Lagrangian Relaxation

Claude Lemaréchal

Inria, 655 avenue de l'Europe, Montbonnot, 38334 Saint Ismier, France
Claude.Lemarechal@inrialpes.fr

Abstract. Lagrangian relaxation is a tool to find upper bounds on a given (arbitrary) maximization problem. Sometimes, the bound is exact and an optimal solution is found. Our aim in this paper is to review this technique, the theory behind it, its numerical aspects, its relation with other techniques such as column generation.

1 Introduction, Motivation

1.1 Scope of the Paper

One of the basic techniques in combinatorial optimization is *bounding*: given a (finite but) nasty set $S \subset \mathbb{R}^n$ and an "easy" function f (say linear or quadratic), find an upper bound for the optimal value of the problem

$$\max f(x)\,, \quad x \in S\,.$$

Lagrangian relaxation is a universal technique for that. To make it applicable, the feasible set S must be written as $S = \mathcal{X} \cap \{x : c(x) = 0\}$, where $\mathcal{X}$ is "easy" in terms of maximizing f over it, while the constraints $c = (c_1, \ldots, c_m)$ are "complicating". *In fine*, the technique produces a bunch of optimal solutions to the problem $\max_{x \in \bar{S}} f(x)$, where $\bar{S}$ is a certain convex enlargement of S. For a general introduction and motivation of this technique, see for example [25, Chap. XII]; or also [19], [63, Chap. 6] for its application to combinatorial problems.

We will first describe in §1.2 what Lagrangian relaxation is. Then a list of possible applications (§2) will illustrate its versatility. Section 3 will be devoted to theory; in particular we will study the relations between the original problem and its relaxed version, i.e. between S and $\bar{S}$. The end of the paper will be devoted to numerical algorithms.

Our development may be deemed too technical at places. In particular, we will not necessarily assume S to be finite. As a result, we will have to care about continuity and boundedness, while combinatorics belong to the polyhedral world, with everything contained in the unit cube. Indeed, there are several reasons for our attitude:

(i) Perhaps paradoxically, it helps to think in abstract terms, forgetting as much as possible about linear algebra. The latter may make life easier, in particular skipping analysis and calculus; however this has its price: linear algebra is often "the tree hiding the forest", which prevents a sound understanding of what one is doing.

M. Jünger and D. Naddef (Eds.): Computat. Comb. Optimization, LNCS 2241, pp. 112–156, 2001.

(ii) SDP optimization, which has entered recently the combinatorial world (see [1] for example), is a form of Lagrangian relaxation; and of course, SDP optimization is definitely beyond the polyhedral world.
(iii) Anyway, Lagrangian relaxation has a much larger field of applicability than combinatorial problems, and it does not harm to care about mathematical rigor.

1.2 The Basic Idea of Lagrangian Relaxation

The present section introduces the mechanism of Lagrangian relaxation as a *simple* yet *general* tool. Consider first an optimization problem put in the abstract form

$$\max f(x)\,, \quad x \in \mathcal{X}\,, \quad c_j(x) = 0,\ j = 1, \ldots, m\,, \tag{1}$$

hereafter called the primal problem. We introduce the *Lagrangian*, a function of the primal variable x and of the *dual variable* $u \in \mathbb{R}^m$:

$$\mathcal{X} \times \mathbb{R}^m \ni (x,u) \mapsto L(x,u) := f(x) - \sum_{j=1}^{m} u_j c_j(x) = f(x) - u^\top c(x)\,, \tag{2}$$

where the last equality introduces the notation c for the m-vector of constraint-values. In plain words, the Lagrangian replaces each constraint by a linear "price" to be paid or received, according to the sign of u_j. Maximizing $L(\cdot, u)$ over the whole of $\mathcal{X}$ is therefore closely related to solving (1). In a sense, $\mathcal{X}$ can be considered as the "universe" in which x lives, while c represents the constraints that are "relaxed".

Definition 1 *The* dual function *associated with* (1), (2) *is the function of* u *defined by*

$$\mathbb{R}^m \ni u \mapsto \theta(u) := \max_{x \in \mathcal{X}} L(x,u)\,. \tag{3}$$

The dual problem *is then*[1]

$$\min \theta(u)\,, \quad u \in \mathbb{R}^m\,. \tag{4}$$

□

Lagrangian relaxation is a very versatile technique, which accepts essentially *any* data $\mathcal{X}, f, c$. The only really crucial assumption is the following, pragmatic but fundamental; it will be in force throughout:

[1] Notationally, we should write sup and inf throughout instead of min and max, since nothing guarantees that the various suprema are attained. Despite our care for mathematical rigor, we will neglect this subtlety and use min and max uniformly.

Note also that minimizing θ excludes the value $\theta(u) = +\infty$. As a result, the dual problem entails possible *dual constraints*, preventing the value $+\infty$ in (3).

Assumption 2 *Solving* (1) *is "difficult", while solving* (3) *is "easy". An* oracle *is available which solves* (3) *for given* $u \in \mathbb{R}^m$. □

Lagrangian relaxation then tries to take advantage of this, and aims at finding an appropriate u. To explain why finding an appropriate u amounts to solving the dual problem (4), observe the following triviality: by definition of θ,

$$\theta(u) \geqslant f(x) \quad \text{for all } x \text{ feasible in (1) and all } u \in \mathbb{R}^m \ ,$$

simply because $u^\top c(x) = 0$. This relation is known as *weak duality*, which gives two motivations for the dual problem:

– Because each $\theta(u)$ bounds from above the optimal cost in (1), minimizing θ amounts to finding the best possible such bound – a very relevant idea when bounding is at stake.
– The other motivation is less known, perhaps more important, and certainly more subtle. Suppose we do want to solve (1) with the help of (3). In other words, suppose we want a u such that (3) produces some argmax x_u of $L(\cdot, u)$ which also solves (1). In particular, this x_u must be feasible in (1) and therefore satisfy $\theta(u) = f(x_u)$. In view of weak duality, we see that the only chance for our u is to minimize θ.

Remark 3 *Suppose a problem with inequality constraints:*

$$\max f(x)\,, \quad x \in \mathcal{X}\,, \quad c_j(x) \leqslant 0,\ j = 1, \ldots, m\,.$$

To get the form (1)*, introduce slack variables* s *and use the flexibility of Lagrangian relaxation: set* $\mathcal{X}' := \mathcal{X} \times \mathbb{R}^m_+$*, so that we have to dualize*

$$\max f(x)\,, \quad x \in \mathcal{X}, s \geqslant 0 \in \mathbb{R}^m\,, \quad c(x) + s = 0 \in \mathbb{R}^m\,.$$

The Lagrangian is

$$L'(x, s, u) = f(x) - u^\top(c(x) + s) = L(x, u) - u^\top s$$

where $L(x, u)$ *is the "ordinary" Lagrangian (as though we had equalities). The resulting dual function* θ' *is clearly*

$$\theta'(u) = \begin{cases} +\infty & \text{if some } u_j < 0\,, \\ \theta(u) & \text{otherwise}\,, \end{cases}$$

where θ *is the "ordinary" dual function. In a word, the dual problem becomes* $\min_{u \geqslant 0} \theta(u)$*: the nonpositive part of the dual space is chopped off, we have an immediate illustration of note 1, p. 113.* □

To bypass the above reasoning, it suffices to memorize the following rule:
– consider a *maximization* problem
– and dualize a constraint c by *subtracting* the term $uc(x)$;
– if the constraint was of the *lower than* type,

– then the corresponding multiplier u must be *nonnegative*
– (one may prefer a minimization problem, with a + sign in the Lagrangian, and $\leqslant$-type constraints; the multipliers must again be nonnegative; memorizing the appropriate rule is a matter of taste).

Along the lines of Remark 3, suppose that one insists on having f linear in (1): adopt the formulation in the new universe $\mathcal{X} \times \mathbb{R}$

$$\max s_0 \,, \quad s_0 \leqslant f(x) \,, \quad x \in \mathcal{X} \,, \quad c(x) = 0 \in \mathbb{R}^m \,.$$

A first possibility is to insert the constraint $s_0 \leqslant f(x)$ in the universe: we form the Lagrangian $s_0 - u^\top c(x)$, whose maximum over $x \in \mathcal{X}$ and $s_0 \leqslant f(x)$ is clearly $\theta(u)$; nothing is changed.

Alternatively, we can dualize that constraint, forming the Lagrangian

$$\mathcal{X} \times \mathbb{R} \times \mathbb{R}^{m+1} \ni (x, s_0, u, u_0) \mapsto L'(x, s_0, u, u_0) := s_0 - u_0[s_0 - f(x)] - u^\top c(x) \,.$$

Maximizing it with respect to s_0 (unconstrained!) gives $+\infty$ if $u_0 \neq 1$; and for $u_0 = 1$, its maximum with respect to $x \in \mathcal{X}$ gives $\theta(u)$; again nothing is changed.

Remark 4 *As a result, there is no loss of generality in assuming f linear in* (1). *In case the given objective is really nonlinear, just use the formulation deemed the most convenient (with or without s_0); they all give the same dual anyway.* □

2 Examples

The examples of this section are intended to familiarize the reader with the duality mechanism, but also to show its versatility. Indeed, the simple principles of §1.2 above cover a wide range of techniques; see also [65, Chap. 4] for a number of applications issued from operations research.

2.1 Linear Programming

Take a linear program in standard form:

$$\max b^\top x \,, \quad x \geqslant 0 \in \mathbb{R}^n \,, \quad Ax = a \in \mathbb{R}^m \,.$$

Set first $\mathcal{X} := \mathbb{R}^n_+$, $c(x) := Ax - a$: with $u \in \mathbb{R}^m$, we obtain

$$L(x, u) = (b - A^\top u)^\top x + a^\top u \,, \quad \theta(u) = \begin{cases} a^\top u & \text{if } b - A^\top u \leqslant 0 \in \mathbb{R}^n, \\ +\infty & \text{otherwise} \,. \end{cases}$$

The dual problem is therefore to minimize $a^\top u$, subject to $A^\top u \geqslant b$.

Now set $\mathcal{X} := \mathbb{R}^n$ and dualize all constraints: $Ax - a = 0$ (dual variable $u \in \mathbb{R}^m$) and $-x \leqslant 0$ (dual variable $v \in \mathbb{R}^n$). We obtain the Lagrangian $L'(x, u, v) =$

$(b-A^\top u+v)^\top x+a^\top u$, to be minimized over x unconstrained. The dual function is therefore

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

Remembering that we also have $v\geqslant 0$, either from Remark 3 or from the rule following it, the dual problem is to minimize $a^\top u$ subject to $b-A^\top u=-v\leqslant 0$, which is exactly as before.

We leave it as an exercise to dualize in various ways various forms of LP, and to realize in each case that the selected duality scheme has little influence on the resulting dual problem. For example, any conceivable duality scheme for

$$\max b^\top x\,,\quad x\geqslant 0\,,\quad -\delta e\leqslant Ax-a\leqslant \delta e \tag{5}$$

($e\in\mathbb{R}^m$ is the vector of all ones, $\delta\geqslant 0$) eventually boils down to

$$\min a^\top u+\delta\sum_{j=1}^m |u_j|\,,\quad A^\top u\leqslant a\,;$$

the simplest scheme is to set $Ax-a=s$ and to dualize just that constraint.

2.2 Quadratic Programming

An example hardly more complicated is

$$\max b^\top x-\tfrac{1}{2}x^\top Qx\,,\quad Ax\leqslant a\,,$$

where Q is a symmetric matrix. Take $u\geqslant 0$ and form the Lagrangian

$$L(u,x)=(b-A^\top u)^\top x-\tfrac{1}{2}x^\top Qx+a^\top u\,.$$

To maximize it with respect to x unconstrained, one must consider three cases:

– If Q is positive definite, there is a unique maximum $x_u=Q^{-1}(b-A^\top u)$; it produces the quadratic dual function

$$\theta(u)=\tfrac{1}{2}u^\top AQ^{-1}A^\top u+(a-Ab)^\top u+\tfrac{1}{2}b^\top Q^{-1}b\,,$$

which is nicely convex and must be minimized over $u\geqslant 0$.

– If Q is positive semidefinite, the Lagrangian has a finite maximum only if $b-A^\top u$ lies in the range of Q (so that the equation $Qx=b-A^\top u$ has a solution). This results in extra linear equality constraints in the dual problem.

– If Q is indefinite, the "maximum" is always at infinity: $\theta(u)=+\infty$ for all u; the dual problem produces no better upper bound than $+\infty$.

This example reveals a situation (the case of Q indefinite, or even $Q\prec 0$) where Lagrangian relaxation is of little help for finding upper bounds. Of course, the trouble here comes from a highly nonconvex primal problem; this will be seen in more detail in §3.

2.3 Max-cut, Max-stable, and Quadratic Constraints

Here we apply Lagrangian relaxation to two special combinatorial optimization problems. Both are particular instances of the general class (9) below, which also lends itself to the approach.

The max-cut problem can be written

$$\min \sum_{i,j=1}^{n} Q_{ij} x_i x_j = x^\top Q x\,, \quad x_i^2 = 1,\ i = 1, \ldots, n\,. \tag{6}$$

Needless to say, the constraints $x_i^2 = 1$ just express that $x_i = \pm 1$. The symmetric matrix Q has nonnegative coefficients, but this is irrelevant for our purpose.

The above problem has the form (1), with $\mathcal{X} := \mathbb{R}^n$ and $c_j(x) := x_j^2 - 1$. Form the Lagrangian

$$L(x,u) = x^\top Q x - \sum_{i=1}^{n} u_i (x_i^2 - 1) = x^\top (Q - D(u)) x + e^\top u\,; \tag{7}$$

here $D(u)$ denotes the diagonal matrix constructed from the vector u, e is the vector of all ones. Minimizing L with respect to x (on the whole of $\mathbb{R}^n$!) is a trivial operation: if $Q - D(u)$ is not positive semidefinite, we obtain $-\infty$; otherwise, the best to do is to take $x = 0$. Now (6) is a minimization problem, for which the dual function provides lower bounds; this dual function must be maximized. In a word, the dual problem associated with (6), (7) is

$$\max e^\top u\,, \quad \text{subject to } Q - D(u) \succcurlyeq 0\,. \tag{8}$$

An interesting point, to be seen in (14) below, is that the optimal value of this SDP (semi-definite programming) problem is just the SDP bound of [20]. This gives an incentive to dualize more general problems with quadratic constraints, such as

$$\begin{cases} \max x^\top Q_0 x + 2{b_0}^\top x + c_0\,, \\ x^\top Q_j x + 2{b_j}^\top x + c_j = 0\,,\ j = 1, \ldots, m\,. \end{cases} \tag{9}$$

The Lagrangian is the quadratic function $L(x,u) = x^\top Q(u) x + 2b(u)^\top x + c(u)$, where $Q(u) := Q_0 - \sum_{j=1}^{m} u_j Q_j$ and likewise for $b(u)$, $c(u)$. Maximizing $L(\cdot, u)$ is slightly more complicated than for maxcut, due to the linear term $2b(u)^\top x$; but the idea is just the same and gives rise again to SDP programming. In fact it can be shown that the dual of (9) is the problem with variables $(u,r) \in \mathbb{R}^m \times \mathbb{R}$

$$\min r\,, \quad \begin{pmatrix} c(u) - r & b(u)^\top \\ b(u) & Q(u) \end{pmatrix} \preccurlyeq 0\,. \tag{10}$$

Such quadratic duality goes far back, at least to [18], continuing with [66], and more recently [38].

This duality tool, available for general quadratically constrained problems, can be appplied to other combinatorial problems fitting in the same framework. One is the maximum stable set problem, which can be formulated as:

$$\max w^\top x\,, \quad x_i x_j = 0,\ (i,j) \in A\,, \quad x_i^2 = x_i,\ i = 1, \dots, n$$

($A \subset \{1, \dots, n\}^2$ being the set of arcs in a graph). Just as for maxcut, the bound obtained by dualizing the quadratic constraints is a known one: Lovasz' ϑ number [46]. For a systematic use of relaxation with quadratic constraints in combinatorial optimization, see [60,38].

2.4 Conic Duality

Remark 3 is just a particular case of the following variant of (1):

$$\max f(x)\,, \quad x \in \mathcal{X}\,, \quad c(x) \in K\,, \tag{11}$$

where K is a given set in $\mathbb{R}^m$. Here we assume that K is a closed convex cone (but see Remark 5 below). To introduce a dual problem, use again slack variables to put (11) in the form (1):

$$\max f(x)\,, \quad (x,s) \in \mathcal{X} \times K\,, \quad c(x) - s = 0 \in \mathbb{R}^m\,.$$

Again with $u \in \mathbb{R}^m$, we form the Lagrangian $L'(x,s,u) = L(x,u) + u^\top s$, where $L(x,u) = f(x) - u^\top c(x)$ as in (2). Then the dual function is

$$\theta'(u) = \max_{(x,s)\in\mathcal{X}\times K} L'(x,s,u) = \theta(u) + \max_{s\in K} u^\top s\,, \tag{12}$$

with θ of (3). The term $\max u^\top s$ is clearly 0 for u in the so-called *polar cone* of K:

$$K^\circ := \{u \in \mathbb{R}^m : u^\top s \leqslant 0 \text{ for all } s \in K\}\,, \tag{13}$$

and $+\infty$ elsewhere: our dual problem is to minimize $\theta(u)$ over $u \in K^\circ$. This is the whole business of conic duality, as developed for example in [47, Chap. 8] or [56, Chap. 6]; see also [25, §XII.5.3(a)]. It contains among other things SDP duality, as developed for example in [1,73].

Remark 5 *One may ask why K should be a closed convex cone. Actually,* (12) *gives the fully general dual function $\theta' = \theta + \sigma_K$, where $\sigma_K(u) := \sup_{s\in K} u^\top s$ is the so-called* support function *of K (a fundamental object of convex analysis). Arbitrary K's are therefore allowed. However, a support function does not distinguish between a set and its closed convex hull, a consequence of which is:*

- *there is no loss of generality in assuming K closed convex;*
- *said otherwise: the dual problem will not change if K is replaced by its closed convex hull.*

Needless to say, just set $K = \{0\}$ *in* (11) *to obtain* (1)*; the set* $\{0\}^\circ$ *of feasible* u*'s is then the whole space (alternatively, observe that* $\sigma_{\{0\}} \equiv 0$*). In the case of Remark 3, the polar of the nonnegative orthant is the nonpositive orthant. In fact, the support function of a cone* K *is easily computed: it is 0 on* K° *and* $+\infty$ *elsewhere – hence the dual constraint* $u \in K^\circ$ *in this case.*

As an illustration of this remark, we leave it as an exercise to dualize the following variant of (1) *– or of* (11) *with* $K = [-\delta, \delta]^m$*, see also* (5)*:*

$$\max f(x)\,, \quad x \in \mathcal{X}\,, \quad \|c(x)\|_\infty \leqslant \delta\,. \qquad \square$$

As a conclusion, we reproduce the rule following Remark 3:
– consider a *maximization* problem
– and dualize a constraint by *subtracting* the linear term $uc(x)$;
– if the constraint was $c(x) \in K$, a cone,
– then the corresponding multiplier u is constrained to vary in K° of (13)
– (or, for a general K, add σ_K to the dual function).

Take (8) as a particular case: the constraint-value $c'(u) := Q - D(u)$ lies in the space of symmetric matrices (we use primes to recall that (8) is already a dual problem; its dual will be a "bidual" of (6)). Symmetric matrices form a finite-dimensional vector space, which can be made Euclidean with the natural scalar product $\langle M, N\rangle := \sum_{i,j} M_{ij}N_{ij}$. Introducing a (bi)dual variable – a symmetric matrix, call it X – we form the (bi)Lagrangian

$$L'(u, X) = e^\top u - \langle X, Q - D(u)\rangle = e^\top u + \sum_{i=1}^n X_{ii}u_i - \langle Q, X\rangle\,,$$

to be maximized over $u \in \mathbb{R}^n$. This gives the (bi)dual function

$$\theta'(X) = \begin{cases} -\langle Q, X\rangle & \text{if } X_{ii} = 1 \text{ for } i = 1, \ldots, n\,, \\ +\infty & \text{otherwise}\,. \end{cases}$$

Now K is here the closed convex cone of positive semidefinite matrices. It can be shown ([27, Cor. 7.5.4] for example) that its polar is the (closed convex) cone of negative semidefinite matrices, which imposes the constaint $X \preccurlyeq 0$. Finally, change X to $-X$ to make the (bi)dual nicer: we obtain

$$\min \langle Q, X\rangle\,, \quad X \succcurlyeq 0\,, \quad X_{ii} = 1,\ i = 1, \ldots, n\,. \tag{14}$$

This is the SDP relaxation of [20]. Note that a positive semidefinite matrix can be put in the form $X = \sum_{k=1}^r x_k x_k^\top$, where r is the rank of X. Observing that $\langle Q, xx^\top\rangle = x^\top Qx$ and that $(xx^\top)_{ii} = x_i^2$, we see that (6) is just (14), but with X constrained to have the form $X = xx^\top$, i.e. to be of rank 1.

2.5 A Large-Scale Problem: Unit-Commitment

Perhaps the most obvious usage of Lagrangian relaxation is when (1) has many variables, which would become independent if the constraints c were not present.

For an illustration consider the so-called unit-commitment problem, which consists in optimizing the production planning of a set I of power plants, over some time horizon T. A possible formulation is as follows (see [42] and the references therein):

– the control variables are x_i^t for $i \in I$ and $t = 1, \dots, T$, they denote the production level of unit i at time t; x_i [resp. x^t] will stand for the vector $(x_i^t)_{t=1}^T$ [resp. $(x_i^t)_{i\in I}$];
– each unit i can operate within a certain feasible set $\mathcal{X}_i$, representing the operating dynamic constraints (for a nuclear plant, $\mathcal{X}_i$ is a finite set of possible plannings: x_i^t may take a few possible values, say $x_i^t \in \{500, 1000, 1500, 2000\}$ MW; when the production has reached a certain level, it must stay there for a certain number of time periods, etc.);
– at each time t, the demand must be satisfied; these are the so-called static constraints, denoted by $c^t(x^t) \leqslant 0$; usually, the functions c^t are additive, say $c^t(x^t) = \sum_{i\in I} c_i^t(x_i^t)$, or even affine;
– finally, the cost to produce x_i by unit i over the whole time horizon is $C_i(x_i)$.

Then the problem is

$$\begin{cases} \min \sum_{i\in I} C_i(x_i)\,, \\ x_i \in \mathcal{X}_i\,, & i \in I\,, \\ c^t(x^t) \leqslant 0\,, & t = 1, 2, \dots, T\,. \end{cases} \tag{15}$$

Insofar as the static constraints are additive, dualizing them produces an additive Lagrangian: with $u = (u^t)_{t=1}^T$,

$$L(x,u) = \sum_{i\in I} C_i(x_i) + \sum_{t=1}^T u^t c^t(x^t) = \sum_{i\in I} C_i(x_i) + \sum_{t=1}^T u^t \sum_{i\in I} c_i^t(x_i^t)\,.$$

After some reordering, we see that L is a sum of "local" Lagrangians, depending only on x_i: minimizing $L(\cdot, u)$ over $\mathcal{X} := \prod \mathcal{X}_i$ gives one problem per plant:

$$\min_{x_i\in\mathcal{X}_i} C_i(x_i) + \sum_{t=1}^T u^t c^t(x_i^t)\,,$$

which is a lot easier than (15) (to fix ideas, the French production set has some 150 plants working each day; optimizing each of them separately is a reasonable task, while optimizing them altogether is unconceivable).

Remark 6 *The above example illustrates perfectly the usual situation in Lagrangian relaxation: the only available information concerning* (1) *– here* (15) *– is the oracle which, given u, maximizes the Lagrangian $L(\cdot, u)$; and this oracle may be fairly complex. Here it is itself composed of some 150 "local oracles", all probably very different (nuclear, gas, hydraulic, ...) and probably very sophisticated, at least for laymen in electrical engineering.* □

2.6 Column Generation in Linear Programming

Consider the following type of problem. We have a very long list of points $\varkappa_1, \ldots, \varkappa_\propto$ in $\mathbb{R}^n$, where $\propto$ is some enormous number, possibly $+\infty$. We want to find a $\varkappa_k$ maximizing a linear function, say $b^\top \varkappa_k$, and satisfying affine constraints, say $A\varkappa_k = a \in \mathbb{R}^m$: we write the problem as

$$\max \{b^\top \varkappa_k : A\varkappa_k = a, k = 1, \ldots, \propto\} . \tag{16}$$

Assumption 7 *Column generation requires the following conditions:*

(i) The number m of constraints is reasonable (as opposed to the cardinality $\propto$ of the list).

(ii) Rather than solving the initial problem, one accepts to relax *it, replacing the set $\{\varkappa_1, \ldots, \varkappa_\propto\}$ by its convex hull.*

(iii) It is "easy" to maximize linear functions over the $\varkappa_k$'s, i.e. to solve the unconstrained version of (16)*: $\max \{c^\top \varkappa_k : k = 1, \ldots, \propto\}$, for given $c \in \mathbb{R}^n$ (possibly different from b).* □

In view of (ii), one is interested by *upper bounds*, as in Lagrangian relaxation. As for (iii), it says that one can compute the support function (see Remark 5) of the $\varkappa_k$'s – or of their convex hull, which is the same function. In other words, the constraints $A\varkappa_k = a$ are *complicating*. Clearly the situation resembles (1).

We refer to [75, Chap. 11] and the references therein for the many applications of column generation. Let us just mention: cutting stock, where each $\varkappa_k$ represents a cut pattern; network optimization, a path from a source to a destination; facility location, a set of clients assigned to a depot; etc.

Remark 8 *Actually, Assumption 7 is just what combinatorial optimization is all about, each $\varkappa_k$ representing one among many possible candidates to solving a certain problem. The most naive instance is the standard 0-1 linear programming problem*

$$\max b^\top x , \quad x \in \{0,1\}^n , \quad Ax = a . \tag{17}$$

After all, it can be formulated with notation of the present section: we want to find a 0-1 vector $\varkappa \in \mathbb{R}^n$ (there are $\propto = 2^n$ of them) maximizing a linear function and satisfying affine constraints. Barring these last constraints, the problem would become trivial.

Admittedly, column generation may not be deemed most appropriate for the present model, though. It is more constructive to distinguish two groups of constraints in (17)*: we write*

$$\max b^\top x , \quad x \in \{0,1\}^n , \; Dx = d , \quad Ax = a ,$$

the $\varkappa_k$'s being now the 0-1 points satisfying $Dx = d$. The approach is still feasible if, among other things, D is structured enough, so that it is easy to maximize a linear function $c^\top x$ on 0-1 vectors satisfying $Dx = d$. Dispatching a given set of constraints among the above A and D belongs to the art of 0-1 programming. □

Let us formulate (16) more compactly as

$$\max b^\top x\,, \quad x \in \mathcal{X}\,, \quad Ax = a\,. \tag{18}$$

This has the advantage of accepting $\infty = +\infty$; but more importantly, the connection with (1) becomes more blatant: the universe $\mathcal{X}$ is made of the $\varkappa_k$'s or, in view of Assumption 7(ii), of their convex hull. The objective is linear and the constraints dualized in (2) are affine. In view of Assumption 7(iii), the Lagrangian problem (3), which is here $\max_{x\in\mathcal{X}} b^\top x - u^\top(Ax-a)$, is "easy": Assumption 7(iii) is nothing other than Assumption 2.

Now column generation is a special technique to solve (18), in which $\mathcal{X}$ is iteratively replaced by smaller sets $\mathcal{X}_K \subset \mathcal{X}$: (18) is replaced by the *master program*

$$\max b^\top x\,, \quad x \in \mathcal{X}_K\,, \quad Ax = a\,. \tag{19}$$

A bounded polyhedron is taken for $\mathcal{X}_K$, so that (19) is an ordinary linear program. Also, the extreme points of $\mathcal{X}_K$ are extracted from the $\varkappa_k$'s in (16); but this is of no importance for the moment anyway: what is interesting is how $\mathcal{X}_K$ is updated for the next iteration.

To obtain $\mathcal{X}_{K+1}$ in column generation, one gets from the resolution of (19) an m-vector u_K (the multipliers associated with $Ax = a$), and one solves the *satellite program*: one finds the most positive among the numbers $(b - A^\top u_K)^\top \varkappa_k$, $k = 1, \ldots, \infty$, which, in view of Assumption 2(iii), is an "easy" problem. Using notation as in (19) and neglecting the constant term $a^\top u_K$, this problem amounts to solving

$$\max_{x\in\mathcal{X}} (b - A^\top u_K)^\top x + a^\top u_K = b^\top x - u_K^\top(Ax - a) = L(x, u_K)\,.$$

We see that the satellite does nothing other than computing the dual function $\theta(u_K)$ as in (3). Let us summarize these observations:

Fact 9 *Insofar as* (1) *has linear data ($\mathcal{X}$ polyhedral, f linear, g affine), column generation and Lagrangian relaxation do the same thing with the same tool, starting from opposite premises:*

- *In Lagrangian relaxation, the constraints $c(x) = 0$ are relaxed and the universe $\mathcal{X}$ is kept as "hard"; the resulting problem is solved in the oracle* (3).
- *Column generation works the other way round: the constraints $c(x) = 0$ are kept as "hard", while the universe is restricted to a smaller set; the resulting problem is solved in the master* (19). □

Thus, any formulation of a problem in terms of column generation can be done in terms of Lagrangian relaxation (and conversely). We believe that this is important because Lagrangian relaxation – duality theory – mostly calls for fairly simple concepts, to be seen in §3. By contrast, column generation has to call for the often fussy language of linear programming. In particular, it cannot be accounted for directly on the compact formulation (19). Remember our point (i) in §1.1.

Anyway Fact 9 has several interesting consequences:

- Column generation can be applied to more general problems than (16). In fact, the resulting methodology is often called *generalized linear programming*, as is lucidly explained in [49].
- Because column generation actually convexifies $\mathcal{X}$ in (18), Lagrangian relaxation should do the same. This will be confirmed in §3.2 below; see more precisely Theorem 17, Fact 18, Example 22.
- In Lagrangian relaxation, the dual function θ of (3) must be minimized. Now we have seen that, in the column generation language, $\theta(u_K) = s + a^\top u_K$, where s is the output from the satellite. The question at stake in column generation *is* therefore to find multipliers u_K such that $s + a^\top u_K$ – i.e. θ – is minimal.
- In other words, column generation *is by necessity* a minimization mechanism. The master (19) *must* provide a possible algorithm to minimize θ; this will be confirmed in §4.2 below.
- Alternatively, any algorithm to minimize θ *must* provide an alternative to the master (19) for column generation; this will be confirmed in §4.3 and 5.2 below; see more precisely Remark 33.

2.7 Entropy Maximization

Even though the following example is definitely out of the combinatorial world, we mention it to emphasize once more the versatility of Lagrangian relaxation. It is indeed an optimization technique that should always be kept in mind.

Entropy maximization appears all the time when an unknown function $x(t)$, $t \in [0,1]$ must be identified with the help of some measurements. Then one seeks the "most probable" $x(\cdot)$ compatible with these measurements. Here is a typical example, where the measurements are some Fourier coefficients of x:

$$\min \int_0^1 x(t) \log x(t) \mathrm{d}t \,, \quad \int_0^1 \cos(2j\pi t) x(t) \mathrm{d}t = z_j \,, \; j = 1, \dots, m \,.$$

This problem has infinitely many variables but finitely many constraints. Changing signs (we have a minimization problem) and forgetting the term $-\sum_j u_j z_j$ (constant in x), the Lagrangian $L(x,u)$ is the integral over $[0,1]$ of the function

$$\ell(x(t),u) := x(t)\log x(t) + \sum_{j=1}^m u_j \cos(2j\pi t) x(t) = x(t)\log x(t) + g_u(t)x(t) \,.$$

Skipping infinite-dimensional technicalities, we can admit the (true) fact that minimizing $L(\cdot,u)$ is achieved by minimizing $\ell(\cdot,u)$ for each t. This last problem is straightforward indeed: differentiate ℓ with respect to its first argument and solve for $x(t)$. We obtain $1 + \log x(t) + g_u(t) = 0$, which has the unique solution

$$e^{-1-g_u(t)} = \exp\bigl(-1 - \textstyle\sum_{j=1}^m u_j \cos(2j\pi t)\bigr) =: x_u(t) \,.$$

Plugging this value (always positive!) in the expression of ℓ gives an explicit formula for the dual function. See [9] for more details and references.

3 Minimal Amount of Convex Analysis

As seen in §1.2, Lagrangian relaxation just replaces a primal problem (1) by its dual (2)-(4). We will first give the main properties of the dual problem. Then we will consider questions posed in the primal when the dual problem is solved: how good is the upper bound? how about recovering a primal optimal solution? Some familiarity with elementary convex analysis is required here. For this, [26] may be useful.

3.1 Minimizing the Dual Function

In this section, we look at (1)-(4) with dual glasses, concentrating on the space $\mathbb{R}^m$ of dual variables, and somehow forgetting the primal space $\mathbb{R}^n$. Here is the fundamental result, whose proof is immediate:

Theorem 10 *Whatever the data $\mathcal{X}, f, c$ in (1) can be, the function θ is always convex and lower semicontinuous.*

Besides, if u is such that (3) has an optimal solution x_u (not necessarily unique), then $g_u := -c(x_u)$ is a subgradient of θ at u:

$$\theta(v) \geqslant \theta(u) + g_u^\top (v-u) \quad \textit{for any } v \in \mathbb{R}^m \ , \tag{20}$$

which is written as $g_u \in \partial\theta(u)$. □

Some comments are useful for a good understanding of this result.

– An important object in convex analysis is the *epigraph* of a function θ, denoted by $\operatorname{epi}\theta$: this is the set of $(u,r) \in \mathbb{R}^m \times \mathbb{R}$ such that $\theta(u) \leqslant r$. It is not difficult to realize that convex functions are exactly those whose epigraph is a convex subset of $\mathbb{R}^m \times \mathbb{R}$.

– Lower semicontinuity means: for any $u \in \mathbb{R}^m$, the smallest cluster value of $\theta(v)$ when $v \to u$ is at least $\theta(u)$: $\liminf_{v\to u} \theta(v) \geqslant \theta(u)$. Alternatively, a function is lower semicontinuous when its epigraph is a closed subset of $\mathbb{R}^m \times \mathbb{R}$.

– Lower semi-continuity is an important property for minimization. It guarantees existence of a minimum point as soon as θ "increases at infinity". By contrast, a non-lower semicontinuous function would be for example

$$[0,+\infty[\ni u \mapsto \theta(u) := \begin{cases} u \text{ if } u > 0 \ , \\ 1 \text{ if } u = 0 \ . \end{cases}$$

Minimizing such a function (over $u \geqslant 0$) is meaningless; said otherwise: this function has no minimum point.

– In our $L \to \theta$ context, it helps intuition to think of $\mathcal{X}$ as a sort of index-set: instead of $\theta(u) = \max_{x\in\mathcal{X}} f(x) - u^\top c(x)$, a writing more suggestive for our purpose would be: $\theta(u) = \max_{k\in\mathcal{K}} f_k - u^\top c_k$ (it goes with the column generation of §2.6, where $\mathcal{K} = \{1,2,\dots,\infty\}$). Thus the Lagrangian defines a family of functions $u \mapsto L$ which are *affine* in u.

– Now observe that taking the supremum of a family of functions corresponds to intersecting their epigraphs.
– Then convexity and lower semicontinuity of θ become easy to accept: in fact, the epigraph of each function $u \mapsto L$ is a closed convex set (a so-called half-space). Their intersection $\operatorname{epi}\theta$ is again closed and convex.
– As for the subgradient property (20), it is straightforward: by definition of θ and looking at (2), we have

$$\theta(v) \geqslant L(x, v) = L(x, u) - c(x)^\top (v - u)$$

for all $x \in \mathcal{X}$, including for $x = x_u$.
– Observe that $-c(u)$ is the vector of partial derivatives of L with respect to u. The subgradient relation gives a rigorous meaning to the following very informal statement: "the (total) derivatives of θ are the partial derivatives of L with respect to u".

Such a statement can be justified in very special situations: suppose that $\mathcal{X}$ is the space $\mathbb{R}^n$, that L is a smooth function of x and u, and that (3) has a unique solution x_u which varies smoothly with u. Then write

$$\frac{\mathrm{d}\theta}{\mathrm{d}u}(u) = \frac{\partial L}{\partial u}(x_u, u) + \frac{\partial L}{\partial x}(x_u, u)\frac{\partial x_u}{\partial u}\,;$$

but because x_u is a maximum point, the partials of L with respect to x vanish. This explains that, to differentiate θ, it suffices to differentiate L with respect to u.

An important consequence of Theorem 10 is that the dual problem is therefore always "easy", insofar as (3) is easy to solve. First, it can be qualified as well-posed, in the sense that minimizing a convex function is well-posed; and this independently of (1), be it NP-hard or anything else. Second, under the sole condition that we are able to maximize the Lagrangian, we obtain "for free" the value $\theta(u)$ of the function, and the value $-c(x_u)$ of a subgradient; both informations are important to solve the dual problem.

As a result, solving a dual problem is exactly equivalent to minimizing a convex function with the help of the oracle (3) providing function- and subgradient-values. Sections 4 and 5 will be devoted to this problem, from a numerical point of view.

3.2 Primal-Dual Relations

A natural question is now: the dual problem is easy, but what is it good for, in terms of the primal? We already know that it provides upper bounds but can we say more? and when can we obtain a primal optimal solution once the dual is solved?

Everett's Theorem. Let us start with a very elementary observation. Suppose we have maximized $L(\cdot, u)$ for a certain u, and thus obtained an optimal x_u,

together with its constraint-value $c(x_u) \in \mathbb{R}^m$; set $g_u := -c(x_u)$ and take an arbitrary $x \in \mathcal{X}$ such that $c(x) = -g_u$. By definition, $L(x,u) \leqslant L(x_u,u)$, which can be written

$$f(x) \leqslant f(x_u) - u^\top c(x_u) + u^\top c(x) = f(x_u)\,. \tag{21}$$

This is a useful result for approximate resolutions of (1):

Theorem 11 (Everett) *With the notation above, x_u solves the following perturbation of* (1)*:*

$$\max f(x)\,, \quad x \in \mathcal{X}\,, \quad c(x) = -g_u\,. \qquad \square$$

The above "a posteriori" perturbed problem consists in replacing the right-hand side of the constraints in (1) by the vector g_u. If this vector is small, x_u can be viewed as approximately optimal; if $g_u = 0$, we are done: x_u is optimal.

More can actually be said in Everett's Theorem.

– First, suppose x_u is only an ε-maximizer of the Lagrangian; in other words: $L(x_u,u) \geqslant L(x,u) - \varepsilon$ for all $x \in \mathcal{X}$. Then (21) becomes $f(x) \leqslant f(x_u) + \varepsilon$, and x_u becomes an ε-optimal solution of the perturbed problem.
– Second observe that, to obtain the desired property $f(x) \leqslant f(x_u)$ in (21), it suffices to have $u^\top(c(x) - c(x_u)) \leqslant 0$. As a result, x_u solves the problem

$$\max f(x)\,, \quad x \in \mathcal{X}\,, \quad u^\top c(x) \leqslant -u^\top g_u\,.$$

– Even more: x_u solves any other problem having more constraints than above, providing these constraints are satisfied by x_u. Thus, consider the particular case of Remark 3: it is clear that x_u solves the perturbed problem

$$\max f(x)\,, \quad x \in \mathcal{X}\,, \quad c_j(x) \leqslant \begin{cases} \max\{0, c_j(x_u)\} & \text{if } u_j = 0\,, \\ c_j(x_u) & \text{if } u_j > 0\,, \end{cases}$$

in which optimality conditions appear explicitly: if x_u is such that $c_j(x_u) \leqslant 0$ (x_u feasible) with equality whenever $u_j > 0$ (complementarity slackness), then x_u solves (1).

Dual Subdifferentials. The above considerations go back to [12]. Here we focus our attention to a differential study of the dual function.

In view of Theorem 10, the partial gradient $\nabla_u L(x_u,u) = -c(x_u)$ of L with respect to u lies in $\partial\theta(u)$. A converse to this property turns out to be crucial for primal-dual relations. Introduce the notation

$$\begin{aligned} \mathcal{X}(u) &:= \{x \in \mathcal{X} : L(x,u) = \theta(u)\}\,, \\ G(u) &:= \{g = -c(x) : x \in \mathcal{X}(u)\} \end{aligned} \tag{22}$$

for the set of optimal solutions in (3) and its image through $\nabla_u L$. Because a subdifferential is always a closed convex set, we deduce that the closed convex hull of $G(u)$ is entirely contained in $\partial\theta(u)$. An important question is whether $\partial\theta(u)$ is thus entirely described; besides, getting rid of the (trouble-making) closure operation is desirable. This explains the following concept:

Definition 12 (Filling Property) *The filling property for* (1)-(3) *is said to hold at* $u \in \mathbb{R}^m$ *if* $\partial\theta(u)$ *is the convex hull of the set* $G(u)$ *defined in* (22). □

A first question is then: when does the filling property hold? This is actually fundamental in subdifferential calculus: when is it possible to characterize the subdifferential of a sup-function? The question is extremely technical and will be skipped in this paper, as it has little to do with our development. To get a positive answer requires additional assumptions, of a topological nature. Let us enumerate some favourable situations:

Theorem 13 *The filling property of Definition 12 holds at any* $u \in \mathbb{R}^m$
- *when* $\mathcal{X}$ *is a compact set on which* f *and each* c_j *are continuous;*
- *in particular, when* $\mathcal{X}$ *is a finite set (usual Lagrangian relaxation of combinatorial problems);*
- *in linear programming of §2.1 and in quadratic programming of §2.2;*
- *more generally, in problems where* f *and the* c_j *are quadratic (§2.3), or even* ℓ_p*-norms,* $1 \leqslant p \leqslant +\infty$*; see [71,37].* □

The interested reader may consult [68] for the most recent results along these lines. When the filling property holds at u, then any anti-subgradient of θ at u is a convex combination of constraint-values at sufficiently many x's in $\mathcal{X}(u)$; say

$$g \in \partial\theta(u) \quad \Longrightarrow \quad g = -\sum_k \alpha_k c(x_k)\,.$$

Now a u minimizing θ is characterized by the property $0 \in \partial\theta(u)$, which means: there exists sufficiently many x's in $\mathcal{X}(u)$ such that the $c(x)$'s have 0 in their convex hull:

Proposition 14 *Let* u^* *solve the dual problem* (4) *and suppose the filling property holds at* u^**. Then there are (at most* $m+1$*) points* x_k *and convex multipliers* α_k *such that*

$$L(x_k, u^*) = \theta(u^*) \text{ for each } k, \text{ and } \sum_k \alpha_k c(x_k) = 0\,. \qquad \square$$

With these x_k's and α_k's, it is tempting to make up the primal point

$$x^* := \sum_k \alpha_k x_k\,, \tag{23}$$

which definitely deserves attention[2]. In fact, the following reasoning reveals conditions under which this x^* does solve (1):

[2] Of course, the writing (23) implies that $\mathcal{X}$ enjoys some structure, namely that its elements can be *averaged*. Besides, observe the connotation of column generation: as a convex combination of primal points computed by the satellite, x^* could very well solve (19) with an appropriate $\mathcal{X}_K$.

– If $\mathcal{X}$ is a convex set (say in $\mathbb{R}^n$) then $x^* \in \mathcal{X}$. Appropriate use of Everett's Theorem 11 tells us to what extent x^* is approximately optimal in (1).
– If, in addition, c is an affine mapping (from $\mathbb{R}^n$ to $\mathbb{R}^m$) then

$$c(x^*) = \sum_k \alpha_k c(x_k) = 0\,.$$

Thus x^* is feasible in (1). Besides $L(x^*,u^*) = f(x^*) - (u^*)^\top c(x^*) = f(x^*)$, so the number $\theta(u^*) - f(x^*)$ (nonnegative because of weak duality) again tells us to what extent x^* is aproximately optimal. This number, the so-called *duality gap*, will be studied in more detail below.
– If, in addition, $L(\cdot,u^*)$ is concave (as a function defined on $\mathbb{R}^n$) then

$$f(x^*) = L(x^*,u^*) \geqslant \sum_k \alpha_k L(x_k,u^*) = \theta(u^*)\,.$$

In view of weak duality, x^* is optimal in (1).

Remark 15 *The above reasoning can be reproduced in the case of conic duality of §2.4, knowing that the dual optimality condition is no longer* $0 \in \partial\theta(u^*)$ *but rather: there is* $g \in \partial\theta(u^*)$ *lying in the normal cone*[3] *to* K° *at* u^**. Alternatively, one can use the "slackened Lagrangian"* $L'(x,y,u) = L(x,u) + u^\top y$*, to be maximized over* $\mathcal{X} \times K$ *and just apply the same reasoning. We leave it as an exercise to do the calculations in the inequality-constrained case of Remark 3.* □

It is important to realize that we have here a *constructive process* to make primal optimal solutions:

Fact 16 *Let a dual algorithm produce* u^* *solving the dual problem. Suppose that*
– *the filling property of Definition 12 holds at* u^**,*
– *appropriate convexity properties hold for* $\mathcal{X}$*,* c *and* $L(\cdot,u^*)$ *so that the above reasoning is valid.*

Under the mere condition that this algorithm is able to declare optimality of u^**, it provides through* (23) *a point* x^* *which solves the primal problem* (1). □

Let us emphasize this fact: unless the dual algorithm minimizing θ is a naive heuristic, without even a proof of convergence, the information necessary to compute x^* – i.e. the x_k's and α_k's in (23) – is certainly available somewhere in this algorithm, namely in its stopping criterion: to detect (approximate) optimality of u^*, the algorithm must know (approximate) maximizers x_k of $L(\cdot,u^*)$ and convex multipliers α_k such that $g_{u^*} := -\sum_k \alpha_k c(x_k)$ is (approximately) zero. More will be seen on this question in Section 5.2.

[3] This is the set of g such that $g^\top(v-u^*) \leqslant 0$ for all $v \in K^\circ$; or equivalently, the set of $g \in K$ such that $g^\top u = 0$; see [25, Ex. III.5.2.6(a)] for example.

Duality Gaps. We have already encountered the *duality gap*: this is the difference of the optimal values in (4) and in (1). It is always a nonnegative number because of weak duality. In view of Proposition 14 and the reasoning following it (and remembering Fact 9), we guess that the dual problem amounts to solving a certain *convex relaxation* of (1). The duality gap would then be the corresponding increase of the optimal value, and this is the object of the present section; see also [10, §16], [14,42].

The most general and best known result is as follows. Introduce the *perturbation function* associated with (1), which is the optimal value as a function of the righthand side of the constraints:

$$\mathbb{R}^m \ni g \mapsto v(g) := \max\{f(x) : x \in \mathcal{X},\, c(x) = g\}\,. \tag{24}$$

Lagrangian relaxation amounts to replacing v by its *concave upper semicontinuous hull*: this is the smallest function v^{**} which is concave, upper semicontinuous (u.s.c.) and larger than v: $v^{**}(g) \geqslant v(g)$ for all $g \in \mathbb{R}^m$. Alternatively, v^{**} is the function whose hypograph (everything that is under the graph) is the closed convex hull of the hypograph of v, as a subset of $\mathbb{R}^m \times \mathbb{R}$. The role of v and v^{**} comes from the following result:

Theorem 17 *The dual optimal value is the value at 0 of the concave u.s.c. hull of the perturbation function:* $\min\theta = v^{**}(0)$. □

Similarly to Proposition 14, this result is completely general, as it assumes no structure whatsoever on $\mathcal{X}$ (while Fact 16, for example, implies that the word "convex combination" has a meaning in $\mathcal{X}$, remember note 2, p.127). In a way, it is minimal: it only requires the image-space $\mathbb{R}^m$ of the constraints to be a Euclidean spacee and neglects completely the. On the other hand, it is rather abstract: visualizing the behaviour of the perturbation function is not easy, in terms of the data $\mathcal{X}, f, c$. This is why some more specific properties are worth investigating when some structure is present in the data.

When $\mathcal{X} \subset \mathbb{R}^n$ (so that taking convex combinations in $\mathcal{X}$ makes sense) it becomes possible to analyze the duality gap in a more primal language.

Assume first that the Lagrangian is affine in x. Then its maximum values are not changed if $\mathcal{X}$ is replaced by its closed convex hull[4]. In view of weak duality, this implies the following:

Fact 18 *For an instance of* (1) *with linear data:*

$$\max b^\top x\,, \quad x \in \mathcal{X} \subset \mathbb{R}^n\,, \quad Ax = a\,, \tag{25}$$

denote by $\bar{\mathcal{X}}$ *the closed convex hull of* $\mathcal{X}$*. The dual minimal value* $\min\theta$ *is not smaller than the maximal value in* (25) *with* $\mathcal{X}$ *replaced by* $\bar{\mathcal{X}}$*.* □

[4] The dual function is then essentially the support function of $\mathcal{X}$, already seen in Remark 5; more precisely, in the notation of (25), $\theta(u) = a^\top u + \sigma_{\mathcal{X}}(b - A^\top u)$.

Beware that "not smaller than" does not mean "equal to": this result does *not* mean that the duality gap amounts to replacing $\mathcal{X}$ by $\bar{\mathcal{X}}$. In terms of Theorem 17, convexifying $\mathcal{X}$ amounts to taking the concave hull of the perturbation function v; but the resulting function need not be upper semicontinuous at 0. Here lies a technical difficulty, which will be illustrated by the counter-example 22 below. Nevertheless, cases where this difficulty does occur can be considered as pathological. The duality gap exactly makes this convexification in most cases, for example those described in the following result:

Theorem 19 *Equality holds in Fact 18 in any of the following cases:*

(i) $\bar{\mathcal{X}}$ *is a bounded set in* $\mathbb{R}^n$*;*

(ii) for any $g \in \mathbb{R}^m$ *close enough to 0, there is* $x \in \bar{\mathcal{X}}$ *such that* $Ax = a + g$*;*

(iii) there exists u^* *minimizing the dual function, and the filling property 12 holds at* u^**.* □

– Property (i) above is of course that of "ordinary" Lagrangian relaxation in combinatorial optimization; the corresponding result goes back to [13], rediscovered on various occasions, see for example [19,49]. At this point, it is worth mentioning that the closed convex hull of a bounded set is bounded ([25, Thm. IV.1.4.3] for example): when $\mathcal{X}$ is a finite set, $\bar{\mathcal{X}}$ is just the convex hull of $\mathcal{X}$.

– Property (ii) is universally invoked, under the name of *Slater condition*. Less known is the fact that it is necessary and sufficient for the dual function to have a nonempty compact set of minimum points.

– As for Property (iii), the result is easy to establish from Proposition 14 and the discussion following it.

Remark 20 *Remember column generation of §2.6: it also replaces a set* $\mathcal{X}$ *by its closed convex hull. When* $\propto < +\infty$*, Theorem 19 applies: both column generation and Lagrangian relaxation find the common optimal value of* (4) *or* (18)*.*

When $\propto = +\infty$ *– or more generally if* $\mathcal{X}$ *in* (18) *is an arbitrary (unbounded) closed convex set – there may be a pathological duality gap. In that case, Lagrangian relaxation does not solve* (18)*, but rather its dual* (4)*. So does column generation as well, this will result from §4.2 below.* □

The previous development applies when the Lagrangian $L(\cdot, u)$ is affine. Such is no longer the case when (1) has non-affine data f and/or c; the situation is then substantially more complicated, we outline the kind of results that can be obtained. Similarly to §2.4 and Remark 4, use a slack-technique and introduce the set

$$\mathcal{G} := \{(x, s_0, s) : x \in \mathcal{X},\ s_0 \leqslant f(x),\ s = c(x)\} \subset \mathbb{R}^n \times \mathbb{R} \times \mathbb{R}^m\,, \qquad (26)$$

which combines the graph of c and the epigraph of f. Put the s_0-axis as vertical. Solving (1) amounts to intersecting $\mathcal{G}$ with the subspace $s = 0$, and then finding the highest point in the resulting set. Now write the dual function as

$$\theta(u) = \max_{(x,s_0,s)\in\mathcal{G}} s_0 - u^\top s$$

to see that it is not changed if $\mathcal{G}$ is replaced by its closed convex hull[5]. This shows how Fact 18 and Theorem 19 can be generalized to nonlinear data: Lagrangian relaxation amounts to close-convexifying the above $\mathcal{G}$. However, the closed convex hull of $\mathcal{G}$ – and its intersection with $s = 0$ – is a rather complicated object; so we stop our development at this point and refer the interested reader to [42].

Exact Relaxation. Finally, a relevant question is when the duality gap is 0. According to Theorem 17, this occurs exactly when the perturbation function v coincides with its concave u.s.c. hull at 0. Such is the case when v itself is concave and upper semicontinuous at 0.

- Concavity holds whenever "everything is convex" in (1) (cf. the discussion following Theorem 14): $\mathcal{X}$ is a convex set (say in $\mathbb{R}^n$), f is a concave function on $\mathcal{X}$, and c is affine: $c(x) = Ax - a$. As already observed in Remark 20, there is little hope to have a 0 duality gap if $\mathcal{X}$, or more generally $\mathcal{G}$ of (26), is not convex.
- Upper semicontinuity is a "normal" property, in the sense that convex problems with a duality gap can be considered as pathological. Two sorts of situations are known, where such a pathology cannot occur:
  - (i) One is similar to the situation where the filling property 12 holds, namely when the primal problem enjoys some compactness – see Theorem 13.
  - (ii) The other is based on the following important result: a convex function is continuous at a point whenever it is defined ($< +\infty$) in a neighborhood of this point. Particularizing this result to our context, assume that the perturbation function v of (24) is concave. It is continuous (hence u.s.c.) at 0
    - if it is finite ($> -\infty$) in a neighborhood of 0, or equivalently,
    - if the feasible domain remains feasible despite small perturbations of the righthand sides.

    This explains the Slater condition popping up in Theorem 19. Interestingly enough, this situation is somehow dual to (i), in that it corresponds to compactness in the dual space.

  We collect these situations in one exactness result:

Theorem 21 *Let* (1) *be a convex optimization problem: $\mathcal{X}$ is a closed convex set in $\mathbb{R}^n$, $f : \mathcal{X} \to \mathbb{R}$ is a concave function, the constraint-functions c_j are affine. Assume that the dual function* (3) *is not identically $+\infty$. Then there is no duality gap* – (1) *and* (4) *have the same optimal value* – *if one at least of the following properties holds:*

(i) $\mathcal{X}$ is a bounded set in $\mathbb{R}^n$, f [resp. each inequality constraint] is upper [resp. lower] semicontinuous on $\mathcal{X}$;

(ii) for any $g \in \mathbb{R}^m$ close enough to 0, there is $x \in \mathcal{X}$ such that $c(x) = g$;

(iii) there exists u^* minimizing the dual function, and the filling property 12 holds at u^*;

[5] Comparing with note 4, p. 129, we have $\theta(u) = \sigma_{\mathcal{G}}(0, 1, -u)$.

(iv) $\mathcal{X}$ *is a polyhedral set,* f *and all constraints are affine functions (linear programming).* □

Adapting this result to (non-affine) inequality contraints (Remark 3) and conic programming (§2.4) is an interesting exercise. We conclude this section with a counter-example, essentially due to R.T. Rockafellar, to show that pathological duality gaps do exist.

Example 22 *Consider the following instance of* (1)*, set in terms of* $(x, y, z) \in \mathbb{R}^3$*;*

$$\begin{cases} \max x\,, \quad \mathcal{X} := \left\{(x,y,z) : \sqrt{x^2+y^2} \leqslant z\right\}, \\ c_1(x,y,z) := x - 1 \leqslant 0\,, \\ c_2(x,y,z) := z - y \leqslant 0\,, \\ c_3(x,y,z) := 1 - z \leqslant 0\,. \end{cases}$$

It is easy to see that the constraints imply $z = y$ *and* $x = 0$*. The optimal value is 0, attained on all of the feasible set, namely* $\{0\} \times]-\infty, 1] \times [1, +\infty[$*.*

Now keep $\mathcal{X}$ *as the universe, introduce three (nonnegative) dual variables* u, v, w *and dualize* c_1*,* c_2*,* c_3*, forming the Lagrangian*

$$\begin{aligned} L &= x - u(x-1) - v(z-y) - w(1-z) \\ &= (1-u)x + vy + (w-v)z + u - w\,. \end{aligned}$$

Maximizing it with respect to z *implies* $w - v \leqslant 0$ *– otherwise we get* $+\infty$ *–, in which case the optimal* z *is* $\sqrt{x^2+y^2}$*. The dual function is therefore obtained by maximizing*

$$L' = (1-u)x + vy - (v-w)\sqrt{x^2+y^2}$$

with respect to x, y*. A key is to realize that the maximum of* L' *is either 0 or* $+\infty$*, depending whether* $\sqrt{(1-u)^2+v^2} \leqslant v - w$*; in fact the maximal value of* L' *and its argmax are given as follows:*

$\sqrt{(1-u)^2+v^2}$	$\max_{x,y} L'$	*maximal* (x,y)
$< v - w$	0	$(0,0)$
$= v - w$	0	$\mathbb{R}_+(1-u, v)$
$> v - w$	$+\infty$	–

To see it[6]*, fix first the norm of* $(x, y) \in \mathbb{R}^2$*; the maximum is attained for* (x, y) *collinear to* $(1-u, v)$*. Then adjust suitably the norm of* (x, y)*.*

Because $w \geqslant 0$*, the condition* $\sqrt{(1-u)^2+v^2} \leqslant v - w$ *implies* $u - 1$ *and* $w = 0$*; then this condition is satisfied for arbitrary* $v \geqslant 0$*. The corresponding value of* L *(the dual function) is* $u - w = 1$*. In summary, the dual optimal value is 1 (*> 0*), attained on all of the dual feasible set, namely* $\{1\} \times \mathbb{R} \times \{0\}$*.* □

[6] An expert in convex analysis will recognize in $\max L'$ the indicator function of the ball of radius $(v - w)$ at the point $(1-u, v) \in \mathbb{R}^2$ (up to the multiplicative term $v - w$).

Among other things, this example illustrates the distinction between Fact 18 and Theorem 19: the objective is linear, the constraints are affine, both primal and dual problems have optimal solutions. Nevertheless there is a duality gap; what is missing is the filling property 12. We mention that the counter-example relies on a crucial property: its primal and dual optimal sets are unbounded, as well as the argmax of the Lagrangian at a dual optimal solution.

4 Dual Algorithms I: Subgradient, Cutting Plane, ACCPM

We now come back to the dual space and study numerical methods solving the dual problem. Mainly for simplicity, we assume that there are no dual constraints, a solution exists in (3) for all $u \in \mathbb{R}^m$. Even though it is not essential for dual algorithms, this assumption eases substantially the exposition.

Thus we are in the following situation:

Assumption 23 *We are given a convex function θ, to be minimized.*
- *It is defined on the whole of $\mathbb{R}^m$.*
- *The only information available on θ is an oracle, which returns $\theta(u)$ and one subgradient $g(u)$, for any given $u \in \mathbb{R}^m$.*
- *No control is possible on $g(u)$ when $\partial\theta(u)$ is not the singleton $\nabla\theta(u)$.* □

The third assumption above just means that it is impossible to select a particular maximizer of the Lagrangian, in case there are several. As far as dual algorithms are concerned, the fact that $g(u)$ has the form $-c(x_u)$ is irrelevant: x_u is an unknown object to the dual algorithm, which deals only with vectors in $\mathbb{R}^m$. The present situation is rather classical in the world of nonlinear progamming, where one has to minimize some objective function, given function- and gradient-values only, say $\theta(u)$ and $g(u)$ respectively. The difference here is that $g(u)$ does not vary continuously with u: we are dealing with *nonsmooth optimization*.

Remark 24 *In some situations, a much richer oracle is available; most notably in the relaxation of quadratically constrained problems of §2.3. The corresponding dual problems are traditionally solved via interior-point methods, coming from* SDP *optimization ([1,73]). However, problems such as* (8)*, or more generally* (10)*, can be fruitfully considered as instances of nonsmooth convex problems, for which a rather complete subdifferential analysis can be performed. For such problems, algorithms of the type below can be used, as well as more specialized ones, taking advantage of the richer information available. This gives birth to useful alternatives to interior-point methods, particularly well-suited for large* SDP *problems; see [24,39,59].* □

There are essentially two types of nonsmooth optimization methods: subgradient (§4.1) and cutting-plane[7] (§4.2), somehow lying at the opposite sides of a continuum of methods; in between, are found analytic center and bundle (§5).

[7] This terminology is unfortunate in a combinatorial community: here the planes in question cut a portion of the space $\mathbb{R}^{m+1}$ where $\operatorname{epi}\theta$ is lying.

In what follows, we will call g_k the $g(u_k)$ computed by the oracle of Assumption 23, when called at some iterate u_k.

4.1 Subgradients and Ellipsoid Methods

Let u_K be the current iterate. Consider the subgradient relation (20) for u strictly better than the current iterate u_K: we have

$$\theta(u_K) > \theta(u) \geqslant \theta(u_K) + g_K^\top (u - u_K)\,,$$

which shows that $g_K^\top(u - u_K) > 0$ for any such u, in particular for $u = u^*$, a point minimizing θ. Then it is easy to see that, for $t > 0$ small enough, the point $u(t) := u_K - t g_K$ is strictly closer than u_K to u, hence to any optimal u^*.

This is the starting idea for the subgradient method, going back to [72], formalized in [11,61], and often called "Lagrangian relaxation" in the combinatorial community[8]. It works as follows:

$$\text{At iteration } K\text{, select } t_K > 0 \text{ and set } u_{K+1} = u_K - t_K g_K\ .$$

It is hard to imagine anything simpler. In fact, the only responsibility of the method is to choose the stepsize t_K – since no control on g_K can be exerted from the dual space. Besides, the choice of t_K is also the simplest thing in the world, at least in theory:

Theorem 25 *Let the subgradient method be applied with the stepsizes given as follows:*

$$t_K = \frac{\lambda_K}{\|g_K\|}\ , \quad \textit{with}\ \ \lambda_K \downarrow 0\ \textit{and}\ \sum_{K=1}^{\infty} \lambda_K = +\infty\,.$$

Then the sequence of best function-values $\theta_K^* := \min\{\theta(u_k) : k = 1, \ldots, K\}$ *tends to the minimum value of* θ *over* $\mathbb{R}^m$. □

However, this extreme simplicity has its price in terms of speed of convergence: the method can only give a coarse approximation of the optimal value. Anyway observe that the above result is of little practical value: who can wait long enough to check that λ_K satisfies the required properties? This is why a big issue when implementing this method is a correct adjustment of t_K; a hard job because the real trouble lies in g_K, a bad direction[9].

Considering that better convergence implies better directions, an idea is to adapt the metric. Make a change of variable in $\mathbb{R}^m$, say $u = Bv$ (B being an invertible matrix). Then minimizing $\theta'(v) := \theta(Bv)$ is obviously equivalent to

[8] This terminology is unfortunate in *any* community: it mixes the general *methodology* of Lagrangian relaxation and the particular *technique* of subgradient optimization: the latter method is by no means the only possibility to implement the mechanism (3), (4) – or more generally, to minimize a convex function.

[9] In nonlinear programming, $-g_K$ corresponds to the gradient method, or steepest descent, only used by amateurs.

minimizing θ. However, suppose that θ is differentiable; then the gradient of θ' is $\nabla\theta'(v) = B^\top \nabla\theta(Bv)$. It turns out that the same formula holds for subgradients. As a result, the subgradient iteration to minimize θ' is $v_{K+1} = v_K - t_K B^\top g_K$. Multiplying by B to return in the u-space, we obtain $u_{K+1} = u_K - t_K BB^\top g_K$; the direction has changed, as requested.

N.Z. Shor thus devised a class of *variable metric* subgradient methods, with a metric $B = B_K$ varying at each iteration. To construct B_K, he used the elementary operator *dilating* the space along a given vector $\xi = \xi_K$, suitably chosen. He gave two choices for ξ:

The Ellipsoid Algorithm. His first idea was motivated by the following observation ([65, §3.1]): assume for simplicity that θ has a unique minimum point u^*; the (sub)gradient direction is so bad that it may be considered as orthogonal to the direction pointing to u^*. It is therefore advisable to favour the subspace orthogonal to $B_K g_K$. This led him to take $\xi_K = B_K g_K$; the resulting algorithm was given in [64]. Later on, A.S. Nemirovski defined specific values for the coefficient of dilatation and for the stepsize t_K [54], and this was the celebrated ellipsoid algorithm.

The *r* Algorithm. Shor was never too convinced by the behaviour of the above algorithm, and he defined another choice for ξ. Consider a point where $g(\cdot)$ is discontinuous: for two values of u rather close together, say u_K and u_{K-1}, the oracle returns two very different subgradient-values g_K and g_{K-1}. Then there are good reasons to think that the desired direction (pointing toward u^*) is orthogonal to $g_K - g_{K-1}$ (draw a picture in two dimensions, with g_K and g_{K-1} quasi-opposite). It is therefore advisable to dilate the space along the bad direction $\xi_K := g_K - g_{K-1}$. This gave birth to the r-algorithm [67].

Remark 26 *Another derivation of the subgradient method is worth mentioning. Take a nonsingular $m \times m$ matrix M and form the quadratic function $q(h) := \theta(u_K) + g^\top h + \frac{1}{2} h^\top M h$. Straightforward calculations show that it has a minimum point at $h = -M^{-1}g$. Thus, a subgradient algorithm can be derived as follows: consider $q(h)$ with $M = B^{-\top} B^{-1}$ as a "model" approximating $\theta(u_K + h)$; then minimize q; then move along the direction thus obtained with a stepsize t_K. This idea of an approximating model such as q is important in optimization; it will be most instrumental in the methods to come.* □

We conclude this section with a comment about primal recovery. Indeed the subgradient algorithm does provide the primal point x^* of Fact 16. This little-known property belonged to folklore ([65, p. 116], [2]), until it was formally given in [35]. In fact, suppose each primal point x_k, computed by the oracle at u_k, are stored along the iterations. Then compute

$$\hat{x}_K := \frac{\sum_{k=1}^K t_k x_k}{\sum_{k=1}^K t_k}\,.$$

Under reasonable assumptions, this averaged point converges to x^* of (23).

4.2 The Cutting-Plane Method, or of Kelley, Cheney-Goldstein

The previous methods were simplest, here is now a fairly sophisticated one, going back to [7,28]. Every call to the oracle, input with u_k say, defines an affine function approximating θ from below: $\theta(u) \geqslant \theta(u_k) + g_k^\top (u - u_k)$ for all $u \in \mathbb{R}^m$ (with equality for $u = u_k$). After K such calls, we have on hand a piecewise affine function $\hat{\theta}$ which underestimates θ: for all $u \in \mathbb{R}^m$,

$$\theta(u) \geqslant \hat{\theta}(u) := \max \{\theta(u_k) + g_k^\top (u - u_k) : k = 1, \dots, K\}, \tag{27}$$

which we will call the *cutting-plane model* of θ. It is entirely characterized by the K-uple of elements $(\theta(u_k), g_k) \in \mathbb{R}^{m+1}$; this is what we will call the *bundle* of information.

Admitting that the model approximates correctly the true θ, minimizing it makes sense; and this is a linear programming problem. Thus, the method commonly called "cutting planes" in nonlinear programming, or also Kelley-Cheney-Goldstein, consists in computing an optimal solution (u_{K+1}, r_{K+1}) of

$$\begin{cases} \min r\,, \quad (u, r) \in \mathbb{R}^{m+1}\,, \\ r \geqslant \theta(u_k) + g_k^\top (u - u_k) \quad \text{for } k = 1, \dots, K\,. \end{cases} \tag{28}$$

The next iterate is u_{K+1}, with its model-value $\hat{\theta}(u_{K+1}) = r_{K+1}$. A call to the oracle input with u_{K+1} gives a new piece $(\theta(u_{K+1}), g_{K+1})$ enriching the bundle (raising the cutting-plane model $\hat{\theta}$) and the process is repeated.

In view of (27), we have by construction $r_{K+1} = \hat{\theta}(u_{K+1}) \leqslant \hat{\theta}(u) \leqslant \theta(u)$ for all $u \in \mathbb{R}^m$. Then the number

$$\delta := \theta(u_K) - r_{K+1} = \hat{\theta}(u_K) - \hat{\theta}(u_{K+1}) \geqslant 0 \tag{29}$$

is of particular interest.

- It represents a *nominal decrease*, which would be dropped from the true θ if the model were accurate enough[10].
- It serves to bracket the dual optimal value: because $r_{K+1} = \theta(u_K) - \delta \leqslant \theta(u)$ for all u, we have

$$\theta(u_K) \geqslant \min \theta \geqslant \theta(u_K) - \delta = r_{K+1}\,.$$

- Hence it can serve as stopping test: the algorithm can be stopped as soon as δ is small[11], and this will eventually occur if (and only if) 0 is a cluster point of the sequence of δ-values.
- In view of the bracket, this last event exactly means

$$\liminf \theta(u_K) = \limsup r_{K+1} = \min \theta\,.$$

[10] It is even a maximal decrease, and even an upper bound on the total possible decrease $\theta(u_K) - \min \theta$.

[11] A tradition is rather to stop when $\theta(u_{K+1}) - r_{K+1}$ is small. We introduce δ because it will be a central object in bundle methods, §5 below. Anyway the most reasonable test is based on the best function-value θ_K^* of Theorem 25 and should be preferred.

– The above sequence $(\theta(u_K))$ can be – and usually is – cahotic. By contrast, the sequence (r_K) is obviously nondecreasing (since the function $\hat{\theta}$ increases at each iteration) and bounded from above (by any $\theta(u)$); it therefore has a limit.

Thus the key issue for convergence is whether $r_K \uparrow \min\theta$. Such is the case indeed, but with a technical restriction which turns out to be important: a mechanism is needed to guarantee *boundedness* of the sequence (u_K).

The Case of Column Generation. To close this section, we proceed to show that the present cutting-plane algorithm is *nothing other* than column generation of §2.6. In fact form the dual of (28): with (bi)dual variables $\alpha_k \geqslant 0$, the (bi)Lagrangian is

$$L'(u,r,\alpha) = r - r\sum_{k=1}^{K}\alpha_k + \sum_{k=1}^{K}\alpha_k\big(\theta(u_k) - g_k^\top u_k\big) + u^\top\sum_{k=1}^{K}\alpha_k g_k\,.$$

Minimizing $L'(u,\cdot,\alpha)$ forces $\sum_k \alpha_k = 1$: the α_k's form a set of convex multipliers. Minimizing $L'(\cdot,r,\alpha)$ forces $\sum_k \alpha_k g_k = 0$. The (bi)dual problem is therefore

$$\max\sum_{k=1}^{K}\alpha_k\big(\theta(u_k) - g_k^\top u_k\big)\,,\quad \alpha_k \geqslant 0\,,\ \sum_{k=1}^{K}\alpha_k = 1\,,\ \sum_{k=1}^{K}\alpha_k g_k = 0\,.$$

Now remember that we are actually solving a dual problem: returning to the primal space of (1), we can write $\theta(u_k) = L(x_k,u_k)$ and $g_k = -c(x_k)$, so that[12] $\theta(u_k) - u_k^\top g_k = f(x_k)$. Thus, the above (bi)dual is

$$\max_{\alpha\in\Delta_K}\sum_{k=1}^{K}\alpha_k f(x_k)\,,\quad \sum_{k=1}^{K}\alpha_k c(x_k) = 0\,, \tag{30}$$

where Δ_K denotes the unit-simplex in $\mathbb{R}^K$.

This problem is definitely posed in the primal space: it involves (convex combinations of) objective- and constraint-values. When the objective function $f(x) = b^\top x$ is linear and the constraints $c(x) = Ax - a$ are affine, we recognize *exactly* the master problem (19) $\mathcal{X}_K$ being the convex hull of $\{x_1,\ldots,x_K\}$. In summary:

Fact 27 *Knowing that* (16) *is just an instance of* (1) *with polyhedral/linear data* $\mathcal{X}, f, c$*, solving it by the column-generation master* (19) *amounts to solving the dual* (4) *by the cutting-plane algorithm* (28)*.* □

This supplements Remark 20 (in particular, it makes no difference to have $\mathcal{X}$ or its convex hull as universe, remember Fact 18). As a consequence, we see that

[12] Remember that $g_k \in \partial\theta(u_k)$. An expert in convex analysis will detect the conjugate function: $\theta^*(g_k) = \theta(u_k) - g_k^\top u_k$, which "explains" Everett's Theorem 11.

– the present cutting-plane algorithm,
– the subgradient algorithm of §4.1,
– the variants of the latter (ellipsoid, r-algorithm),
– ACCPM, which will be seen below in §4.3,
– the bundle algorithms of §5,
all of these are various alternatives to solve the same problem: (1) or (16), or at least their convex relaxation.

Reverting the above formulation, we find generalized linear programming:

Fact 28 *The cutting-plane algorithm* (28)*, as well as all of the above-mentioned alternatives, generalize the column-generation mechanism of* §*2.6 to nonlinear/-nonpolyhedral data in* (16) *– knowing that* (28) *is just the so-called generalized linear programming algorithm.* □

A final word, with relation to Fact 16, going along the end of §4.1. Again suppose the primal points x_k, computed by the oracle at u_k, are stored along with the dual elements $\theta(u_k)$ and g_k. Then an optimal solution $\hat{\alpha}$ of the (bi)dual (30) allows the computation of the primal point $\hat{x} := \sum_k \hat{\alpha}_k x_k$. This point, which actually solves (19), approximates a primal optimal solution as mentioned in §3.2: see (23) again.

4.3 Stability Problems: ACCPM

From a practical point of view, observe that (28) is obviously feasible (take r large enough) but has no reason to be bounded from below: $\hat{\theta}$ has no reason to have a finite minimum. Think of $K = 1$: we get $r_2 = -\infty$ and u_2 at infinity, and even *no matter how close* u_1 is to a minimum of θ. The cutting-plane algorithm is inherently unstable. The mechanism to bound u_K, required by theory, has also a practical role: to stabilize the algorithm.

Although apparently naive, the above argument *is* a serious one and serves as a basis for an impressive counter-example found in [54] (and reproduced in [25, § XV.1.1]). A function is defined, not particularly pathological – essentially $\theta(u) = \max\{0, -1 + \|u\|\}$ –, and for which obtaining the approximate optimality condition $\delta \leqslant \varepsilon^2$, requires $K(\varepsilon) \simeq (1/\varepsilon)^m$ cutting-plane iterations.

By comparison, the ellipsoid algorithm performs as follows: to obtain the same accuracy ε^2 in the worst case (no matter how nasty the oracle θ can be), an order of $m^2 \log(1/\varepsilon) \simeq m \log K(\varepsilon)$ iterations are needed at most[13].

This bad behaviour of the cutting-plane algorithm is due to the fact that the model $\hat{\theta}$ is much too optimistic; something else is needed.

Remark 29 *In nonlinear programming, it is commonly admitted that an optimization algorithm can hardly be good if it is not based on a model approximating*

[13] For bundle methods to come, the worst-case number of iterations is of the order $1/\varepsilon^6$ (see [74,33]). This is definitely worse than ellipsoid, but does not depend on the dimension m of the space.

the objective function to second order. Here $\hat{\theta}$ can be considered as a "global" approximation of θ (in the sense that it provides information all over the space $\mathbb{R}^m$); but it is a good local approximation nowhere. The best we can hope is to have the 1st-order approximation property $\hat{\theta}(u_k + h) = \theta(u_k + h) + o(\|h\|)$ near each reference point u_k.

Unfortunately, if it is conceivable to construct convenient first-order approximations of convex functions, second-order is at present fictional (do not forget that the θ we are minimizing does not even have continuous first derivatives); for efforts along these lines see for example [44,45,40] and the references therein. This is true only for oracle-functions, though; when more information is known from the objective functions, more is possible; see [15, Chap. 14], the more recent [59,52], and Remark 24. □

A stabilizing idea in the spirit of interior-point methods works as follows. Let θ^*_K be the best function-value of Theorem 25 and form the set

$$\begin{aligned} P_K &:= \{(u,r) : \hat{\theta}(u) \leqslant r \leqslant \theta^*_K\} \\ &= \{(u,r) : \theta(u_k) + g_k^\top (u - u_k) \leqslant r \leqslant \theta^*_K \ \text{ for } k = 1, \dots, K\}\,. \end{aligned}$$

This is a polyhedron in the graph-space $\mathbb{R}^{m+1}$, which we assume bounded (possibly via the mechanism used to bound the cutting-plane iterates).

The cutting-plane algorithm consists in taking for (u_{K+1}, r_{K+1}) a point that is lowest in P_K; according to Nemirovski's counter-example, this is a bad point. A better idea is to take a point that is *central* in P_K. A good concept for this, due to G. Sonnevend [69], is the *analytic center* of P_K: the point maximizing the product of slacks (or sum of their logs) of the constraints defining P_K. Based on this idea, the method ACCPM (analytic center cutting-plane method) has been defined in [21]; see [22] for a recent overview.

To conclude this section, let us mention that the same stability problem exists for subgradient methods of §4.1: g_k jumps along the iterations, reflecting the nondifferentiability[14] of θ. It can even be said that instability is an inherent difficulty with Lagrangian relaxation, and more generally with nonsmooth optimization. Indeed remember Proposition 14: let only to prove optimality of a given u^*, the dual algorithm has to get from the oracle a number (possibly as large as $m+1$) of distinguished primal points x_k, in order to make up the point x^* of Fact 16.

5 Dual Algorithms II: Bundle Methods

Just as ACCPM, bundle methods start from cutting planes and aim at overcoming the instability problem mentioned in §4.3. However, while ACCPM preserves the global character of the cutting-plane model $\hat{\theta}$ (in the sense that P_K does not favour any of the successive interates $u_1, \dots, u_K$), they borrow from nonlinear programming the concept of local approximation – see Remark 29.

[14] This phenomenon is best explained on the column generation model of §2.6: when the $c = c_u$ of Assumption 7(iii) varies, the $\varkappa$ returned by the oracle can only jump from one $\varkappa$ to another.

5.1 The Algorithmic Scheme

The rationale of bundle methods is as follows:
- A *stability center* is chosen; denote it by $\hat{u}$: it is a point one would like u_{K+1} to be close to, its choice will be made precise below.
- As said already, the cutting-plane model $\hat{\theta}$ is much too optimistic: say $\hat{\theta} \ll \theta$ in a neighborhood of $\hat{u}$.
- As a result, minimizing $\hat{\theta}$ gives a u_{K+1} much too far from $\hat{u}$ (a disaster revealed by Nemirovski's counter-example).
- It is therefore advisable to add a positive term to $\hat{\theta}$, so as to improve its local approximation character;
- we choose a quadratic stabilizing term, of the type $\|u-\hat{u}\|^2$,
- whose effect is to pull u_{K+1} toward $\hat{u}$, thereby stabilizing the algorithm.

Using a *spring strength* $t>0$ (which may depend on the iteration number) aimed at tuning the stabilization effect, the method therefore approximates the true function $\theta(u)$ by the "stabilized" cutting-plane model

$$\breve{\theta}(u) := \hat{\theta}(u) + \frac{1}{2t}\|u-\hat{u}\|^2\,, \tag{31}$$

which is minimized at each iteration. Instead of (28), a linearly constrained quadratic problem is solved:

$$\begin{cases} \min r + \dfrac{1}{2t}\|u-\hat{u}\|^2\,, \quad (u,r)\in\mathbb{R}^{m+1}\,, \\ r \geqslant \theta(u_k) + g_k^\top(u-u_k) \quad \text{for } k=1,\dots,K\ . \end{cases} \tag{32}$$

This problem has always a unique optimal solution (u_{K+1}, r_{K+1}). Besides, the two positive numbers

$$\begin{aligned} \delta &:= \theta(\hat{u}) - \hat{\theta}(u_{K+1})\,, \\ \breve{\delta} &:= \theta(\hat{u}) - \breve{\theta}(u_{K+1}) = \delta - \frac{1}{2t}\|u_{K+1}-\hat{u}\|^2 \end{aligned} \tag{33}$$

still represent a prediction of the expected decrease $\theta(\hat{u})-\theta(u_{K+1})$, probably more reasonable than the δ of (29) produced by (28) (remember note 10, p. 136).

To complete the iteration, we must take care of the stability center. For this we test the actual decrease, as compared to δ; say

$$\theta(u_{K+1}) \leqslant \theta(u_K) - \kappa\delta, \tag{34}$$

κ being a fixed tolerance ($\breve{\delta}$ could be preferred to δ but this is a detail: one shows that $\delta\in[\breve{\delta}, 2\breve{\delta}]$). If (34) holds, we set $\hat{u}=u_{K+1}$; in the bundle jargon, this is called a *descent-step*, as opposed to a *null-step*: if (34) does not hold, $\hat{u}$ is left as it is (but $\hat{\theta}$ is enriched). For convergence, κ must be positive. Besides, one easily sees that the descent property (34) equivalently means that the model is accurate enough:

$$\theta(u_{K+1}) \leqslant \hat{\theta}(u_{K+1}) + (1-\kappa)\delta\,.$$

Thus, κ must also be smaller than 1, otherwise $\hat{u}$ will never be updated.

Let us now consider the stopping test. Because u_{K+1} minimizes $\check{\theta}$ instead of $\hat{\theta}$, and because $\check{\theta}$ need not be smaller than the actual objective function, neither δ nor $\check{\delta}$ of (29) provide a safe bracket of the optimal value $\min\theta$. Unless one really trusts the approximation $\check{\theta}$ of (31), stopping simply when δ or $\check{\delta}$ is small may be unduly optimistic. Something safer is as follows: u_{K+1} minimizes the model (31), which is a convex function, hence $0 \in \partial\check{\theta}(u_{K+1}) = \partial\hat{\theta}(u_{K+1}) + \frac{1}{t}(u_{K+1} - \hat{u})$. This can be written

$$u_{K+1} - \hat{u} = -t\hat{g} \quad \text{for some } \hat{g} \in \partial\hat{\theta}(u_{K+1}) \;, \tag{35}$$

which reveals an important object: $\hat{g} = (\hat{u} - u_{K+1})/t$, called the *regularized gradient* in the bundle jargon. As shown in Theorem 32 below, it can be computed from the dual of (32).

Thus we can write for all $u \in \mathbb{R}^m$:

$$\begin{aligned} \theta(u) \geqslant \hat{\theta}(u) &\geqslant \hat{\theta}(u_{K+1}) + \hat{g}^\top(u - u_{K+1}) \\ &= \theta(\hat{u}) - \delta + \hat{g}^\top(u - u_{K+1}) \,, \end{aligned} \tag{36}$$

so that we can stop when *both* δ and $\hat{g}$ are small[15]. We summarize various convergence properties of bundle methods, collected from [74,29,8], [25, Chaps. IX, XIII, XV]; see also [36].

Theorem 30 *We assume for simplicity that the stepsize t remains fixed along the iterations; say $t \equiv 1$.*

(i) There holds at each iteration

$$\begin{aligned} \theta(u) &\geqslant \theta(\hat{u}) - \delta + \hat{g}^\top(u - \hat{u}) \\ &\geqslant \theta(\hat{u}) - \delta - \|\hat{g}\|\,\|u - \hat{u}\| \quad \textit{for all } u \in \mathbb{R}^m \,. \end{aligned}$$

(ii) Any of the following equivalent events

$$\hat{g} = 0 \quad \textit{or} \quad \delta = 0 \quad \textit{or} \quad \check{\delta} = 0 \quad \textit{or} \quad u_{K+1} = \hat{u}$$

guarantees that $\hat{u}$ minimizes θ over $\mathbb{R}^m$.

(iii) The sequences $(\hat{g})$, (δ), $(\check{\delta})$ and $(\hat{u} - u_{K+1})$ tend to 0.

(iv) The sequence of stability centers is minimizing: $\theta(\hat{u}) \downarrow \min\theta$.

(v) If θ has a nonempty set of minimizing points, then the entire sequence of stability centers $(\hat{u})$ converges to such a minimizing point.

(vi) The events mentioned in (ii) do occur at some finite iteration K when the set of possible answers from the oracle is finite. □

Case (vi) above concerns a finite universe $\mathcal{X}$ in (1) and means that θ is polyhedral; this is for example the case in column generation.

Even though this result considers a fixed stepsize, it should be noted that numerical efficiency depends crucially on a varying t, suitably updated at each iteration. This is for sure a weak point of the approach, although some update strategies do exist: [32,43].

[15] In anticipation of §5.2 below, note that the cutting-plane algorithm of §4.2 produces (36) with $\hat{g} = 0$.

Remark 31 *Having thus derived bundle methods from the cutting-plane method of §4.2, we can also compare* (35) *with the subgradient iteration of §4.1:*
- *Here the next iterate is moved from the stability center $\hat{u}$, instead of the current iterate u_K.*
- *Alternatively, a subgradient method can be seen as a sort of bundle method whithout any check for descent such as* (34)*: one systematically sets $\hat{u} = u_K$.*
- *The move is made in a direction resembling a subgradient. Actually, it can be shown that $\hat{g} \in \partial\theta(\hat{u})$ if t is small enough.*
- *We also mention that the $\hat{g}$ of* (35) *is vaguely related to the direction used in the r-algorithm (not necessarilty with t small). Specifying this link more clearly needs investigation, though.* □

Thus, a subgradient method can in a way be seen as a particular bundle method, obtained for small t. Conversely, it is rather clear in (32) that a large t results in a weak stabilization: we come close to the pure cutting-plane method. Between these two extremes, bundle methods form a whole continuum parameterized by t.

5.2 Primal-Dual Relations: Augmented Lagrangian

Forming the dual of (32) is a good exercise illustrating §2.2, and is quite parallel to the derivation of (30). With nonnegative (bi)dual variables $\alpha \in \mathbb{R}^K$, the (bi)Lagrangian is

$$L'(u, r, \alpha) = r + \frac{1}{2t}\|u - \hat{u}\|^2 + \sum_{k=1}^{K} \alpha_k[-r + \theta(u_k) + g_k^\top(u - u_k)]\,.$$

Minimizing $L'(u, \cdot, \alpha)$ forces $\sum_k \alpha_k = 1$: the α_k's form a set of convex multipliers. Minimizing $L'(\cdot, r, \alpha)$ produces the unique $u = u_\alpha = \hat{u} - tg(\alpha)$, where we have set $g(\alpha) := \sum_k \alpha_k g_k \in \mathbb{R}^m$. Plugging this value into L' gives the (bi)dual function

$$-\frac{t}{2}\|g(\alpha)\|^2 + \sum_{k=1}^{K} \alpha_k\big(\theta(u_k) - g_k^\top u_k\big) + g(\alpha)^\top \hat{u}\,.$$

Now remember that we are actually solving a dual problem: returning to the primal space of (1), we can write $\theta(u_k) = L(x_k, u_k)$ and $g_k = -c(x_k)$, so that $\theta(u_k) - g_k^\top u_k = f(x_k)$. Recalling the notation $\Delta_K \subset \mathbb{R}^K$ for the unit-simplex, the (bi)dual problem can be written

$$\max_{\alpha\in\Delta_K} \sum_{k=1}^{K} \alpha_k f(x_k) - \frac{t}{2}\|g(\alpha)\|^2 + \hat{u}^\top g(\alpha)\,, \quad \sum_{k=1}^{K} \alpha_k c(x_k) = -g(\alpha)\,. \tag{37}$$

Duality between the pair of linear-quadratic problems (32), (37) gives (invoke for example Theorem 21, since (37) has a compact feasible set):

Theorem 32 *Let* $\hat{\alpha}$) *solve* (37). *The optimal solution* $(u_{K+1}, r_{K+1} = \hat{\theta}(u_{K+1}))$ *of* (32) *is then obtained from* (35) *with* $\hat{g} := g(\hat{\alpha})$.

Besides, we have

$$\hat{f} := \sum_{k=1}^{K} \alpha_k f(x_k) = \theta(\hat{u}) - \delta + t\|\hat{g}\|^2 - \hat{u}^\top \hat{g}\,. \qquad \square$$

The interesting point is to compare (37) with (30): while the cutting-plane method imposes the constraint $\sum_k \alpha_k c(x_k) = 0$ at each iteration, a bundle method *relaxes* this constraint via two terms in the objective function:

– one is a *penalization* with the penalty coefficient $t > 0$;

– the other is a *dualization* with the value $\hat{u}$ for the dual variable; incidentally, the term $\sum_k \alpha_k f(x_k) + \hat{u}^\top g(\alpha)$ in (37) can also be written $\sum_k \alpha_k L(x_k, \hat{u})$.

This is the so-called *augmented Lagrangian* technique, well-known in nonlinear programming, and much studied in [3] for example. It combines Lagrangian relaxation with an alternative way of eliminating constraints: penalization.

Remark 33 *As already said on several occasions, a bundle algorithm can be used for column generation. After convergence of the algorithm, the primal point* $\hat{x} := \sum_k \alpha_k x_k$ *of* (37) *can still be constructed. In contrast with the end of §4.2, however, this point is not feasible during the course of the algorithm. Theorem 30 guarantees via* (36) *that this point becomes feasible* asymptotically*: $\hat{g}$ (standing for $g(\hat{x})$ or $A\hat{x} - a$) tends to 0. Besides, $\hat{f}$ (standing for $f(\hat{x})$ or $b^\top \hat{x}$) tends to* $\min \theta$ *if $\hat{g}^\top \hat{u}$ tends to 0 – for example if $\hat{u}$ is* bounded*; in this case, $\hat{x}$ becomes optimal for the relaxed primal problem.* $\square$

Augmented Lagrangian is an important theory, let us consider it from a higher point of view. To eliminate constraints in a general optimization problem such as (1), an alternative to dualizing them is to penalize them: one maximizes without constraints $f(x) - \frac{t}{2}\|c(x)\|^2$ over the whole of $\mathcal{X}$; quite a simple idea. It is not difficult to establish the asymptotic equivalence or this penalized problem with (1), when $t \to +\infty$. Augmented Lagrangian does something more subtle: it applies Lagrangian relaxation to

$$\max f(x) - \frac{t}{2}\|c(x)\|^2\,, \quad x \in \mathcal{X}\,, \quad c(x) = 0\,. \tag{38}$$

This seems of no avail: using simultaneously relaxation and penalization seems simply redundant. However, the situation is substantially different in the dual. In fact duality applied to (38) gives the *augmented Lagrangian*

$$L_t(x, u) := f(x) - \frac{t}{2}\|c(x)\|^2 - u^\top c(x)$$

and the corresponding augmented dual function

$$\theta_t(u) := \max_{x \in \mathcal{X}} L_t(x, u)\,.$$

It can be shown that

(i) if $u = u^*$ minimizes θ of (3), *and if there is no duality gap*, maximizing $L_t(\cdot, u^*)$ can only produce primal points which are feasible, hence optimal: any "parasitic" infeasible primal point is eliminated;

(ii) if $t \to +\infty$, maximizing $L_t(\cdot, u)$ produces again primal optimal solutions, even for bad u.

Concerning (ii), the influence of t can be drastic: compare the next result with Theorem 17.

Fact 34 *Under mild and natural assumptions (always satisfied, for example, if $\mathcal{X}$ is a finite set), augmented Lagrangian cancels out the duality gap: for t large enough,* $\min \theta_t = v(0)$ *of* (24). □

The price to pay is that Assumptions 2 and 7(iii) are usually no longer valid when L is replaced by L_t; as a result, augmented Lagrangian can hardly be used *in practice*. Nevertheless, (i) suggests the ability of augmented Lagrangian to stabilize primal infeasibilities, so that the idea can be translated to the numerical field – and this is what bundle methods do.

For an illustration, consider (16). Using Assumption 7(ii), formulate its convex relaxation in terms of convex multipliers α:

$$\max_{\alpha \in \Delta_\infty} \sum_{k=1}^{\infty} \alpha_k b^\top \varkappa_k \,, \quad \sum_{k=1}^{\infty} \alpha_k A \varkappa_k = a \in \mathbb{R}^m \,. \tag{39}$$

Standard column generation defines approximations $\hat{x}$, with ∞ replaced by a smaller number K; a very natural idea, just requiring an appropriate mechanism to generate $\varkappa_{K+1}$, namely the satellite. The bundle approach takes a multiplier vector $u \in \mathbb{R}^m$, a penalty coefficient $t > 0$, forms the augmented Lagrangian

$$L_t(\alpha, u) = \sum_{k=1}^{\infty} \alpha_k f(x_k) - \frac{t}{2} \Big\| \sum_{k=1}^{\infty} \alpha_k c(x_k) \Big\|^2 - u^\top \sum_{k=1}^{\infty} \alpha_k c(x_k) \,, \tag{40}$$

and makes the following reasoning:

– Standard duality/penalty theory says that minimizing $L_t(\cdot, u)$ is equivalent to (39) in two cases:
  – if u is a dual solution (and $t > 0$ is arbitrary); this results from Theorem 21(i), for example (the unit-simplex Δ_∞ is compact);
  – if $t \to +\infty$ (and u is arbitrary in $\mathbb{R}^m$); this natural result can be proved rather easily.
– Unfortunately, both problems have comparable complexity, so nothing is really gained so far.
– A column-generation-like technique is therefore applied to the minimization of $L_t(\cdot, u)$ (even though it is unconstrained – but one can always set $g := -\sum_k \alpha_k c(x_k)$, considered as a constraint): ∞ is replaced by K to obtain estimates $\hat{x} = \sum_{k=1}^{K} \alpha_k \varkappa_k$; again a very natural idea if the augmented Lagrangian idea is accepted as natural.

– However, the optimal u is not known; hence, simultaneously to the process generating $\hat{x}$, one generates dual estimates $\hat{u}$. In addition to the satellite, a mechanism is required to generate u_{K+1} – this is (32).

Inequality Constraints. As seen in Remark 3, primal inequality constraints entail nonnegative dual variables: (32) is changed to

$$\begin{cases} \min r + \dfrac{1}{2t}\|u - \hat{u}\|^2 \,, \quad (u, r) \in \mathbb{R}^{m+1} \,, \\ r \geqslant \theta(u_k) + g_k^\top (u - u_k) \quad \text{for } k = 1, \ldots, K \,, \\ u \geqslant 0 \end{cases}$$

(knowing that (28) is recovered if $t \to +\infty$). As for the primal space, the reader may check that dualizing the above quadratic problem – with the additional term $s^\top u$ in the (bi)Lagrangian – results in transforming (37) to maximizing over $\alpha \in \Delta_K$ and $s \geqslant 0$

$$\sum_{k=1}^{K} \alpha_k f(x_k) - \frac{t}{2}\|g(\alpha) - s\|^2 + \hat{u}^\top (g(\alpha) - s) \,,$$

where $g(\alpha) := -\sum_{k=1}^{K} \alpha_k c(x_k)$, as before: this formulation reveals the term $\sum_{k=1}^{K} \alpha_k c(x_k) + s$, coming from the constraint $c(x) + s = 0$, $s \geqslant 0$.

This again goes with the general theory: using the slackened formulation of an inequality-constrained primal problem, the augmented Lagrangian is

$$L'_t(x, s, u) = f(x) - \frac{t}{2}\|c(x) + s\|^2 - u^\top (c(x) + s) \,,$$

to be maximized with respect to $x \in \mathcal{X}$ and $s \geqslant 0 \in \mathbb{R}^m$. Incidentally, maximization with respect to s can be carried out explicitly; for given x, we obtain the m-vector $s = s(x)$ whose jth component is $\max\{0, -c_j(x) - u_j/t\}$. With this notation, the augmented dual function is then

$$\theta'_t(u) = \max_{x \in \mathcal{X}} L(x, u) - u^\top s(x) - \frac{t}{2}\|c(x) + s(x)\|^2 \,.$$

5.3 Quadratic Solvers and Poorman Bundle Methods

The main argument against bundle methods is that they replace an "easy" linear program (28) or (30) by a "difficult" quadratic program (32) or (37). This explains various proposals, in which the stabilizer is polyhedral instead of quadratic; the model $\check{\theta}$ of (31) is replaced by, say

$$\check{\theta}(u) := \hat{\theta}(u) + \frac{1}{t}\|u - \hat{u}\|_\infty \,.$$

A similar stabilization was proposed in the seminal *boxstep* method of [50]: the idea was to minimize the model

$$\overset{\sqcup}{\theta}(u) := \begin{cases} \hat{\theta}(u) & \text{if } \|u - \hat{u}\|_\infty \leqslant \rho \,, \\ +\infty & \text{otherwise} \,, \end{cases}$$

i.e. to minimize $\hat{\theta}$ over the *trust-region* $\|u-\hat{u}\|_\infty \leqslant \rho$. The purpose of this section is to study the issue.

Why a Quadratic Master? First there are good reasons to use a quadratic term in the model.

(i) One is by analogy with smooth optimization. As mentioned in Remark 29, quadratic terms in the model, incorporating second-order information on the objective function, are crucial for efficiency.

Indeed it is shown in [25, § II.2.2(b)] that even a very naive quadratic term such as in (31) (i.e. an extremely coarse second-order approximation), may improve drastically the performances: the number of iterations may be divided by a factor of 10^3.

(ii) Similarly, for the so-called *composite optimization* problem (which basically means to minimize the maximum of finitely many smooth functions), a good model is the sum of a polyhedral and of a quadratic function, even quite naive as above; see [15, Chap. 14], but also [62, Chap. 3].

By contrast, a polyhedral stabilization – such as in $\overset{\sqcup}{\theta}$ of boxstep – is strongly biased: it favors the extreme points of the corresponding polyhedral unit-ball, irrespective of the original problem (think of the basis vectors, which are the extreme points of the ℓ_1 unit-ball; their cosine with the "ideal" direction – pointing to an optimal solution – is most probably of the order $1/n$).

(iii) In our present context, motivated in §4.3, a quadratic term has several advantages.

- It guarantees compactness: the model $\breve{\theta}$ of (31) has always a minimum point, for any $t > 0$; such is not the case with the above $\check{\theta}$, for example.
- It preserves the possible first-order approximation property of the cutting-plane model: $\breve{\theta}(\hat{u}+h) = \hat{\theta}(\hat{u}+h) + o(\|h\|)$; here again, $\check{\theta}$ does not.
- It consistently stabilizes u_{K+1}, no matter how $t > 0$ is chosen, and no matter how $\hat{u}$ is close to a minimum point of θ. By contrast, $\breve{\theta}$ may be minimal at $\hat{u}$, an unfortunate event; as for $\overset{\sqcup}{\theta}$, its asymptotic efficiency necessitates to have $\rho \to 0$ at a suitable speed (if $\hat{\theta}$ reaches its minimum in the interior of the box of radius ρ centered at $\hat{u}$, the boxstep precaution could just be forgotten).

(iv) The case of ACCPM is interesting. First, it does not cure by itself the compactness problem: a center is not defined for an unbounded set P_K in §4.3. Besides, a naive quadratic term (with $\hat{u}$ fixed to 0) is again beneficial, at least theoretically: see [55].

A prospective argument could be added to the above list: in some future, research in convex analysis might very well produce appropriate quadratic models, less primitive than the mere Euclidean norm in $\breve{\theta}$. Then a quadratic master will become a must: the difference between pure cutting planes of §4.2 and such "second-order like" methods will be as large as, say, the difference between Gauss-Seidel and Newton for systems of equations.

Efficiency of Quadratic Solvers. The belief that a quadratic master cannot be solved as efficiently as a linear one is indeed misleading.

Observe first that linear and quadratic programming call essentially for the same techniques:

– either pivotal ([4], [15, Chap. 10]), the only real difference being that the dimension of the optimal basis is not known in advance;
– or using interior points [57,76].

Actual implementations solve the dual (37), which has a fairly particular form: a fixed simplicial constraint structure, and a Hessian which is simply the Gram matrix $\left[g_k^\top g_{k'}\right]$. Besides, its complexity is just proportional to K, the number of pieces in the bundle: the dimension m of the dual space does not count. Observe for example that, with $K = 1$, we obtain from (35) the explicit solution $u_2 = \hat{u} - tg_1$ (in this case, $\partial\hat{\theta}$ is everywhere reduced to the singleton $\{g_1\}$; alternatively, (37) has the single feasible point $\alpha_1 = 1$).

Another important point is that the present context lends itself to restarts, just as in column generation. From the Kth master (37) to the $(K+1)$st, the differences are:

– one more variable α_{K+1}; setting it to 0 and appending it to the old optimal solution $\hat{\alpha}$ produces a feasible point in the new master (37);
– one more row-column in the Hessian matrix, obtained by computing the $K+1$ scalar products $g_k^\top g_{K+1}$, $k = 1, \ldots, K+1$;
– one more linear term $f(x_{K+1}) - \hat{u}^\top g_{K+1}$;
– possibly a change in the old linear terms $f(x_k) - \hat{u}^\top c(x_k)$, in case the stability center $\hat{u}$ has been updated (descent step);
– a few other minor changes in case the bundle has been cleared – see below.

Among available software exploiting fully these features, let us cite [30,31,16].

Indeed, it is safe to say that the future of nonlinear optimization now requires a substantial development of quadratic programming software, analogous to the tremendous impetus of the last 50 years for linear programming. Such software, incidentally, is also required for "ordinary" smooth optimization problems, which are nowadays solved by the so-called *sequential quadratic programming* approach (SQP); see [15, §12.4], [58, Chap. 18].

Clearing the Bundle: Cheap Quadratic Masters. At least in theory, the quadratic master can be made straigtforward, as the bundle size can be kept as small as 2. In fact, it is always possible to eliminate from the bundle any number of pieces, provided that one keeps a so-called *aggregate* piece, which is the affine function

$$\theta^{\mathrm{a}}(u) := \hat{\theta}(u_{K+1}) + \hat{g}^\top (u - u_{K+1})\,. \tag{41}$$

Using (35), we see that

$$\theta^{\mathrm{a}} \leqslant \theta \quad \text{and} \quad \theta^{\mathrm{a}}(u_{K+1}) = \theta(u_{K+1})\,.$$

The aggregate piece can thus be viewed as another linearization of θ, which can be incorporated into the bundle without harm.

Remark 35 *The above aggregation has a nice primal interpretation. First observe that each piece in the bundle can be expressed in terms of the Lagrangian:*

$$\theta(u_k) + g_k^\top (u - u_k) = L(x_k, u)\,.$$

Next, using complementary slackness in (32) *and remembering the primal points* x_k*, we see that* $\hat\theta(u_{K+1}) = r_{K+1}$ *is equal to*

$$\sum_{k=1}^K \hat\alpha_k \big(\theta(u_k) - g_k u_k + \hat g^\top u_{K+1}\big) \;= \sum_{k=1}^K \hat\alpha_k L(x_k, u_{K+1}) + g_k^\top u_{K+1}\,,$$

so that the aggregate piece

$$\theta^{\mathrm{a}}(u) = \sum_{k=1}^K \hat\alpha_k L(x_k, u) = \sum_{k=1}^K \hat\alpha_k f(x_k) - u \sum_{k=1}^K \hat\alpha_k c(x_k)$$

is a convex combination of Lagrangians. Besides, this writing connotes the primal candidate $\hat x = \sum_k \hat\alpha_k x_k$*, already encountered on several occasions.* □

The aggregate piece summarizes the information contained in the bundle in such an efficient way that it suffices by itself to guarantee convergence of the dual process:

Theorem 36 *The convergence properties (i)–(v) of Theorem 30 remain valid if, prior to any iteration* $K+1$*, the following operations are performed:*
– *any existing piece* $(\theta(u_k), g_k)$ *(possibly all) is cancelled from the bundle;*
– *the aggregate piece* $(\hat\theta(u_{K+1}), \hat g)$ *is inserted;*
– *the new piece* $(\theta(u_{K+1}), g_{K+1})$ *is appended.* □

As a result, the complexity of the master can be controlled quasi at will: any number of pieces can be eliminated, at any time[16]. Naturally, if any eliminated piece k has $\hat\alpha_k = 0$, there is no need to insert the aggregate piece: it is implicitly present. On the other hand, the simplest possible master at step $K+1$ is

$$\min r + \frac{1}{2t}\|u - \hat u\|^2\,, \quad r \geqslant \theta^{\mathrm{a}}(u)\,,\; r \geqslant \theta(u_{K+1}) + g_{K+1}^\top (u - u_{K+1})\,. \tag{42}$$

Theorem 36 is explained as follows. The only existing scheme to prove convergence of a bundle method is to argue that the optimal value in (32), with K replaced by $K+1$, is certainly larger than that of (42), which has less constraints.

[16] Incidentally, this is by no means shared by the cutting-plane methods of §4.2: although some works exist to eliminate inactive pieces ([70]), a bundle of dimension at least $m+1$ must of course be maintained (otherwise $\hat\theta$ cannot be bounded from below).

Then it is rather easy to analyze the *poorman bundle method*, in which the bundle is fully cleared at each iteration and the master (42) is solved. Convergence results for this method are *a fortiori* true for any "richer" strategy.

In other words: no theoretical result establishes a difference between maximal and minimal bundles. This is rather frustrating, as a rich bundle method can reasonably be expected to converge faster than a poor one. This belief can be assessed only by computational experience, though, which is the subject of the next section.

5.4 Numerical Illustrations

For illustration, we applied bundle method to the Held-Karp dual of a traveling salesman problem [23], with datasets from tsplib; all the graphs were complete, with a dimension m of the dual space equal to the number of nodes. The dual algorithm was that of [43], with the quadratic solver of [30]. The oracle to compute the 1-tree for given u was very elementary, inspecting systematically the $O(m^2)$ possibilities, and storing no distance matrix: the m^2 (Euclidean) distances were recomputed everytime they were needed. Table 1 gives the results obtained when the algorithm was pushed to optimality[17]. The runs were performed with a maximum bundle size of 1000.

Table 1. Bundle method pushed to full optimality

Problem	m	$-\theta(u_K)$	K	K_{d}	ℓ	CPU	% master	% oracle
gr120	120	1606.3125	132	40	42	1s	35	44
pcb442	442	50499.499	556	109	116	151s	64	28
pcb1173	1173	56350.993	502	65	81	438s	31	64
pcb3038	3038	136587.50	4212	324	367	10h	30	56
fnl4461	4461	181569.21	6965	470	361	30h?	8?	87?

In this table, K is the number of iterations needed (each iteration is one resolution of the master (37) + one maximization of the Lagrangian); K_{d} is the number of descent steps, updating the stability center $\hat{u}$; ℓ is the number of active constraints in the final master (32). The last three columns concern computing times. Some observations are worth mentioning.

– We also tested fl1577 but failed: after some 80 000 iterations (about a week of computer time), we still had $\theta(u_k) \simeq -21\,833$, a value not close to $\min \theta$.
– The last three columns were obtained by the unix command `gprof` and are indicative only. In particular, `gprof` gave for fnl4461 an implausible computing time of 4200s; hence our question marks in the corresponding entries of Table 1.

[17] The stopping test is explained in [41]. Given a threshold ε set by the user, two corresponding tolerances are computed for δ of (29) and $\|\hat{g}\|$ of (35), implying that θ is minimized within relative accuracy ε. All the runs reported in Table 1 had $\varepsilon = 10^{-6}$ and terminated with $\delta \leqslant 10^{-3}$ and $\|\hat{g}\| \leqslant 10^{-5}$.

– The CPU time spent in Lagrangian maximizations (last column) becomes heavier when m is larger; by comparison, the time spent in the master problem (37) is light during the first iterations, when $k \ll 1\,000$.
– The values of ℓ are interesting. With a pure cutting-plane method, ℓ could only equal $m+1$. Now ℓ has several interpretations:
  – It is the dimensionality of $\partial\theta(\hat{u})$; if θ were differentiable, we would have $\ell = 1$, the oracle answering some $g_k = 0$.
  – From a primal point of view, ℓ is the number of x_k's making up the optimal solution $\hat{x}$ of the relaxed primal problem.
  – Because $\ell \ll m$, we have to conclude that the set of dual optimal solutions has fairly large dimensionality[18].
  – Alternatively, the optimal set of the primal relaxed problem has fairly small dimensionality.

  Note, however, that all this is highly informal, and subject to some approximation anyway; Table 4 shows that it could be true only in a perfect world with everything computed exactly – including the solution of (32).

Using the same test-problems, we report the same information with the poorman bundle method of §5.3. More precisely, the method is still that of [43], with a bundle systematically reduced to three elements: the aggregate and the new ones, as required by Theorem 36, and the g returned by the oracle at the current $\hat{u}$ (this element is useful for the t-update of [43]). Table 2 reads as Table 1, knowing that the algorithm was interrupted "manually" when 4 exact digits were obtained in the objective function.

Table 2. Poorman bundle method

Problem	$-\theta(u_K)$	K	K_{d}	CPU	% master	% oracle
gr120	1606.1746	173	26	1s	1	89
pcb442	50495.121	233	29	19s	2	94
pcb1173	56345.576	236	26	155s	1	94
pcb3038	136573.85	6920	482	5h45	0	94
fnl4461	181551.12	411	41	45min	0	95

For a more complete comparison, Table 3 reports the number of iterations respectively needed by the two versions (poor and rich), to obtain 2, 3, 4 exact digits (i.e. relative accuracies from 10^{-2} to 10^{-4}; $u_1 = 0$ is already 10^{-1}-optimal in these problems). The last double column summarizes the computing time to reach 10^{-3} relative accuracy. In each double column, the cheap version is reported first.

Our final experiments illustrate how the algorithm proves optimality of a given $\hat{u}$. Still the same test-problems are used, and the bundle method is run

[18] We have $\dim \partial\theta(\hat{u}) = \ell - 1$. Because θ is piecewise affine, $\theta(u) = \theta(\hat{u}) + \theta'(\hat{u}, u-\hat{u})$ for $u - \hat{u}$ small enough; and $\theta'(\hat{u}, u-\hat{u}) = 0$ if $u - \hat{u} \perp \partial\theta(\hat{u})$.

Table 3. Comparison of poor and rich versions

Problem	10^{-2}		10^{-3}		10^{-4}		CPU (10^{-3})	
gr120	21	24	93	72	173	102	0s	0s
pcb442	24	23	131	79	233	261	10s	6s
pcb1173	27	22	104	93	236	187	61s	63s
pcb3038	26	28	128	146	6920	782	540s	630s
fnl4461	23	21	126	120	411	811	760s	800s

starting from an optimal u_1 (obtained from the runs in Table 1); $t > 0$ is kept small. Then only null-steps are performed and every sampling point u_k is close to $\hat{u} = u_1$, so that $g_k \in \partial\theta(\hat{u})$[19].

Table 4. Proving optimality

Problem	$-\theta(u_K)$	K	K_{d}	ℓ^*	final $\|\hat{g}\|$
gr120	1606.3125	47	18	42	2×10^{-15}
pcb442	50500.	105	24	105	3×10^{-10}
pcb1173	56361.	93	27	91	5×10^{-11}
pcb3038	136587.5	700	39	570	2×10^{-6}
fnl4461	181569.21	357	35	357	1×10^{-10}

The results are given in Table 4. Because we are dealing with approximations only, the tests serve as purification: the starting u_1 is not exactly optimal, a few descent-steps may occur, the u_K's may not all be in an appropriate face of $\operatorname{epi}\theta$, the final ℓ-value may not be the same as in Table 1. This is why we report again the final θ_K, the number K_{d} of descent-steps, and ℓ. The latter is denoted by ℓ^*, to be distinguished from the ℓ of Table 1. Of course, we must have $K \geqslant \ell^*$; actually, $K - \ell^*$ represents the number of "errors" made by the algorithm in identifying the necessary optimal faces; said otherwise, the algorithm computes K subgradients at $\hat{u}$ (or close to), ℓ^* of which are finally used to make up the 0 vector. Table 4 gives also the norm of this (approximate) 0 vector, whose accuracy is assessed by the fact that each g_k from the oracle is an integer vector, and therefore has norm larger than 1.

Concluding Remarks. Inspection of the above numerical results suggests a number of comments.

(i) Bundle methods appear as reasonably efficient alternatives to traditional algorithms for Lagrangian relaxation, hence for column generation. Note in particular that all our reported experiments used a fairly simple oracle,

[19] i.e. $(u_k, \theta(u_k))$ lies in a face of $\operatorname{epi}\theta$ containing the optimal $(\hat{u}, \min\theta)$.

delivering one column at a time. Notwithstanding, serious comparisons are still to be made between bundle, subgradient, cutting planes and ACCPM.

(ii) A dual problem of large dimension m is likely to be difficult. However, ℓ^*, which resembles the dimensionality of an optimal $\partial\theta$, seems a lot more important: a problem with small ℓ^* – or with a solution set of large dimension – seems relatively "easy".

(iii) Just as finiteness, piecewise affinity is a concept that should be taken with care: there is no clearcut between a piecewise affine θ (finitely many possible answers from the oracle) and a truly nonlinear one (a continuum of possible answers). This is illustrated by some observations, in our present context:
- Any run of gr120, with various values given to various parameters (starting point, stopping tolerance, bundle size, etc.), invariably produced $\ell^* = 42$. Besides, $\|\hat{g}\|$ decreased regularly along the iterations down to about 10^{-5}, and then fell abruptly to 10^{-15}, clearly revealing finite convergence.
- For all other test-problems, variations in the parameters produced fluctuations of about 10% in the final ℓ^*. This occurred even with pcb1173 (seemingly easy), and also when starting from an optimal u_1. Besides, $\|\hat{g}\|$ decreased regularly to its minimal value around 10^{-10}. Such a behaviour definitely connotes a truly nonlinear problem.

(iv) The poorman variant is surprisingly efficient. With Remark 31 in mind, it assesses older works such as [6,5]. Good behaviour of cheap bundle methods was already observed in [34] and related works. Such variants are of course very attractive, as they use little computing time, and are relatively easy to encode (although a good knowhow is strongly advised: nonsmooth optimization is by no means an easy exercise).

The above observation can be turned the other way: "richman" bundle methods are disappointing (observe in particular the disaster for fnl4461 in Table 3, in which the richer version is the slower). The information contained in a bundle of size 1000 has to be worth something: if this information does not help to improve efficiency, for sure it is misused; and of course the culprit is the Euclidean norm (a "poorman" stabilizer). This is why research has been and still is conducted, to define richer stabilizers: see Remark 29 again. Some works already exist proposing stabilizers of the form $\frac{1}{2}(u-\hat{u})^\top M_k(u-\hat{u})$: see [51,17,53], and also [48], where convexity of θ is not even used. On this "quasi-second-order" subject, the final word is far from being said.

(v) By contrast, Table 4 illustrates quite an interesting behaviour: since ℓ^* is a lower bound for K, the algorithm could hardly do better in the situation of the experiments[20]. See also the results in [25, § IX.3.3], which display a similar behaviour. The role of the stabilizer is here blatantly demonstrated.

[20] However pcb3038 shows that fast convergence is never guaranteed. Our trials to obtain$\|\hat{g}\|$ around 10^{-10} remained fruitless: the quadratic solver ran into difficulties, K was much larger than ℓ^*, the algorithm had to be restarted several times, etc.

Indeed the conditions are here most favourable for θ to look like a piecewise affine function: many answers from the oracle are eliminated from the picture. Yet the far-from-piecewise-affine model $\breve{\theta}$ of (31) allows an accurate description of θ.

We believe that here lies the crux of nonsmooth optimization. To minimize a nonsmooth function such as θ, two processes are involved:

– To diminish θ down to its minimal value; in Lagrangian relaxation, this means to generate best possible upper bounds for the primal problem.
– To prove optimality of $\hat{u}$, i.e. to genarate an $\hat{x}$ which solves the relaxed primal problem. In cutting-plane methods of §4.2, each $\hat{x}$ is feasible, what is needed is to increase the corresponding value $\sum_k \alpha_k f(x_k)$ to $\theta(\hat{u})$. In a bundle method, the problem is mainly to decrase $\|\hat{g}\|$ down to 0.

Because of our two observations (iv) and (v), the two processes seem fairly unrelated and call for fairly different tools. The difficulty is to find an appropriate model (such as $\hat{\theta}$ or $\breve{\theta}$) in which they groove together harmoniously.

References

1. F. Alizadeh. Interior point methods in semidefinite programming with applications to combinatorial optimization. *SIAM Journal on Optimization*, 5(1):13–51, 1995.
2. K. Anstreicher and L.A. Wolsey. On dual solutions in subgradient opimization. Unpublished manuscript, CORE, Louvain-la-Neuve, Belgium, 1993.
3. D.P. Bertsekas. Projected Newton methods for optimization problems with simple constraints. *SIAM Journal on Control and Optimization*, 20:221–246, 1982.
4. M.J. Best. Equivalence of some quadratic programming algorithms. *Mathematical Programming*, 30:71–87, 1984.
5. U. Brännlund. *On relaxation methods for nonsmooth convex optimization.* PhD thesis, Royal Institute of Technology - Stockholm, 1993.
6. P.M. Camerini, L. Fratta, and F. Maffioli. On improving relaxation methods by modified gradient techniques. *Mathematical Programming Study*, 3:26–34, 1975.
7. E. Cheney and A. Goldstein. Newton's method for convex programming and Tchebycheff approximations. *Numerische Mathematik*, 1:253–268, 1959.
8. R. Correa and C. Lemaréchal. Convergence of some algorithms for convex minimization. *Mathematical Programming*, 62(2):261–275, 1993.
9. A. Decarreau, D. Hilhorst, C. Lemarechal, and J. Navaza. Dual methods in entropy maximization. application to some problems in crystallography. *SIAM Journal on Optimization*, 2(2):173–197, 1992.
10. I. Ekeland and R. Temam. *Convex Analysis and Variational Problems.* North Holland, 1976; reprinted by SIAM, 1999.
11. Y.M. Ermol'ev. Methods of solution of nonlinear extremal problems. *Kibernetica*, 2(4):1–17, 1966.
12. H. Everett III. Generalized lagrange multiplier method for solving problems of optimum allocation of resources. *Operations Research*, 11:399–417, 1963.
13. J.E. Falk. Lagrange multipliers and nonconvex programs. *SIAM Journal on Control*, 7(4):534–545, 1969.
14. S. Feltenmark and K. C. Kiwiel. Dual applications of proximal bundle methods, including lagrangian relaxation of nonconvex problems. *SIAM Journal on Optimization*, 10(3):697–721, 2000.

15. R. Fletcher. *Practical Methods of Optimization*. John Wiley & Sons, Chichester (second edition), 1987.
16. A. Frangioni. Solving semidefinite quadratic problems within nonsmooth optimization algorithms. *Computational Operational Research*, 23(11):1099–1118, 1996.
17. M. Fukushima and L. Qi. A globally and superlinearly convergent algorithm for nonsmooth convex minimization. *SIAM Journal on Optimization*, 6:1106–1120, 1996.
18. C. Garrod and J.K. Percus. Reduction of the N-particle variational problem. *Journal of Mathematical Physics*, 5(12), 1964.
19. A.M. Geoffrion. Lagrangean relaxation for integer programming. *Mathematical Programming Study*, 2:82–114, 1974.
20. M. X. Goemans and D. P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, 6:1115–1145, 1995.
21. J.L. Goffin, A. Haurie, and J.Ph. Vial. Decomposition and nondifferentiable optimization with the projective algorithm. *Management Science*, 38(2):284–302, 1992.
22. J.L. Goffin and J.Ph. Vial. Convex nondifferentiable optimization: a survey focussed on the analytic center cutting plane method. to appear in Optimization Methods and Software; also as HEC/Logilab Technical Report 99.02, Univ. of Geneva, Switzerland.
23. M. Held and R. Karp. The travelling salesman problem and minimum spanning trees: Part II. *Mathematical Programming*, 1:6–25, 1971.
24. C. Helmberg and F. Rendl. A spectral bundle method for semidefinite programming. *SIAM Journal on Optimization*, 10(3):673–696, 2000.
25. J.-B. Hiriart-Urruty and C. Lemaréchal. *Convex Analysis and Minimization Algorithms*. Springer Verlag, Heidelberg, 1993.
26. J.-B. Hiriart-Urruty and C. Lemaréchal. *Fundamentals of Convex Analysis*. Springer Verlag, Heidelberg, 2001. to appear.
27. R.A. Horn and Ch.R. Johnson. *Matrix Analysis*. Cambridge University Press, 1989. (New edition, 1999).
28. J. E. Kelley. The cutting plane method for solving convex programs. *J. Soc. Indust. Appl. Math.*, 8:703–712, 1960.
29. K.C. Kiwiel. *Methods of Descent for Nondifferentiable Optimization*. Lecture Notes in Mathematics 1133. Springer Verlag, Heidelberg, 1985.
30. K.C. Kiwiel. A method for solving certain quadratic programming problems arising in nonsmooth optimization. *IMA Journal of Numerical Analysis*, 6:137–152, 1986.
31. K.C. Kiwiel. A dual method for certain positive semidefinite quadratic programming problems. *SIAM Journal on Scientific and Statistical Computing*, 10:175–186, 1989.
32. K.C. Kiwiel. Proximity control in bundle methods for convex nondifferentiable minimization. *Mathematical Programming*, 46(1):105–122, 1990.
33. K.C. Kiwiel. Efficiency of proximal bundle methods. *Journal of Optimization Theory and Applications*, 104(3):589–603, 2000.
34. K.C. Kiwiel, T. Larsson, and P.O. Lindberg. The efficiency of ballstep subgradient level methods for convex optimization. *Mathematics of Operations Research*, 24(1):237–254, 1999.
35. T. Larsson, M. Patriksson, and A.B. Strömberg. Ergodic, primal convergence in dual subgradient schemes for convex programming. *Mathematical Programming*, 86(2):283–312, 1999.
36. C. Lemaréchal, A.S. Nemirovskii, and Yu.E. Nesterov. New variants of bundle methods. *Mathematical Programming*, 69:111–148, 1995.

37. C. Lemaréchal, Yu. Nesterov, and F. Oustry. Duality gap analysis for problems with quadratic constraints, 2001. In preparation.
38. C. Lemaréchal and F. Oustry. Semi-definite relaxations and lagrangian duality with application to combinatorial optimization. Rapport de Recherche 3710, Inria, 1999. `http://www.inria.fr/rrrt/rr-3710.html`.
39. C. Lemaréchal and F. Oustry. Nonsmooth algorithms to solve semidefinite programs. In L. El Ghaoui and S-I. Niculescu, editors, *Advances in Linear Matrix Inequality Methods in Control*, Advances in Design and Control, 2, pages 57–77. SIAM, 2000.
40. C. Lemaréchal, F. Oustry, and C. Sagastizábal. The $\mathcal{U}$-lagrangian of a convex function. *Transactions of the AMS*, 352(2):711–729, 2000.
41. C. Lemaréchal, F. Pellegrino, A. Renaud, and C. Sagastizábal. Bundle methods applied to the unit-commitment problem. In J. Doležal and J. Fidler, editors, *System Modelling and Optimization*, pages 395–402, 1996.
42. C. Lemaréchal and A. Renaud. A geometric study of duality gaps, with applications. *Mathematical Programming*, 90(3):399–427, 2001.
43. C. Lemaréchal and C. Sagastizábal. Variable metric bundle methods: from conceptual to implementable forms. *Mathematical Programming*, 76(3):393–410, 1997.
44. C. Lemaréchal and J. Zowe. Some remarks on the construction of higher order algorithms in convex optimization. *Applied Mathematics and Optimization*, 10(1):51–68, 1983.
45. C. Lemaréchal and J. Zowe. The eclipsing concept to approximate a multi-valued mapping. *Optimization*, 22(1):3–37, 1991.
46. L. Lovász. On the Shannon capacity of a graph. *IEEE Transactions on Information Theory*, IT 25:1–7, 1979.
47. D.G. Luenberger. *Optimization by Vector Space Methods*. Wiley, New York, 1969.
48. L. Luksan and J. Vlcek. A bundle-newton method for nonsmooth unconstrained minimization. *Mathematical Programming*, 83A(3):373–391, 1998.
49. T.L. Magnanti, J.F. Shapiro, and M.H. Wagner. Generalized linear programming solves the dual. *Management Science*, 22(11):1195–1203, 1976.
50. R.E. Marsten, W.W. Hogan, and J.W. Blankenship. The boxstep method for large-scale optimization. *Operations Research*, 23(3):389–405, 1975.
51. R. Mifflin. A quasi-second-order proximal bundle algorithm. *Mathematical Programming*, 73(1):51–72, 1996.
52. R. Mifflin and C. Sagastizábal. On $\mathcal{VU}$-theory for functions with primal-dual gradient structure. *SIAM Journal on Optimization*, 11(2):547–571, 2000.
53. R. Mifflin, D.F. Sun, and L.Q. Qi. Quasi-Newton bundle-type methods for nondifferentiable convex optimization. *SIAM Journal on Optimization*, 8(2):583–603, 1998.
54. A.S. Nemirovskii and D. Yudin. Informational complexity and efficient methods for the solution of convex extremal problems. *Ékonomika i Mathematicheskie Metody*, 12:357–369, 1976. (in Russian. English translation: *Matekon*, 13, 3-25).
55. Yu.E. Nesterov. Complexity estimates of some cutting plane methods based on the analytic barrier. *Mathematical Programming*, 69(1):149–176, 1995.
56. Yu.E. Nesterov and A.S. Nemirovskii. *Interior-Point Polynomial Algorithms in Convex Programming*. Number 13 in SIAM Studies in Applied Mathematics. SIAM, Philadelphia, 1994.
57. Yu.E. Nesterov and A.S. Nemirovskii. *Interior-Point Polynomial Algorithms in Convex Programming*. SIAM Studies in Applied Mathematics 13. SIAM, Philadelphia, 1994.

58. J. Nocedal and S.J. Wright. *Numerical Optimization*. Springer Verlag, New York, 1999.
59. F. Oustry. A second-order bundle method to minimize the maximum eigenvalue function. *Mathematical Programming*, 89(1):1–33, 2000.
60. S. Poljak, F. Rendl, and H. Wolkowicz. A recipe for semidefinite relaxation for (0,1)-quadratic programming. *Journal of Global Optimization*, 7:51–73, 1995.
61. B.T. Polyak. A general method for solving extremum problems. *Soviet Mathematics Doklady*, 8:593–597, 1967.
62. B.N. Pshenichnyi. *The Linearization Method for Constrained Optimization*. Springer Verlag, 1994.
63. C.R. Reeves. *Modern Heuristic Techniques for Combinatorial Problems*. Blackwell Scientific Publications, New York, 1993.
64. N. Shor. Utilization of the operation of space dilatation in the minimization of convex functions. *Cybernetics*, 6(1):7–15, 1970.
65. N.Z. Shor. *Minimization methods for non-differentiable functions*. Springer Verlag, Berlin, 1985.
66. N.Z. Shor and A.S. Davydov. Method of opbtaining estimates in quadratic extremal probems with boolean variables. *Cybernetics*, 21(2):207–211, 1985.
67. N.Z. Shor and N.G. Zhurbenko. A method for minimization, using the space-dilation operation in the direction of difference between two gradient sequences. *Cybernetics*, 7(3):450–459, 1971.
68. V.N. Solov'ev. The subdifferential and the directional derivatives of the maximum of a family of convex functions. *Izvestiya: Mathematics*, 62(4):807–832, 1998.
69. G. Sonnevend. An "analytical centre" for polyhedra and new classes of global algorithms for linear (smooth, convex) programming. In A. Prékopa, J. Szelezsan, and B. Strazicky, editors, *Proc. 12th IFIP Conf. System Modelling and Optimization*, L.N. in Control and Information Sciences, pages 866–875. Springer Verlag, Berlin, 1986.
70. D.M. Topkis. A note on cutting-plane methods without nested constraint sets. *Operations Research*, 18:1216–1224, 1970.
71. T.Terlaky. On ℓ_p programming. *European Journal of Operational Research*, 22:70–100, 1985.
72. H. Uzawa. Iterative methods for concave programming. In K. Arrow, L. Hurwicz, and H. Uzawa, editors, *Studies in Linear and Nonlinear Programming*, pages 154–165. Stanford University Press, 1959.
73. L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review*, 38(1):49–95, 1996.
74. P. Wolfe. A method of conjugate subgradients for minimizing nondifferentiable functions. *Mathematical Programming Study*, 3:145–173, 1975.
75. L.A. Wolsey. *Integer Programming*. Wiley-Interscience, 1998.
76. S.J. Wright. *Primal-Dual Interior-Point Methods*. SIAM Publication, Philadelphia, 1997.