European Journal of Operational Research 239 (2014) 437–448

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Stochastics and Statistics

Applying oracles of on-demand accuracy in two-stage stochastic programming – A computational study

Christian Wolf [a], Csaba I. Fábián [b], Achim Koberstein [c,*], Leena Suhl [a]

[a] DS&OR Lab, University of Paderborn, Warburger Str. 100, 33098 Paderborn, Germany
[b] Department of Informatics, Kecskemét College, 10 Izsáki út, 6000 Kecskemét, Hungary
[c] Goethe University Frankfurt, Grüneburgplatz 1, 60323 Frankfurt am Main, Germany

ARTICLE INFO

Article history:
Received 30 September 2013
Accepted 6 May 2014
Available online 17 May 2014

Keywords:
Stochastic programming
Two-stage problems
Decomposition
Bundle methods

ABSTRACT

Traditionally, two variants of the L-shaped method based on Benders' decomposition principle are used to solve two-stage stochastic programming problems: the aggregate and the disaggregate version. In this study we report our experiments with a special convex programming method applied to the aggregate master problem. The convex programming method is of the type that uses an oracle with on-demand accuracy. We use a special form which, when applied to two-stage stochastic programming problems, is shown to integrate the advantages of the traditional variants while avoiding their disadvantages. On a set of 105 test problems, we compare and analyze parallel implementations of regularized and unregularized versions of the algorithms. The results indicate that solution times are significantly shortened by applying the concept of on-demand accuracy.

1. Introduction

Decomposition is an effective and time-honoured means of handling two-stage stochastic programming problems. It can be interpreted as a cutting-plane scheme applied to the first-stage variables. Traditionally, there are two approaches: one can use a disaggregate or an aggregate model. A major drawback of the aggregate model is that an aggregate master problem cannot contain all the information obtained by the solution of the second-stage problems. The disaggregate master problem, on the other hand, may grow excessively. It is not easy to find a balance between the effort spent in solving the master problem on the one hand, and the second-stage problems on the other hand. The computational results of Wolf and Koberstein (2013) give insights into this question.

In this study we report our experiments with a special inexact convex programming method applied to the aggregate master problem of the two-stage stochastic programming decomposition scheme. The convex programming method is of the type that uses an oracle with on-demand accuracy, a concept proposed by Oliveira and Sagastizábal (2014). We are going to use a special form which, when applied to two-stage stochastic programming problems, integrates the advantages of the aggregate and the disaggregate models. This latter feature is discussed in Fábián (2012). We also examine the on-demand accuracy idea in an un-regularized context, which results a pure cutting-plane method in contrast to the level bundle methods treated in Oliveira and Sagastizábal (2014).

The paper is organized as follows. In Section 1.1 we outline the on-demand accuracy approach to convex programming, and present an algorithmic sketch of the partly inexact level method. In Section 2 we overview two-stage stochastic programming models and methods. Specifically, in Section 2.1 we sketch a decomposition method for two-stage problems based on the partly inexact level method. Section 3 discusses implementation issues. Our computational results are reported in Section 4, and conclusions are drawn in Section 5.

1.1. Convex programming: applying oracles of on-demand accuracy

Let us consider the problem

$$\begin{array}{ll} \min & \varphi(\boldsymbol{x}) \\ \text{such that} & \boldsymbol{x} \in X, \end{array}$$

where $\varphi : \mathrm{IR}^n \to \mathrm{IR}$ is a convex function, and $X \subset \mathrm{IR}^n$ is a convex closed bounded polyhedron. We assume that φ is Lipschitz continuous over X with the constant Λ.

* Corresponding author. Tel.: +49 6979834626; fax: +49 6979834526.
E-mail addresses: christian.wolf@dsor.de (C. Wolf), fabian.csaba@gamf.kefo.hu (C.I. Fábián), koberstein@wiwi.uni-frankfurt.de (A. Koberstein), suhl@dsor.de (L. Suhl).

http://dx.doi.org/10.1016/j.ejor.2014.05.010
0377-2217/

Oliveira and Sagastizábal (2014) developed special regularization methods for unconstrained convex optimization, namely, bundle level methods that use oracles with on-demand accuracy. The methods work with approximate function data, which is especially useful in solving stochastic problems. Approximate function values and subgradients are provided by an oracle with on-demand accuracy. The accuracy of the oracle is regulated by two parameters: the first is a descent target, and the second is a tolerance. If the estimated function value reaches the descent target, then the prescribed tolerance is observed. Otherwise the oracle just detects that the target cannot be met, and returns rough estimations of the function data, disregarding the prescribed tolerance. The method includes the ideas of Lemaréchal, Nemirovskii, and Nesterov (1995), Kiwiel (1995) and Fábián (2000); and integrates the level-type and the proximal approach.

In this paper we are going to use a special method that falls into the 'partly inexact' category according to Oliveira and Sagastizábal, and applies only the level regularization of Lemaréchal et al. (1995). The method is discussed in detail in Fábián (2012).

In the following description, $\bar{\phi}$ denotes the best function value known, and $\underline{\phi}$ is a lower estimate of the optimum. The gap $\Delta = \bar{\phi} - \underline{\phi}$ measures the quality of the current approximation. The descent target is $\bar{\phi} - \delta$, where the tolarence δ is regulated by the current gap. If the descent target is reached, then the oracle returns an exact subgradient. Otherwise the oracle just detects that the target cannot be met, and returns rough estimations of the function data. Iterations where the descent target is reached will be called substantial.

Algorithm 1. A partly inexact level method.

1.0 Parameter setting.
Set the stopping tolerance $\epsilon > 0$.
Set the level parameter λ $(0 < \lambda < 1)$.)
Set the tolerance regulating parameter γ such that $0 < \gamma < (1-\lambda)^2$.

1.1 Bundle initialization.
Let $i = 1$ (iteration counter).
Find a starting point $\boldsymbol{x}_1 \in X$.
Let $l_1(\boldsymbol{x})$ be a linear support function to $\varphi(\boldsymbol{x})$ at $\boldsymbol{x}_1$
Let $\delta_1 = 0$ (meaning that l_1 is an exact support function).

1.2 Near-optimality check.
Compute $\bar{\phi}_i = \min_{1 \leqslant j \leqslant i} \varphi(\boldsymbol{x}_j)$.
Let $\underline{\phi}_i = \min_{\boldsymbol{x} \in X} \varphi_i(\boldsymbol{x})$, where $\varphi_i(\boldsymbol{x}) = \max_{1 \leqslant j \leqslant i} l_j(\boldsymbol{x})$ is the current model function.
Let $\Delta_i = \bar{\phi}_i - \underline{\phi}_i$. If $\Delta_i < \epsilon$ then near-optimal solution found, stop.

1.3 Finding a new iterate.
Let $\boldsymbol{x}_{i+1}$ be the projection of $\boldsymbol{x}_i$ onto $X_i = \left\{ \boldsymbol{x} \in X \mid \varphi_i(\boldsymbol{x}) \leqslant \underline{\phi}_i + \lambda \Delta_i \right\}$.

1.4 Bundle update.
Let $\delta_{i+1} = \gamma \Delta_i$.
Let $l_{i+1}(\boldsymbol{x})$ be a linear function such that $l_{i+1}(\boldsymbol{x}) \leqslant \varphi(\boldsymbol{x})$ $(\boldsymbol{x} \in X)$, $\|\nabla l_{i+1}\| \leqslant \Lambda$, and
either $l_{i+1}(\boldsymbol{x}_{i+1}) \geqslant \bar{\phi}_i - \delta_{i+1}$ (descent target could not be *reached*),
or $l_{i+1}(\boldsymbol{x}_{i+1}) = \varphi(\boldsymbol{x}_{i+1})$ (descent target has been reached).
Increment i, and repeat from step *1.2*.

In step *1.3*, above, the projection of $\boldsymbol{x}_i$ onto X_i means finding the point in X_i nearest to $\boldsymbol{x}_i$. It means solving a convex quadratic programming problem.

Convergence of Algorithm 1 follows from Theorem 3.9 in Oliveira and Sagastizábal (2014). It yields the following theoretical estimate: to obtain $\Delta < \epsilon$, it suffices to perform $c(V/\epsilon)^2$ iterations, where the constants c and V depend on parameter settings, and problem characteristics, respectively.

Remark 2. Concerning the practical efficiency of the (exact) level method of Lemaréchal et al. (1995), in Nemirovski (2005) (Chapter 5.3.2) observes the following experimental fact. When solving a problem of dimension n with accuracy ϵ, the level method makes no more than $n \ln(V/\epsilon)$ iterations, where V is a problem-dependent constant.

This observation was confirmed by the experiments reported in Fábián and Szőke (2007) and Zverovich, Fábián, Ellison, and Mitra (2012), where the level method was applied in decomposition schemes for the solution of two-stage stochastic programming problems.

Following Lemaréchal et al. (1995), we define *critical iterations* for Algorithm 1. Let us consider a maximal sequence of iterations such that $\Delta_1 \geqslant \Delta_2 \geqslant \cdots \geqslant \Delta_s \geqslant (1-\lambda)\Delta_1$ holds. Maximality of this sequence means that $(1-\lambda)\Delta_1 > \Delta_{s+1}$. Then $\boldsymbol{x}_s \to \boldsymbol{x}_{s+1}$ will be labeled critical. The above construction is repeated starting from the index s. Thus the iterations are grouped into sequences, and the sequences are separated with critical iterations.

There is an analogy between the critical iterations of level-type methods, and the serious steps of traditional bundle methods. In this paper we use the former terminology which we feel more precise in the present setting.

2. Two-stage stochastic programming models and methods

First we present the notation with a brief overview of the models. The first-stage decision is represented by the vector $\boldsymbol{x} \in X$, the feasible domain being defined by a set of linear inequalities. We assume that the feasible domain is a non-empty convex bounded polyhedron, and that there are S possible outcomes (*scenarios*) of the random event, the s th outcome occurring with probability p_s.

Suppose the first-stage decision has been made with the result $\boldsymbol{x}$, and the s th scenario has realized. The second-stage decision $\boldsymbol{y}$ is computed by solving the *second-stage problem* or *recourse problem* that we denote by $\mathcal{R}_s(\boldsymbol{x})$. This is a linear programming problem whose dual is $\mathcal{D}_s(\boldsymbol{x})$:

$$\mathcal{R}_s(\boldsymbol{x}) \quad \begin{array}{l} \min\ \boldsymbol{q}_s^T \boldsymbol{y} \\ \text{such that} \\ T_s\boldsymbol{x} + W_s\boldsymbol{y} = \boldsymbol{h}_s, \\ \boldsymbol{y} \geqslant \boldsymbol{0}, \end{array} \qquad \Bigg| \qquad \mathcal{D}_s(\boldsymbol{x}) \quad \begin{array}{l} \max\ \boldsymbol{z}^T(\boldsymbol{h}_s - T_s\boldsymbol{x}) \\ \text{such that} \\ W_s^T\boldsymbol{z} \leqslant \boldsymbol{q}_s, \\ \boldsymbol{z} \text{ is a real} - \text{valued vector.} \end{array} \tag{1}$$

In the above formulae, $\boldsymbol{q}_s$, $\boldsymbol{h}_s$ are given vectors and T_s, W_s are given matrices, with compatible sizes. We assume that $\mathcal{R}_s(\boldsymbol{x})$ is feasible for any $\boldsymbol{x} \in X$ and $s = 1, \ldots, S$. Moreover we assume that $\mathcal{D}_s(\boldsymbol{x})$ is feasible for any $s = 1, \ldots, S$. Let $q_s(\boldsymbol{x})$ denote the common optimum. This is a polyhedral convex function called the *recourse function*.

The customary formulation of the *first-stage problem* is

$$\min\ \boldsymbol{c}^T\boldsymbol{x} + \sum_{s=1}^{S} p_s\ q_s(\boldsymbol{x}) \quad \text{such that } \boldsymbol{x} \in X. \tag{2}$$

The expectation part of the objective, $q(\boldsymbol{x}) = \sum_{s=1}^{S} p_s\ q_s(\boldsymbol{x})$, is called the *expected recourse function*.

Since the two-stage stochastic programming problem (2)– (1) features discrete finite distributions and linear functions, it can be formulated as a single linear programming problem that we call the *equivalent linear programming problem*.

Given a finite subset $\widetilde{U}_s$ of the feasible domain of $\mathcal{D}_s(\boldsymbol{x})$, the function

$$\tilde{q}_s(\boldsymbol{x}) := \max_{\boldsymbol{u_s} \in \widetilde{U}_s} \boldsymbol{u_s^T}(\boldsymbol{h_s} - T_s\boldsymbol{x}) \quad (\boldsymbol{x} \in X) \tag{3}$$