boldsymbol{z} \leq \begin{pmatrix} \overline{\boldsymbol{B}} \\ -\underline{\boldsymbol{B}} \end{pmatrix}, \tag{7}$$

where the underline and overline denote lower and upper interval bounds, respectively. To determine the tightest interval enclosing the solution set, one can then solve, in each orthant, the set of $2n$ optimization problems

$$\max_{\boldsymbol{z}} z_j, \qquad j = 1, 2, \ldots, n, \tag{8}$$

$$\min_{\boldsymbol{z}} z_j, \qquad j = 1, 2, \ldots, n, \tag{9}$$

each with the $2n$ linear inequality constraints given by Eq. (7). These can be solved using linear programming (LP) techniques. However, in general, there are 2^n orthants and so the solution time complexity will be exponential, as expected since this problem is known to be NP-hard.

In the context of the interval-Newton method, however, the exponential time complexity can be avoided. This is because only that part of the solution set of Eq. (3) that intersects $\boldsymbol{X}^{(k)}$ needs to be found. Consider the choice of the real point $\boldsymbol{x}^{(k)}$ in Eq. (3). Here $\boldsymbol{x}^{(k)}$ is an arbitrary point in $\boldsymbol{X}^{(k)}$ typically taken to be the midpoint. However, if $\boldsymbol{x}^{(k)}$ is chosen to be a corner of $\boldsymbol{X}^{(k)}$ instead, then the part of the solution set for $\boldsymbol{N}^{(k)} - \boldsymbol{x}^{(k)}$ of Eq. (3) that intersects $\boldsymbol{X}^{(k)}$ lies in just one orthant. Thus, in the context of interval-Newton, only $2n$ LP subproblems, each with $2n$ constraints, needs to be solved. Furthermore, the LP subproblems have properties that can be exploited. First, all the $2n$ subproblems share the same constraints; that is, they all have the same feasible region. Thus, an initial feasible basis for the LP subproblems needs to be found only once. Second, the objective function of each subproblem consists of just one variable. This makes the problem much simpler since it is not necessary, as it is in the general case, to calculate the gain in objective value when choosing variables to enter and exit the basis.

Lin and Stadtherr [23] have implemented the approach outlined above in the procedure LISS_LP (Linear Interval System Solver by Linear Programming), and incorporated it in an interval-Newton method for global optimization. Two key aspects of the implementation are:

1. The choice of a corner of $\boldsymbol{X}^{(k)}$ to be used in the LP problem. For this purpose, a heuristic approach [23] was developed that incorporates ideas from the pivoting preconditioner approach of Gau and Stadtherr [10].
2. Determination of rigorous error bounds on the solution of the LP problems. This is done using a procedure based on primal/dual relationships [16,23,30]. Complete details of the implementation are given by Lin and Stadtherr [23]. We turn next to some examples that demonstrate the performance of the LP-based strategy as implemented in LISS_LP.

Cases

Lin and Stadtherr [21,23] have tested the performance of the LP-based interval-Newton strategy for global optimization on a variety of problems. In this section, we summarize the results on a group of parameter estimation problems. Each parameter estimation case used was formulated using the error-in-variables approach, with complete details given by Gau and Stadtherr [8,9]. Comparisons are made to an interval Gauss–Seidel method with a hybrid preconditioning approach (HP/RP), which has been shown [10] to provide a substantial improvement in computational performance relative to standard implementations of the interval-Newton approach. Comparisons are made in terms of the number of interval-Newton (I-N) tests required, i. e., the number of times Eq. (3) must be solved, and in terms of the CPU time on a Sun Blade 1000 Model 1600 workstation. On a current (early 2007) workstation, these CPU times would be approximately an order of magnitude less.

Problem 1

This problem [6,20] involves estimation of binary parameters in the van Laar equation for activity coefficients. These two parameters are estimated from vapor-liquid equilibrium data for the binary system of methanol and 1,2-dichloroethane. Computational performance results are shown in Table 1. When the LP-based strategy (LISS_LP) is applied, the number of I-N tests is substantially reduced relative to HP/RP, indicating its effectiveness in reducing the number of intervals that must be tested. Essentially, by reducing the size of $\boldsymbol{N}^{(k)}$, the LP approach is able to more quickly identify and discard intervals that do not contain a stationary point. However, the percent reduction in overall CPU time is less than the percent reduction in I-N tests. This occurs due to the overhead in solving the LP subproblems.

LP Strategy for Interval-Newton Method in Deterministic Global Optimization, Table 1
Computational performance of LP-based method (LISS_LP) and preconditioned interval Gauss–Seidel method (HP/RP) on a Sun Blade 1000 Model 1600

Problem	Variables (n)	HP/RP		LISS_LP	
		I-N Tests	CPU time (s)	I-N Tests	CPU time (s)
1	12	303589	664.4	156182	496.7
2	264	220	1357.3	81	504.9
3	22	9505	24.0	1258	12.7
4	32	144833	976.2	24817	837.2
5	59	55255	2315.9	9757	1692.4

Problem 2

In this problem [4], the rating parameters are estimated for a steady-state heat exchanger network, which consists of four heat exchangers. The four parameters can be estimated from experimental measurements, including six flow measurements and thirteen temperature measurements. In the version of the problem considered here, 20 data points were considered, leading to an optimization problem involving 264 independent variables. Due to the large number of variables, sparse linear programming routines were implemented in LISS_LP for this problem. In this case, both I-N tests and CPU time are substantially reduced when the LP-based method is used, as shown in Table 1, indicating that the LP overhead is less significant on this relatively large problem.

Problems 3 and 4

Both of these problems involve the estimation of kinetic parameters for an irreversible, first-order reaction $A \rightarrow B$. In Problem 3 [6,20], data from an adiabatic continuous-stirred-tank reactor (CSTR) is used, and in Problem 4 [2] data from an isothermal batch reactor is used. In both cases, the reaction rate constant k is given by an Arrhenius expression

$$k = \theta_1 \exp\left(-\frac{\theta_2}{T}\right), \tag{10}$$

in which two parameters, θ_1 and θ_2, must be determined from experimental measurements. Again, the computational performance results (Table 1) show that use of the LP-based strategy substantially reduces the number of I-N tests required relative to HP/RP, but that a comparable reduction in CPU time is not achieved. For example, on Problem 4, the number of I-N tests is reduced by nearly a factor of 6, but there is only about a 15% reduction in CPU time. This reflects the fact that an I-N test performed using the LP method requires greater computational effort than an I-N test using the HP/RP approach. However, on problems studied by Lin and Stadtherr [21,23], this overhead was always offset by a large reduction in the number of I-N tests, resulting in computational savings on all but very small problems.

Problem 5

In this problem, parameters are estimated in a model of an isothermal pseudo-differential reactor for the catalytic hydrogenation of phenol on a palladium catalyst [32]. There are 28 experimental kinetic data points of the partial pressure of phenol (P_1), the partial pressure of hydrogen (P_2), and the initial reaction rate (r). It is desired to fit this kinetic data to a semi-empirical model of the form

$$r = \frac{\theta_1 \theta_2^2 \theta_3 P_1 P_2^2}{(1 + \theta_1 P_1 + \theta_2 P_2)^3}, \tag{11}$$

where θ_1, θ_2 and θ_3 are the parameters to be estimated. This global optimization problem has 59 independent variables. Due to the relatively large number of variables in this problem, a sparse linear programming routine was used in LISS_LP. As seen in Table 1, both I-N tests and CPU time are reduced nicely compared to HP/RP when the LP-based method is used. As in the case of Problem 2, on this relatively large problem the impact of the LP overhead appears to be less significant.

Conclusions

In this article, we have described an LP-based strategy [21,23] for solving the linear interval equation system arising in the context of the interval-Newton approach for deterministic global optimization. The method can obtain tighter bounds on the solution set of the linear interval system than the preconditioned interval Gauss–Seidel approach, and thus leads to a large reduction in the number of subintervals that must be tested during the interval-Newton procedure. However, the difference between the overhead required to solve the LP subproblems and that required to perform the preconditioned Gauss–Seidel method may lead to relatively smaller or larger improvements in overall computational time, depending on the size of the problem. With sparse linear algebra in the LP subproblems, the method can be successfully applied to deterministic global optimization problems involving over two hundred variables, providing a rigorous guarantee of global optimality.

References

1. Aberth O (1997) The solution of linear interval equations by a linear programming method. Lin Algebr Appl 259:271–279
2. Bard Y (1974) Nonlinear Parameter Estimation. Academic Press, New York
3. Beaumont O (1998) Solving interval linear systems with linear programming techniques. Lin Algebr Appl 281:293–309
4. Biegler LT, Tjoa IB (1980) A parallel implementation for parameter estimation with implicit models. Anns Opns Res 42:1–23
5. Burgos-Solórzano GI, Brennecke JF, Stadtherr MA (2004) Validated computing approach for high-pressure chemical and multiphase equilibrium. Fluid Phase Equilib 219:245–255
6. Esposito WR, Floudas CA (1998) Parameter estimation of nonlinear algebraic models via the error-in-variables approach. Ind Eng Chem Res 37:1841–1858
7. Gau CY, Brennecke JF, Stadtherr MA (2000) Reliable nonlinear parameter estimation in VLE modeling. Fluid Phase Equilib 168:1–18
8. Gau CY, Stadtherr MA (2000) Reliable nonlinear parameter estimation using interval analysis: Error-in-variable approach. Comput Chem Eng 24:631–637
9. Gau CY, Stadtherr MA (2002) Deterministic global optimization for error-in-variables parameter estimation. AIChE J 48:1192–1197
10. Gau CY, Stadtherr MA (2002) New interval methodologies for reliable chemical process modeling. Comput Chem Eng 26:827–840
11. Hansen ER, Walster GW (2004) Global Optimization Using Interval Analysis. Marcel Dekker, New York
12. Hua JZ, Brennecke JF, Stadtherr MA (1996) Reliable prediction of phase stability using an interval-Newton method. Fluid Phase Equilib 116:52–59
13. Hua JZ, Brennecke JF, Stadtherr MA (1998) Enhanced interval analysis for phase stability: Cubic equation of state models. Ind Eng Chem Res 37:1519–1527

14. Hua JZ, Brennecke JF, Stadtherr MA (1998) Reliable computation of phase stability using interval analysis: Cubic equation of state models. Comput Chem Eng 22:1207–1214
15. Jansson C (1997) Calculation of exact bounds for the solution set of linear interval systems. Lin Algebr Appl 251:321–340
16. Jansson C (2004) A rigorous lower bound for the optimal value of convex optimization problems. J Glob Optim 28:121–137
17. Jaulin L, Kieffer M, Didrit O, Walter É (2001) Applied Interval Analysis. Springer, London
18. Kearfott RB (1990) Preconditioners for the interval Gauss–Seidel method. SIAM J Numer Anal 27:804–822
19. Kearfott RB (1996) Rigorous Global Search: Continuous Problems. Kluwer, Dordrecht
20. Kim I, Leibman M, Edgar T (1990) Robust error-in-variables estimation using nonlinear programming techniques. AIChE J 36:985–993
21. Lin Y, Stadtherr MA (2004) Advances in interval methods for deterministic global optimization in chemical engineering. J Glob Optim 29:281–296
22. Lin Y, Stadtherr MA (2004) Locating stationary points of sorbate-zeolite potential energy surfaces using interval analysis. J Chem Phys 121:10159–10166
23. Lin Y, Stadtherr MA (2004) LP strategy for the interval-Newton method in deterministic global optimization. Ind Eng Chem Res 43:3741–3749
24. Lin Y, Stadtherr MA (2005) Deterministic global optimization of molecular structures using interval analysis. J Comput Chem 26:1413–1420
25. Lin Y, Stadtherr MA (2006) Deterministic global optimization for parameter estimation of dynamic systems. Ind Eng Chem Res 45:8438–8448
26. Maier RW, Stadtherr MA (2001) Reliable density-functional-theory calculations of adsorption in nanoscale pores. AIChE J 47:1874–1884
27. McKinnon KIM, Millar CG, Mongeau M (1996) Global optimization for the chemical and phase equilibrium problem using interval analysis. In: Floudas CA Pardalos PM (eds) State of the Art in Global Optimization: Computational Methods and Applications. Kluwer, Dordrecht, pp 365–382
28. Neumaier A (1990) Interval Methods for Systems of Equations. Cambridge University Press, Cambridge
29. Neumaier A (2003) Taylor forms – Use and limits. Reliable Comput 9:43–79
30. Neumaier A, Shcherbina O (2004) Safe bounds in linear and mixed-integer programming. Math Prog 99:283–296
31. Oettli W, Prager W (1964) Compatibility of approximate solution of linear equation with given error bounds for coefficients and right-hand sides. Numer Math 6:405–408
32. Rod V, Hancil V (1980) Numerical methods for estimating parameters in nonlinear models with errors in the variables. Technometrics 27:33
33. Rohn J, Kreinovich V (1995) Computing exact componentwise bounds on solution of linear systems with interval data is NP-hard. SIAM J Matrix Anal 16:415–420
34. Schnepper CA, Stadtherr MA (1996) Robust process simulation using interval methods. Comput Chem Eng 20(2):187–199
35. Scurto AM, Xu G, Brennecke JF, Stadtherr MA (2003) Phase behavior and reliable computation of high-pressure solid-fluid equilibrium with cosolvents. Ind Eng Chem Res 42:6464–6475
36. Stadtherr MA, Schnepper CA, Brennecke JF (1995) Robust phase stability analysis using interval methods. AIChE Symp Ser 91(304):356
37. Tessier SR, Brennecke JF, Stadtherr MA (2000) Reliable phase stability analysis for excess Gibbs energy models. Chem Eng Sci 55:1785–1796
38. Westerberg KM, Floudas CA (1999) Locating all transition states and studying the reaction pathways of potential energy surfaces. J Chem Phys 110:9259–9295
39. Xu G, Brennecke JF, Stadtherr MA (2002) Reliable computation of phase stability and equilibrium from the SAFT equation of state. Ind Eng Chem Res 41:938–952
40. Xu G, Haynes WD, Stadtherr MA (2005) Reliable phase stability analysis for asymmetric models. Fluid Phase Equilib 235:152–165
41. Xu G, Scurto AM, Castier M, Brennecke JF, Stadtherr MA (2000) Reliable computational of high-pressure solid-fluid equilibrium. Ind Eng Chem Res 39:1624–1636

L-shaped Method for Two-stage Stochastic Programs with Recourse

FRANCOIS LOUVEAUX[1], JOHN R. BIRGE[2]
[1] Université Namur, Namur, Belgium
[2] Northwestern University, Evanston, USA

MSC2000: 90C15

Article Outline

Keywords

Stochastic programming; Recourse; Decomposition techniques

A *stochastic linear program with recourse* (SLP) is a mathematical program of the form

$$\begin{cases} \min & c \cdot x + Q(x) \\ \text{s.t.} & Ax = b, \quad x \geq 0, \end{cases}$$

where $Q(x) = \mathsf{E}_\xi Q(x, \xi)$,

$$Q(x, \xi) = \begin{cases} \min & q, y(\xi) \\ \text{s.t.} & W \cdot y(\xi) = h - T \cdot x, y \geq 0, \end{cases}$$

and E_ξ denotes the mathematical expectation with respect to ξ, x is an $(n_1 \times 1)$ decision vector, and for each ξ, y is $(n_2 \times 1)$. A is $(m_1 \times n_1)$ and for each ξ, h is $(m_2 \times 1)$. All other matrices and vectors have conformable dimensions. Transposes are omitted for simplicity. The random vector ξ is formed by the random components of q, h and $T \cdot Q(x, \xi)$ is the second-stage value function for a given ξ and $Q(x)$ the expected value-function or expected recourse.

In the case where random vectors are described by discrete distributions, $Q(x)$ is a piecewise linear convex function of x, so that classical decomposition techniques may apply. Let $k = 1, \ldots, K$ index the possible realizations of ξ, let p_k be their probabilities and y_k be the corresponding second stage decision variables. SLP is then equivalent to the *extensive form*

$$\text{(EF)} \quad \begin{cases} \min & cx + \sum_{k=1}^{K} p_k q_k y_k \\ \text{s.t.} & Ax = b, \\ & T_k x + W y_k = h_k, \\ & \quad k = 1, \ldots, K, \\ & x, y_k \geq 0. \end{cases}$$

This extensive form possesses a *dual block-angular structure*. It is thus well suited to application of *Benders decomposition*, which in the case of stochastic programming is known as the *L-shaped method*. An abbreviated presentation is as follows. It is restricted to the case where all second stage programs are bounded and feasible for any choice of first-stage decision.

L-Shaped Method for Two-Stage Stochastic Program with Bounded, Complete Recourse

Consider the master linear program

$$\text{(MLP)} \quad \begin{cases} \min & cx + \theta \\ \text{s.t.} & Ax = b, \\ & \mathsf{E}_j x + \theta \geq e_j, \quad j = 1, \ldots, s, \\ & x \geq 0, \end{cases}$$

with s *optimality cuts* (initially $s = 0$) and θ a lower bound on $Q(x)$, (θ is omitted when $s = 0$).

Using the solution x^s, θ^s of (MLP) at iteration s, find optimal solutions to the K subproblems,

$$\begin{cases} \min & w = q_k y \\ \text{s.t.} & Wy = h_k - T_k x^s, \\ & y \geq 0, \end{cases}$$

with optimal simplex multipliers π_k^s, $k = 1, \ldots, K$.

Define $E_{s+1} = \sum_{k=1}^{K} p_k \pi_k^s T_k$ and $e_{s+1} = \sum_{k=1}^{K} p_k \pi_k^s h_k$.

If $e_{s+1} - E_{s+1} x^s \leq \theta^s$, then stop as x^s is an optimal solution. Otherwise, set $s = s + 1$ and return to the master program.

Finite convergence of the L-shaped method is proved through classical convexity arguments. When the second stage does not have complete recourse, some first stage decisions may imply that no feasible recourse exists for some k. Then, a number of *feasibility cuts* are also needed. They are obtained through the optimal simplex multipliers of some phase-1 problem. Although these cuts should theoretically be generated for all realizations $k = 1, \ldots, K$, there are many situations where the search for feasibility cuts can be limited to one selected second-stage only [9].

The L-shaped method can be made more efficient by performing some bunching to obtain optimal multipliers for several realizations of ξ at once (see the experiments in [4]). Efficiency can sometimes be gained by sending disaggregated cuts (also called multicuts) instead of one fully aggregated cut at each iteration [2]. Another way of improving the efficiency of the L-shaped is to include a quadratic regularizing term in the first-stage objective function. This additional term is typically the square of the Euclidean distance between

the decision x and the previous iterate point x^s [8]. *L*-shaped methods have also been combined with statistical estimation, in particular to cope with continuous random variables (see [5] and ► Discretely distributed stochastic programs: Descent directions and efficient points).

A number of alternatives to the *L*-shaped techniques have been proposed to solve SLP. One is to use the *Lagrangian finite generation method*, also known as *scenario aggregation* [7]. Another is to use interior points techniques [1]. For a general presentation of stochastic programming, see [3] or [6].

See also

- ► Approximation of Extremum Problems with Probability Functionals
- ► Approximation of Multivariate Probability Integrals
- ► Discretely Distributed Stochastic Programs: Descent Directions and Efficient Points
- ► Extremum Problems with Probability Functions: Kernel Type Solution Methods
- ► General Moment Optimization Problems
- ► Logconcave Measures, Logconvexity
- ► Logconcavity of Discrete Distributions
- ► Multistage Stochastic Programming: Barycentric Approximation
- ► Preprocessing in Stochastic Programming
- ► Probabilistic Constrained Linear Programming: Duality Theory
- ► Probabilistic Constrained Problems: Convexity Theory
- ► Simple Recourse Problem: Dual Method
- ► Simple Recourse Problem: Primal Method
- ► Stabilization of Cutting Plane Algorithms for Stochastic Linear Programming Problems
- ► Static Stochastic Programming Models
- ► Static Stochastic Programming Models: Conditional Expectations
- ► Stochastic Integer Programming: Continuity, Stability, Rates of Convergence
- ► Stochastic Integer Programs
- ► Stochastic Linear Programming: Decomposition and Cutting Planes
- ► Stochastic Linear Programs with Recourse and Arbitrary Multivariate Distributions
- ► Stochastic Network Problems: Massively Parallel Solution
- ► Stochastic Programming: Minimax Approach
- ► Stochastic Programming Models: Random Objective
- ► Stochastic Programming: Nonanticipativity and Lagrange Multipliers
- ► Stochastic Programming with Simple Integer Recourse
- ► Stochastic Programs with Recourse: Upper Bounds
- ► Stochastic Quasigradient Methods in Minimax Problems
- ► Stochastic Vehicle Routing Problems
- ► Two-stage Stochastic Programming: Quasigradient Method
- ► Two-stage Stochastic Programs with Recourse

References

1. Birge JR, Holmes DF (1992) Efficient solution of two-stage stochastic linear programming using interior point methods. Comput Optim Appl 1:245–276
2. Birge JR, Louveaux FV (1988) A multicut algorithm for two-stage stochastic linear programs. Europ J Oper Res 34:384–392
3. Birge JR, Louveaux FV (1997) Introduction to stochastic programming. Springer, Berlin
4. Gassmann HI (1990) MSLiP: A computer code for the multistage linear programming problem. Math Program 47:407–423
5. Higle EJ, Sen S (1991) Stochastic decomposition: An algorithm for two stage linear programs with recourse. Math Oper Res 16:650–669
6. Kall P, Wallace SW (1994) Stochastic programming. Wiley, New York
7. Rockafellar RT, Wets RJ-B (1986) A Lagrangian finite generation technique for solving linear-quadratic problems in stochastic programming. Math Program Stud 23:63–96
8. Ruszczyński A (1986) A regularized decomposition for minimizing a sum of polyhedral functions. Math Program 35:309–333
9. Walkup D, Wets RJ-B (1967) Stochastic programs with recourse. SIAM J Appl Math 15:1299–1314