Control
and Cybernetics
VOL. 15 (1986) No. 3-4

A decomposition method for linear programs with dual block angular structure

by

KRZYSZTOF C. KIWIEL

Systems Research Institute
Polish Academy of Sciences
Newelska 6
01-447 Warszawa, Poland

This paper presents a decomposition method for solving large-scale linear programs with dual block-angular structure. The method stems from nondifferentiable optimization algorithms of descent, requires a priori bounded storage and has finite convergence. Its quadratic programming subproblems can be solved efficiently by the existing software for large-scale optimization.

1. Introduction

This paper presents a decomposition method for solving linear programs with the following dual block-angular structure

$$\begin{aligned} &\text{minimize } \sum_{i=1}^{n} \langle c^i, z^i \rangle \quad \text{over all} \quad z^i \in R^{n_i}, \quad x \in R^N \\ &\text{satisfying } A_i z^i + B_i x \leqslant b^i \quad \text{for} \quad i=1, \ldots, n, \end{aligned} \tag{1.1}$$

where $c^i \in R^{n_i}$, $b^i \in R^{m_i}$, A_i and B_i are $m_i \times n_i$ and $m_i \times N$ matrices, respectively, whereas $\langle \cdot, \cdot \rangle$ denotes the usual inner product. Under reasonable conditions (see. e.g. [8]), the functions

$$f_i(x) = \min \{\langle c^i, z^i \rangle : A_i z^i \leqslant b^i - B_i x\} \tag{1.2}$$

are finite-valued, convex and piecewise linear (polyhedral), with easily computable subgradients. Then problem (1.1) may be restated as

$$\text{minimize } f(x) := \sum_{i=1}^{n} f_i(x) \quad \text{over all} \quad x \in R^N.$$

The method presented in this paper modifies the one given in [2] (see also [3]) to make use of the special structure of f. We suppose that for each $x \in R^N$ we can compute $f_i(x)$ and an arbitrary subgradient $g_f(x) \in \partial f_i(x)$ of f_i at x, for $i=1, \ldots, n$.

The method is based on the observation that for any $i \in I=\{1, ..., n\}$ and $I_i=I\setminus\{i\}$ the objective f can be expressed as

$$f(x)=f_i(x)+f_{(i)}(x) \quad \text{for all} \quad x,$$

where the function

$$f_{(i)}(x)=\sum_{l\in I_i} f_l(x)$$

can be approximated by the polyhedral function

$$\hat{f}_{(i)}(x)=\max\{f_{(i)}(y)+\langle g_{(i)}(y), x-y\rangle : y \in Y_i\} \tag{1.3}$$

defined for any finite set $Y_i \subset R^N$ by the subgradient mapping

$$g_{(i)}(x)=\sum_{l\in I_i} g_{f_l}(x).$$

Our algorithm is a descent method in the sense of generating iterates $x^k \in R^N$ for $k=1, 2, ...$ with $f(x^{k+1})<f(x^k)$ if $x^{k+1}\neq x^k$. Also a sequence of trial points $\{y^k\}\subset R^N$ is computed. At the k-th iteration, n auxiliary points $y^{k+1,i}$ are computed by minimizing for $i=1, ..., n$ the functions

$$\hat{f}_i^k(x)=f_i(x)+\hat{f}_{(i)}^k(x)+\frac{1}{2}|x-x^k|^2$$

over all x in R^N, where $\hat{f}_{(i)}^k$ is defined by (1.3) with $Y_i\subset\{1, ..., k\}$ having at most $N+2$ elements, and $|\cdot|$ is the Euclidean norm. The combined trial point

$$y^{k+1}=\frac{1}{n}\sum_{i=1}^{n} y^{k+1,i} \tag{1.4}$$

becomes x^{k+1} only if it reduces the objective value significantly; otherwise, a null step $x^{k+1}=x^k$ occurs, but the new subgradients computed at y^{k+1} will enrich the polyhedral models (1.3) at the next iteration, thus helping to find a better next trial point.

In effect, each iteration involves n subproblems of the form

$$\underset{(z^i,\,u,\,x)\in R^{n_i+1+N}}{\text{minimize}} \quad \langle c^i, z^i\rangle+u+\frac{1}{2}|x-x^k|^2 \tag{1.5a}$$

$$\text{subject to } A_i z^i+B_i x\leqslant b^i, \tag{1.5b}$$

$$f_{(i)}(y^j)+\langle g_{(i)}(y^j), x-y^j\rangle\leqslant u \quad \text{for} \quad j\in J_i^k. \tag{1.5c}$$

In problems of interest to us, the numbers n_i of "internal" variables (in z^i) of problem (1.1) are much larger than the number N of "linking" variables (in x), and the matrices A_i and B_i are large, but sparse. In such cases subproblem (1.5) can be solved efficiently by the existing software for large-scale optimization, e.g. the MINOS code [5]. We refer the reader to [1, 6] for specific examples of possible applications of our method.

Relations of our method with the existing algorithms are discussed in a companion paper [4]. Here we only note that our method seems to be unique in possessing the three properties: descent, a priori bounded storage, and finite termination.

The paper is organized as follows. The algorithm is stated and discussed in Section 2. Section 3 contains a result on finite convergence of the method. Finally, we have a conclusion section.

We denote by

$$\partial f(x)=\{g\in R^N: f(y)\geqslant f(x)+\langle g, y-x\rangle \quad \text{for all} \quad y\in R^N\}$$

the subdifferential of f at x.

2. The method

We shall first state the method in detail, commenting on its rules in what follows.

ALGORITHM 2.1.

STEP 0 (INITIALIZATION). Select a starting point $x^1\in R^N$, a final accuracy tolerance $\varepsilon_s\geqslant 0$ and a line search parameter $m\in(0, 1)$. Set $y^1=x^1$ and $J_i^1=\{1\}$ for $i\in I$. Set $k=1$.

STEP 1 (TRIAL POINT FINDING).). For each $i\in I$, find the solution $(z^i, u_i^k, y^{k+1,i})$ to subproblem (1.5). Let $\hat{J}_i^k$ denote the set of indices j with nonzero Lagrange multipliers for constraints (1.5c), $i\in I$. Compute y^{k+1} by (1.4) and

$$v_i^k=f_i(y^{k+1,i})+u_i^k-f(x^k), \quad i\in I, \quad v^k=\frac{1}{n}\sum_{i=1}^{n} v_i^k.$$

STEP 2 (STOPPING CRITERION). If $v_i^k\geqslant-\varepsilon_s$ for some $i\in I$, terminate; otherwise, continue.

STEP 3 (TRIAL POINT TESTING). If

$$f(y^{k+1})\leqslant f(x^k)+mv^k,$$

set $x^{k+1}=y^{k+1}$ (serious step); otherwise, set $x^{k+1}=x^k$ (null step).

STEP 4 (APPROXIMATION UPDATING). Set

$$J_i^{k+1}=\hat{J}_i^k\cup\{k+1\} \quad \text{for} \quad i\in I.$$

Increase k by 1 and go to Step 1.

A few remarks on the algorithm are in order.

Observe that typical procedures for solving subproblem (1.5) will automatically produce at most $N+1$ nonzero Lagrange multipliers for the linear constraints (1.5c), since these constraints involve only $N+1$ variables. This will be the case if MINOS [5] is used.

In Step 2 we always have

$$f(x^k)\leqslant \inf\{f(x): x\in R^N\}-v_i^k+|v_i^k|^{1/2}\sup\{|x-x^k|: f(x)\leqslant f(x^k)\}$$

(see [4]). The above optimality estimate justifies the stopping criterion of the method. If no termination occurs, Step 3 is entered with $v^k<-\varepsilon\leqslant 0$.

The sets J_i^{k+1} are selected so that the new linear pieces from y^{k+1} and the "active" pieces indexed by $\hat{J}_i^k$ (which contributed to $y^{k+1,i}$) are retained, whereas the remaining pieces are dropped. Note that y^j need not be stored, since one can use the recursive formulae

$$f_{(i)}(y^j)+\langle g_{(i)}(y^j), x-y^j\rangle=\hat{f}_{(i)}(x^k; y^j)+\langle g_{(i)}(y^j), x-x^k\rangle,$$

$$\hat{f}_{(i)}(x^{k+1}; y^j)=\hat{f}_{(i)}(x^k; y^j)+\langle g_{(i)}(y^j), x^{k+1}-x^k\rangle$$

at Step 4.

3. Convergence

Convergence properties of the method are analyzed in detail in [4] (including the case of not necessarily polyhedral f_i). We shall now strengthen the analysis of [4] by presenting a result on finite termination in the polyhedral case.

We suppose that each f_i can be represented as the pointwise maximum of a finite number of affine functions, and that the subgradient mappings g_{f_i} have finitely many values. This assumption is realistic when the minimization in (1.2) is carried out by a simplex-like method, since then we may compute

$$g_{f_l}(x)=(\lambda^l)^T B_l,$$

where λ^l is the Lagrange multiplier of (1.2). Moreover, we assume that f is bounded from below, i.e. problem (1.1) has a solution. We also suppose that Algorithm 2.1 uses parameter values $\varepsilon_s=0$ and $m=1$ (the case of $\varepsilon_s\geqslant 0$ and $m\in(0, 1)$ is discussed in [4]).

THEOREM 3.1 *Under the preceding assumptions Algorithm* 2.1 *will terminate in a finite number of iterations with a point* x^k *that minimizes* f.

Proof. (Due to lack of space, we only give an outline of the proof.) For contradiction purposes, suppose that the algorithm does not stop.

(i) Suppose that only finitely many serious steps occur, i.e. $x^{k+1}=x^k$ for all large k. Reasoning as in the proof of Lemma 3.5 in [4], one can show that after each null step the optimal value of at least one of the n subproblems (1.5) strictly increases, whereas the remaining optimal values do not decrease. But we have constant x^k for large k and only a finite number of possible linear constraints (1.5c) (generated by the finitely many pieces of f_l, $l\in I$), so there is but a finite number of possible optimal values of subproblems (1.5), a contradiction.

(ii) Next, suppose that the algorithm executes infinitely many serious steps. define a subsequence $\{z^l\}$ of $\{x^k\}$ by setting $z^1=x^1$ and $l=1$ at Step 0, and at Step 3 by setting $z^{l+1}=x^{k+1}$ and increasing l by 1 if $x^{k+1}\neq x^k$, so that $f(z^{l+1})\leqslant f(z^l)+v^k$ for all l and the corresponding k. Using this fact and simple properties of subproblems (1.5), one can show that

$$z^{l+1}=\arg\min\left\{f(z)+\frac{1}{2}|z-z^l|^2 : z\in R^N\right\} \text{ for all } l.$$

Hence the sequence $\{z^l\}$ is generated as in the proximal point method of Rockafellar [7], and [7, Proposition 8] implies that z^l is constant for large l. But $z^{l+1}\neq z^l$ for all l by construction. This contradiction completes the proof. ■

4. Conclusions

We have presented a new decomposition method for large-scale linear programs with dual block-angular structure. It differs from the existing algorithms in that it is a descent method with finite termination and a priori bounded storage.

We shall present elsewhere an extension of the method for problems with induced constraints and not necessarily finite-valued partial objectives (1.2).

References

[1] Cohen G., Balducchi J. F. Computation of saddle-points of non-strictly convex Lagrangians. In: Current Problems in Computational and Applied Mathematics (A. S. Alekseiev, ed.). Novosibirsk Nauka, 1983.

[2] Kiwiel K. C. An aggregate subgradient method for nonsmooth convex minimization. *Mathematical Programming*, **27** (1983), 320–341.

[3] Kiwiel K. C. Methods of Descent for Nondifferentiable Optimization. Lecture Notes in Mathematics 1133. Berlin, Springer, 1985.

[4] Kiwiel K. C. A decomposition method of descent for minimizing a sum of convex functions. *Journal of Optimization Theory and Applications* **52** (1987), 1-17.

[5] Murtagh B. A., Saunders M. A. Large-scale linearly constrained optimization. *Mathematical Programming*, **14** (1978), 41–72.

[6] Nurminski E., Balabanov T. Decomposition of a large-scale energy model. *Large Scale Systems*, **4** (1983), 295–308.

[7] Rockafellar R. T. Monotone operators and the proximal point algorithm, *SIAM Journal of Control and Optimization*, **14** (1976), 877–898.

[8] Shor N. Z. Minimization Methods for Nondifferentiable Functions. Berlin, Springer, 1985.

Metoda dekompozycji dla zadań programowania liniowego z dualną blokowo trójkątną strukturą

W pracy przedstawiono metodę dekompozycji dla zadań programowania liniowego wielkiej skali z dualną blokowo trójkątną strukturą. Metoda ta wywodzi się z algorytmów spadku dla optymalizacji nieróżniczkowalnej, wymaga ograniczonej pamięci i ma skończoną zbieżność. Jej podzadania programowania kwadratowego można efektywnie rozwiązywać istniejącymi programami optymalizacji wielkiej skali.

Метод декомпозиции для задач линейного программирования с дуальной блочно-треугольной структурой

В работе представлен метод декомпозиции для решения больших задач линейного программирования с дуальной блочно-треугольной структурой. Этот метод исходит из методов спуска для недифференцируемой оптимизации, использует ограниченную памаять и имеет конечную сходимость. Его подзадачи квадратического программирования можно эффективно решать существующими программами оптимизации большой размерности.