Ann Oper Res (2006) 142: 259–268
DOI 10.1007/s10479-006-6171-1

A splitting method for stochastic programs

Teemu Pennanen · Markku Kallio

Abstract This paper derives a new splitting-based decomposition algorithm for convex stochastic programs. It combines certain attractive features of the progressive hedging algorithm of Rockafellar and Wets, the dynamic splitting algorithm of Salinger and Rockafellar and an algorithm of Korf. We give two derivations of our algorithm. The first one is very simple, and the second one yields a preconditioner that resulted in a considerable speed-up in our numerical tests.

1. Introduction

Consider a stochastic program over a scenario tree of a finite set I of nodes. Except for the initial node, which is denoted by 0, every node $i \in I$ has a unique predecessor node that will be denoted by i_-. The probability of reaching node i will be denoted by π_i, and it is assumed strictly positive. Every node $i \in I$ has an associated decision variable $x_i \in \mathbb{R}^{n_i}$, and the problem is to

$$\begin{aligned} &\text{minimize} && \sum_{i \in I} \pi_i f_i(x_i), \\ &\text{subject to} && A_i x_i + B_i x_{i_-} = b_i \quad \forall i \in I \setminus \{0\}, \end{aligned} \tag{SP}$$

where f_i are closed convex functions taking values in $(-\infty, +\infty]$, $A_i \in \mathbb{R}^{m_i \times n_i}$, $B_i \in \mathbb{R}^{m_i \times n_{i_-}}$, and $b_i \in \mathbb{R}^{m_i}$. The value $f_i(x_i)$ can be interpreted as the penalty that has to be

The first author was supported by the Finnish Foundation for Economic Education under grants 20728 and 21599, and by Jenny ja Antti Wihuri Foundation.

T. Pennanen (✉) · M. Kallio
Department of Business Technology, Helsinki School of Economics, PL 1210, 00101 Helsinki, Finland
e-mail: teemu.pennanen@hse.fi

paid if action x_i is taken in node i. We emphasize that by allowing infinite penalties, any constraints of the form

$$x_i \in X_i,$$

where X_i is a closed convex set, can be incorporated into f_i.

Stochastic programs are typically very large due to the large number of scenarios needed to represent the underlying uncertainty accurately enough. Several decomposition algorithms have been developed to take advantage of the special structure of the problem; see the survey articles of Birge (1997); Ruszczyński (1997). The purpose of this paper is to derive yet another decomposition algorithm for (SP), using operator splitting methods. The algorithm achieves a decomposition over the nodes of the scenario tree, and the subproblems are coordinated by solving simplified stochastic programs where the objective is replaced by a quadratic function. These "master problems" can be solved with standard methods of linear algebra, and it turns out that only one matrix factorization is needed in solving all of them in the course of the algorithm. The algorithm is closely related to Algorithm 17 in Pennanen (2002), but the approach we take here is simpler and applies to a more general problem format. Another contribution of this paper is a new scaling strategy that is suggested by well known properties of the *proximal point algorithm* in Rockafellar (1976) from which our decomposition algorithm is derived.

Our algorithm has close connections with the *progressive hedging* algorithm of Rockafellar and Wets (1991), the *dynamic splitting* algorithm of Salinger (1997); Salinger and Rockafellar, and the decomposition algorithm of Korf (1998). Like progressive hedging, our algorithm can be easily derived from the Douglas-Rachford operator splitting algorithm. The main difference comes from the fact that progressive hedging is based on scenario-wise formulation with nonanticipativity constraints, which leads to decomposition over the scenarios, and to a very simple master problem. Both the dynamic splitting and the algorithm of Korf (1998) were derived by applying a splitting algorithm to a primal-dual formulation of the original problem. Both algorithms decompose the original problem into node-wise subproblems and a linear coordination problem like our algorithm. The node-wise subproblems are variational inequalities that in Korf (1998) were solved directly by a pivoting algorithm, but in Salinger (1997) dualization was used to reformulate them as minimization problems. The objectives in the resulting minimization problems involved certain conjugate functions whose evaluation is a nontrivial task in general. The algorithm derived in this paper combines the simplicity of the progressive hedging with the node-wise decomposition of dynamic splitting and the algorithm of Korf (1998).

In the next section, we derive the decomposition algorithm from the Douglas-Rachford splitting algorithm. In Section 3, we give an alternative, a bit more complicated derivation based on the *method of partial inverses* of Spingarn (1983), and obtain a way to scale the problem. In the last section, we present the results of our first computational experiments.

2. Douglas-Rachford splitting—a decomposition algorithm

Consider the product space $X = \prod_{i\in I} \mathbb{R}^{n_i}$ equipped with the inner product

$$\langle x, y\rangle_\Pi = \sum_{i\in I} \pi_i x_i \cdot y_i$$

and the corresponding norm $\|x\|_\Pi = (\sum_{i\in I} \pi_i \|x\|^2)^{\frac{1}{2}}$. Let $C \subset X$ denote the affine set described by the dynamic constraints in (SP). Defining the function f on X by $f(x) = \sum_{i\in I} \pi_i f_i(x_i)$, problem (SP) can be written as

$$\text{minimize} \quad f(x) + \delta_C(x),$$

where δ_C is the indicator function of C. By the sum-rule of subdifferentiation, we have $\partial(f + \delta_C) \supset \partial f + \partial\delta_C$, where equality holds if $0 \in \text{ri}(\text{dom } f - C)$ (Rockafellar, 1970, Theorem 23.8). Thus, the inclusion

$$T_1(x) + T_2(x) \ni 0, \tag{1}$$

where $T_1 = \partial f$ and $T_2 = \partial\delta_C$, is a sufficient condition for optimality in (SP), and under the constraint qualification $0 \in \text{ri}(\text{dom } f - C)$ it is also necessary.

Several operator splitting algorithms have been developed for problems of the form (1); see for example Eckstein (1989); Eckstein and Ferris (1998). In particular, we can use the Douglas-Rachford splitting algorithm which can be written as follows.

Douglas-Rachford splitting

1. Compute

$$z^{k+1} = (I + T_1)^{-1}(x^k + y^k);$$

2. Compute

$$x^{k+1} = (I + T_2)^{-1}(z^{k+1} - y^k),$$

and set $y^{k+1} = y^k - z^{k+1} + x^{k+1}$, $k := k + 1$ and go to step 1

In our case, step 1 means that z^{k+1} solves the inclusion

$$\partial f(z) + z \ni x^k + y^k,$$

or equivalently, minimizes the function

$$f(z) + \frac{1}{2}\|z - (x^k + y^k)\|_\Pi^2.$$

Similarly, step 2 means that x^{k+1} minimizes

$$\delta_C(x) + \frac{1}{2}\|x - (z^{k+1} - y^k)\|_\Pi^2,$$

or in other words, x^{k+1} is the projection of $z^{k+1} - y^k$ on the feasible set C. It follows that if $x^0 \in C$ and $y^0 \in C^\perp$, then $x^k \in C$ and $y^k \in C^\perp$ for all k, so in particular, $(I + T_2)^{-1}(z^{k+1} - y^k) = (I + T_2)^{-1}(z^{k+1})$. Using the definitions of f, C and $\|\cdot\|_\Pi$, we can write the Douglas-Rachford splitting algorithm for (1) as

Algorithm 1.

```
0. Choose $x^0 \in C$ and $y^0 \in C^{\perp}$;
1. For each $i \in I$
   minimize $f_i(z_i) + \frac{1}{2}\left\|z_i - \left(x_i^k + y_i^k\right)\right\|^2,$
      for $z_i^{k+1}$;
2.
   minimize $\sum_{i\in I} \pi_i \frac{1}{2}\left\|x_i - z_i^{k+1}\right\|^2$
   subject to $A_i x_i + B_i x_{i_-} = b_i \quad \forall i \in I \setminus \{0\}$
      for $x^{k+1}$, set
   $y^{k+1} = y^k - z^{k+1} + x^{k+1},$
      $k = k + 1$ and go to step 1.
```

Note that the minimization in step 1 has been split into a separate subproblem for each node of the scenario tree. The additional quadratic term guarantees that each one has a unique solution. All the dynamic constraints of the original problem are dealt with in step 2, where one is asked to solve a stochastic program which is obtained from the original one by replacing the nonlinear extended real-valued objective by a quadratic one. Such problems can be solved with well-known methods of linear algebra or by recursion as in Salinger (1997), Salinger and Rockafellar, Blomvall and Lindberg (2002), Steinbach (2001). Viewing step 2 as a projection, suggests the use of matrix factorization techniques Golub and Van Loan (1996). Since only the point to be projected changes from iteration to iteration, one can use the same factorization at every iteration.

The general convergence properties of the Douglas-Rachford algorithm yield the following; (see Eckstein and Ferris, 1998, Proposition 6).

Theorem 1. If f_i are closed proper convex functions, then $\{x^k\}$ converges to a solution of (SP), provided one exists.

Note that (Eckstein and Ferris, 1998, Proposition 6) allows inexact computations as well as over/under-relaxed updates. For simplicity of presentation, we will not consider these extensions explicitly.

It is interesting to compare Algorithm 1 with the progressive hedging algorithm of Rockafellar and Wets (1991) (see also Robinson 1991; Mulvey and Vladimirou 1991), which can be derived by applying the Douglas-Rachford algorithm to the "split-variable" (scenario-wise) formulation of (SP), much like we have done above; see Ruszczyński (1997). Progressive hedging algorithm also consist of a sequence of simple minimization problems and projection steps. In progressive hedging, the decomposition is achieved over the scenarios of the decision tree, as opposed to nodes as in Algorithm 1. On the other hand, in progressive hedging, the projection step comes down to computing conditional expectations, which is very easy to implement in the case of a finite scenario tree.

From the implementation point of view, Algorithm 1 is closer to the dynamic splitting algorithm developed in Salinger (1997), Salinger and Rockafellar. There, as in Algorithm 1, the decomposition is achieved over the nodes, and the projection step involves the solution of a stochastic program with a quadratic objective. The main difference is that, in Algorithm 1, the objectives of the subproblems in step 1 can be directly obtained from the original problem, whereas in dynamic splitting, one has to compute certain conjugate functions, which can be difficult. Also, in the projection step of dynamic splitting, the quadratic term changes between the iterations, whereas in Algorithm 1, the coefficient matrix remains fixed throughout the algorithm.

In Korf (1998), Korf presented an algorithm that also decomposes a stochastic program over the nodes, but there, also the coordination step was decomposable, and instead of minimization problems, the node-wise subproblems were in the form of variational inequalities.

3. Method of partial inverses – scaling

In our first computational experiments we found that if Algorithm 1 is implemented as such, it converges slowly; see Section 4. It is well known that splitting methods are very sensitive to scaling of the problem, that is, on the choice of the inner product on the underlying space. The purpose of this section is to present an alternative derivation of Algorithm 1 that suggests a particular way to scale the problem.

Write the feasible set of (SP) as $C = S + a$, where S is a subspace and a is a feasible point. Setting $\tilde{x} = x - a$ and $\tilde{f}(x) = f(x + a)$, (SP) can be written as

$$\text{minimize} \quad \tilde{f}(\tilde{x}) + \delta_S(\tilde{x}).$$

Just as in Section 2, we obtain the optimality condition $\partial \tilde{f}(x) + \partial\delta_S(x) \ni 0$. Using the fact that S is a subspace, it is easily verified that

$$\partial\delta_S(x) = \begin{cases} S^\perp & \text{if } x \in S, \\ \emptyset & \text{otherwise,} \end{cases}$$

where $S^\perp$ is the orthogonal complement of S. The optimality condition can thus be written as

$$x \in S, \quad y \in S^\perp, \quad y \in T(x), \tag{2}$$

where $T = \partial \tilde{f}$.

Problem (2) is exactly in the form to which Spingarn's *method of partial inverses* (MPI) can be applied Spingarn (1983). Spingarn showed that (2) is equivalent to

$$T_S(z) \ni 0, \tag{3}$$

where T_S is a monotone mapping called the *partial inverse* of T, and that the proximal point algorithm applied to (3) can be written in the following form.

Method of partial inverses

0. Choose $x^0 \in S$ and $y^0 \in S^\perp$;
1. Find $z^{k+1}, w^{k+1} \in H$ such that

$$x^k + y^k = z^{k+1} + w^{k+1}$$
$$\frac{1}{c} P_S w^{k+1} + P_{S^\perp} w^{k+1} \in T\left(P_S z^{k+1} + \frac{1}{c} P_{S^\perp} z^{k+1} \right);$$

2. Let $x^{k+1} = P_S z^{k+1}$ and $y^{k+1} = P_{S^\perp} w^{k+1}$, $k = k+1$ and go to 1,

where c is the (in this case, constant) step-size parameter of the proximal point algorithm, and P_D denotes the projection (with respect to the inner product $\langle \cdot, \cdot \rangle_\Pi$) onto a set D.

If $c = 1$, step 1 can be written as $z^{k+1} = (I + T)^{-1}(x^k + y^k)$, and MPI becomes a special case of the Douglas-Rachford algorithm, as noted by Eckstein (1989); see also Eckstein and Bertsekas (1992). This is how MPI is usually applied Spingarn (1983); Robinson (1991). It is well-known, however, that the proximal point algorithm tends to converge faster if the step-size parameter is kept high. We will now proceed with an arbitrary value of c.

Defining $M_c = P_S + c^{-1} P_{S^\perp}$, the inclusion in step 1 can be written as

$$\frac{1}{c} M_c^{-1} w^{k+1} \in T\big(M_c z^{k+1}\big) \iff \frac{1}{c} w^{k+1} \in (M_c T M_c)(z^{k+1}).$$

Eliminating w^{k+1} from the equations, MPI becomes

0. Choose $x^0 \in S$ and $y^0 \in S^\perp$;
1. Solve

   $$(M_c T M_c)(z) + \frac{1}{c}(z - x^k - y^k) \ni 0$$

   for z^{k+1};
2. Set

   $$x^{k+1} = P_S z^{k+1},$$
   $$y^{k+1} = y^k - P_{S^\perp} z^{k+1},$$

   $k = k+1$, and go to 1.

Note that since $P_{S^\perp} = I - P_S$, the updating formula for y^k can be written as $y^{k+1} = y^k - z^{k+1} + x^{k+1}$. Since M_c is symmetric, we have by (Rockafellar, 1970, Theorem 23.9) that $M_c T M_c = \partial(\tilde{f} \circ M_c)$, so step 1 means that z^{k+1} minimizes the function

$$f(M_c z + a) + \frac{1}{2c} \|z - (x^k + y^k)\|_\Pi^2.$$

Making the change of variables

$$z := M_c z + a, \quad x := M_c x + a, \quad y := M_c y,$$

we get the following algorithm.

0. Choose $x^0 \in C$ and $y^0 \in C^\perp$;
1.

minimize $\quad f(z) + \dfrac{1}{2c} \left\| M_c^{-1}(z - (x^k + y^k)) \right\|_\Pi^2$

for z_{k+1};

2.

$$\text{minimize} \quad \frac{1}{2}\left\| M_c^{-1}(x - z^{k+1}) \right\|_\Pi^2 \quad \text{subject to} \quad x \in C$$

for x^{k+1}, and set

$$y^{k+1} = y^k - z^{k+1} + x^{k+1}$$

$k = k + 1$, and go to 1,

where C is the feasible set of (SP). We see that this is just the Douglas-Rachford splitting algorithm applied to (1) under the new inner product

$$\langle x, y \rangle_{H_c} = \langle H_c x, y \rangle ,$$

where $H_c = M_c^{-1} \Pi M_c^{-1}$, and $M_c^{-1} = (P_S + c^{-1} P_{S^\perp})^{-1} = P_S + c P_{S^\perp}$. Since P_S and $P_{S^\perp}$ are projections with respect to the inner product $\langle \cdot, \cdot \rangle_\Pi$, they commute with Π, so that

$$H_c = \Pi M_c^{-2} = \Pi\big(P_S + c^2 P_{S^\perp}\big). \tag{4}$$

Noticing that $P_{S^\perp} = I - P_S$, where S is the kernel of the constraint matrix in (SP), the computation of H_c comes down to forming the projection matrix for S; see Golub and Van Loan (1996).

Note that if $c = 1$, then $M_c^{-2} = I$ and we recover Algorithm 1. The fact that c is the step-size parameter of the proximal point algorithm, suggests choosing $c > 1$. The problem with this, however, is that the corresponding inner product need not be separable, and then we cannot decompose the minimization in step 1 over the nodes as in Algorithm 1. A natural idea then is to approximate the matrix H_c by a matrix D such that

$$\langle y, Dx \rangle = \sum_{i \in I} y_i \cdot D_i x_i .$$

A very simple choice is to let D be the diagonal part of H_c. While a high value of c corresponds to a large step-size in the proximal point algorithm, the (block-)diagonal approximation of (4) becomes worse as c increases. It is thus not clear what is the best value of c, but in practice, some insights can be gained by experimentation like in the example below.

4. Numerical experiments

Algorithm 1 was implemented for a stochastic program that arises in asset liability management of a Finnish pension insurance company Ainassaari, Kallio, and Ranne (1998); see also Steinbach (2001). This problem can be written as (SP), where each f_i is a sum of exponential and quadratic terms. Six symmetric scenario trees with five stages were randomly generated by a time series model. The branching structures and the numbers of variables are shown in Table 1.

Table 1 Test problems

Problem	Branching structure	Variables
1	$2 \times 2 \times 2 \times 2 \times 2$	310
2	$4 \times 4 \times 2 \times 2 \times 2$	1170
3	$8 \times 2 \times 2 \times 2 \times 2$	2330
4	$2 \times 4 \times 2 \times 2 \times 2$	4650
5	$16 \times 4 \times 2 \times 2 \times 2$	8490
6	$16 \times 4 \times 4 \times 2 \times 2$	16810

We tried Algorithm 1 with and without the scaling strategy presented in the previous section. In all cases, we used the starting points $x^0 = P_C 0$ and $y^0 = 0$, and the progress of the algorithm was measured by

$$e_k = \|z^{k+1} - x^k\|_\infty,$$

which is motivated by the observation that if $e_k = 0$, then x^k is optimal. Indeed, since z^{k+1} satisfies

$$\partial f(z^{k+1}) \ni y^k - (z^{k+1} - x^k),$$

where $x^k \in C$ and $y^k \in C^\perp$, we have that if $e_k = 0$, then z^{k+1} satisfies the optimality condition (1). The implementation was done in FORTRAN 77 programming language, and the subproblems in step 1 were solved with MINOS 5.4.

Figure 1 shows the development of $\log(e_k)$ for problems 1–6 as a function of k. The bundle of six curves converging just below 1 in 1000 iterations resulted when just the scaling $D = \Pi$ (i.e. Algorithm 1 without the additional scaling of Section 3) was used. Next, the diagonal scaling $D = \Pi\,\mathrm{diag}(M_c^{-2})$, suggested by (4), was employed. The parameter value of $c = 5.7$ was found by experimentation on a small set of small problems. This resulted in the six remaining curves in Figure 1. It can be seen that with the additional scaling, the

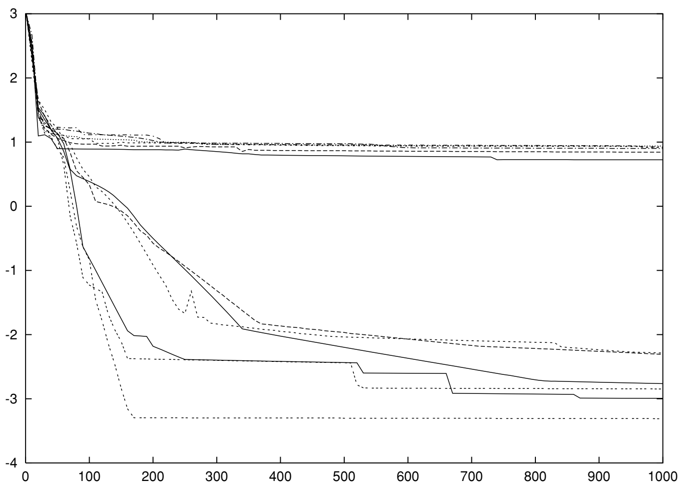

Fig. 1 $\log(e_k)$ as a function of k for problems 1–6 with and without scaling

Table 2 Number of iterations needed to reach the accuracy $e_k \leq 0.01$

Problem	Iter
1	117
2	140
3	154
4	378
5	521
6	441

size of the error reduces much faster for all problems than without the scaling. In the current implementation, the time to compute the diagonal scaling matrix took less than 5 percent of the total computation time, which seems reasonable in view of the improved convergence properties. With the error tolerance $e_k \leq 10^{-2}$ the iteration counts with scaling are shown in Table 2.

Figure 1 clearly displays a feature that was observed in all the test runs: at first, the error reduces relatively fast but after some point, the algorithm seems to stall. Similar "fat tails" have been reported in Salinger and Rockafellar. The reason for the stalling is not clear, but nevertheless, the proposed scaling seems to reduce the effect.

It is clear that more tests and a more careful implementation of Algorithm 1 would be required in order to draw definite conclusions about its practicality. It would also be interesting to study the causes of the sudden stalling that was observed in the tests.

Acknowledgements The authors would like to thank Jonathan Eckstein for suggesting several improvements to an earlier manuscript and for suggesting the use of Douglas-Rachford splitting that lead to the simple derivation of our algorithm in Section 2.

References

Ainassaari, K., M. Kallio, and A. Ranne. (1998). *Selecting an optimal investment portfolio for a pension insurance company,* in the collection of papers presented in the 8th International AFIR Collogium in Cambridge.

Birge, J.R. (1997). "Stochastic Programming Computation and Applications." *INFORMS J. Comput.* 9, 111–133.

Blomvall, J. and P.O. Lindberg. (2002). "A Riccati-Based Primal Interior Point Solver for Multistage Stochastic Programming." Interior point methods (Budapest, 2000). *European J. Oper. Res.* 143, 452–461.

Eckstein, J. (1989). "Splitting Methods for Monotone Operators with Applications to Parallel Optimization." Ph.D. Thesis, Massachusetts Institute of Technology.

Eckstein, J. and D. Bertsekas. (1992). "On the Douglas-Rachford Splitting Method and the Proximal Point Algorithm for Maximal Monotone Operators." *Math. Programming* 55 (3), Ser. A, 293–318.

Eckstein, J. and M.C. Ferris. (1998). "Operator-splitting Methods for Monotone Affine Variational Inequalities, with a Parallel Application to Optimal Control." *INFORMS J. Comput.* 10, no. 2 218–235.

Golub, G.H. and C.F. Van Loan. (1996). "Matrix Computations." 3rd edition. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, MD.

Korf, L. (1998). "Approximation and Solution Schemes for Stochastic Dynamic Optimization Problems." Doctoral Dissertation, University of California Davis.

Mulvey, J.M. and H. Vladimirou. (1991). "Applying the Progressive Hedging Algorithm to Stochastic Generalized Networks." Stochastic programming, *Ann. Oper. Res.,* 31, 399–424.

Pennanen, T. (2002). "A Splitting Method for Composite Mappings." *Numerical Functional Analysis and Optimization* 23, 875–890.

Robinson, S.M. (1991). "Extended Scenario Analysis." *Annals of Oper. Res.* 31, 385–398.

Rockafellar, R.T. (1970). "Convex Analysis." Princeton University Press.

Rockafellar, R.T. (1976). “Monotone Operators and the Proximal Point Algorithm.” *SIAM J. Control Optim.* 14, 877–898.

Rockafellar, R.T. and R.J-B. Wets. (1991). “Scenarios and Policy Aggregation in Optimization Under Uncertainty.” *Math. Oper. Res.* 16(1), 119–147.

Ruszczyński, A. (1997.) “Decomposition Methods in Stochastic Programming. Lectures on Mathematical Programming.” *Math. Programming* 79 (1-3), Ser. B, 333–353.

Salinger, D. (1997). “A Splitting Algorithm for Multistage Sochastic Programming with Application to Hydropower Scheduling.” Ph.D. Thesis, Dept. of Appied Math., University of Washington.

Salinger, D.H. and R.T. Rockafellar. “Dynamic Splitting: An Algorithm for Deterministic and Stochastic Multiperiod Optimization.” preprint.

Spingarn, J.E. (1983). “Partial Inverse of a Monotone Operator.” *Appl. Math. Optim.* 10, 247–265.

Steinbach, M.C. (2001). “Hierarchical Sparsity in Multistage Stochastic Programs.” *Stochastic Optimization: Algorithms and Applications* (Gainesville, FL, 2000), pp. 385–410, Appl. Optim., 54, Kluwer Acad. Publ., Dordrecht.