Report of Benders decomposition

Barbara Caldeira[a], Luis Rojo-González[a,b]

[a]Department of Statistics and Operations Research, Universitat Politècnica de Catalunya, Spain
[b]Department of Industrial Engineering, Universidad de Santiago de Chile, Chile

1. Introduction

The Benders' decomposition algorithm is a powerful technique to tackle problems with complicating variables such as stochastic problems and mixed-integer nonlinear programming problems. The main idea is to remodel the original problem into a Sub problem and a Master problem. And, following, solve the Master problem by generating Benders' optimality and feasibility cuts as prompted by the subproblems that are obtained by fixing the complicating variables.[1]

This project focuses on Benders' decomposition techniques for solving a two-stage stochastic problem. The problem is to determine the amount of money deposited in an ATM by a bank branch. First, the methodology is discussed, and it is further implemented for the proposed problem in software AMPL. Numerical data is used to test the method for the problem. We then compare the results obtained by applying the Benders' decomposition method with the ones obtained by directly solving as a linear optimization problem.

2. The ATM problem

The ATM problem is a two-stage stochastic program where a bank branch wants to determine the amount of money to be deposited in an ATM on Fridays, before the weekend. The branch estimates that money has a cost (associated with the loss of benefit by interest rates) of c € for each € in the ATM. The demand for money during the weekend is a discrete random variable ξ, taking s values ξ_i with probabilities $p_i, i = 1, ..s$. The ATM has a capacity of u € a cost of q € for each € the demand exceeds x.

3. Mathematical model and Benders' decomposition

The bank branch formulates the following stochastic optimization problem in the extensive form:

$$\begin{aligned} min \quad & cx + \sum_{i=1}^{s} p_i q y_i & & (1)\\ st. \quad & l \leq x \leq u & & (2)\\ & x + y_i \geq \xi_i & \forall i = \{1, \ldots, s\} & \quad (3)\\ & y_i \geq 0 & \forall i = \{1, \ldots, s\} & \quad (4) \end{aligned}$$

In order to apply the Benders' decomposition for this problem, we separate it into two problems. One is related to the first stage variable x, and the other corresponds to the second stage decision y_i. Let π_i be the dual variables associated to y_i; thus, we can write the dual form of the above problem such as follows

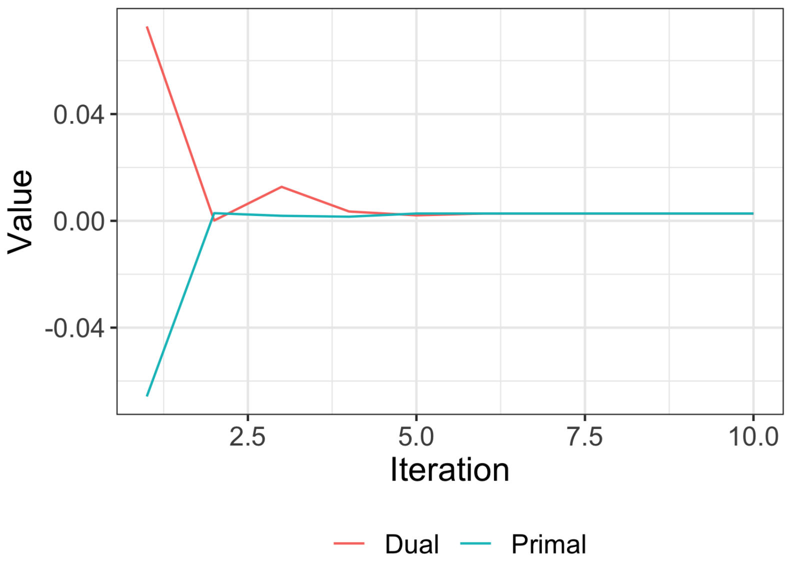
3.1. Sub-problem

$$\begin{aligned} Q_D := \quad max \quad & \sum_{i=1}^{s} \pi_i (\xi_i - x) & & (5)\\ st. \quad & \pi_i \leq p_i q & \forall i = \{1, \ldots, s\} & \quad (6)\\ & \pi_i \geq 0 & \forall i = \{1, \ldots, s\} & \quad (7) \end{aligned}$$

Email addresses: barbaracaldeira@hotmail.com (Barbara Caldeira), luis.rojo.g@usach.cl (Luis Rojo-González)

[1] You can find the code in https://drive.google.com/drive/folders/18Fe8YS5dQ94N1IXfZxQD4iXRA4u6i5VT?usp=sharing

Figure 1: Master problem and Sub-problem evolution throughout Bender's decomposition iterations.



As we can see in the mathematical model, we can derive a feasibility condition the model must accomplish is related to satisfy $(\xi_i - x) \leq 0$.

3.2. Master problem

Once we have the dual formulation, we are able to formulate the Master problem considering a subset of constraints such that $r \leq s$ and an auxiliary variable that bound our solution $z \in \mathbb{R}$. Then, the master problem is

$$B_{p_r} := \quad min \quad z \tag{8}$$

$$st. \quad z \geq cx + \sum_{i=1}^{s} \pi_i^r (\xi_i - x) \qquad \forall r \leq s \tag{9}$$

$$\sum_{i=1}^{s} \theta_i^r (\xi_i - x) \leq 0 \qquad \forall r \leq s \tag{10}$$

$$l \leq x \leq u \tag{11}$$

where θ_i^r represents the rays for the solution found at solving Q_D.

4. Computational experiments

Proposed instance to solve using Bender's decomposition algorithm described above considers $c = 0.0025$, $q = 0.0011$, $l = 21$, $u = 147$ and $s = 7$, and scenarios setting is shown in Table 1. The algorithm was implemented in AMPL software, as shown in Appendix A, and Appendix B. Figure 1 shows the obtained results for the evolution of objective functions for both Dual and Primal forms through *cuts*. Table 2 shows the value for variable x, which we are looking for, with the corresponding value of the objective function. In this, we can see that at 4th cut could get the optimal solution.

Table 1: Instance.

i	1	2	3	4	5	6	7
p_i	0.04	0.09	0.10	0.21	0.27	0.23	0.06
ξ_i	150	120	110	100	80	60	50

Table 2: Evolution of variable through iterations.

	1	2	3	4	5	6	7	8	9	10
x	147.00	84.58	107.00	114.74	110.00	110.00	110.00	110.00	110.00	110.00
cost	0.0000	-0.0290	0.0240	0.0286	0.0302	0.0303	0.0303	0.0303	0.0303	0.03025

5. Conclusion

As we can see in Table 2, the procedure converges in 5 iterations. The optimal value for x is 110 €; in other words, the bank should decide to deposit 110 €in the ATM on Friday to get the best possible profit.

In order to check the solution applying the benders' decomposition, we compared the results obtained by directly solving the extensive form as a linear optimization problem. The problem was solved in AMPL as in Appendix C. The optimal solution of the direct linear problem is 0.0305, with the optimal value of $x = 110$, the same results of applying benders' decomposition. That confirms that the benders' decomposition algorithm was able to find the optimal solution for the ATM problem.

Appendix A. AMPL code: ATM.mod

```
# ----------------------------------------
# ATM PROBLEM
# USING BENDERS DECOMPOSITION
# (using dual formulation of subproblem)
# ----------------------------------------

### SUBPROBLEM ###
set CASE := {1 .. 7}; # scenario

param p {CASE} > 0; # probability of scenario
param chi {CASE} > 0; # random number from random variable
param x_dual > 0; # auxiliar variable in second stage problem
param q_fix > 0; # fixed cost for money reposition

param BIG := 1.0e+9;          # objective > BIG ==> unbounded

var pi {CASE} >= 0; # dual variables

maximize dual_cost: sum {i in CASE} pi[i]*(chi[i] - x_dual);

subject to bound {i in CASE}: pi[i] <= p[i]*q_fix; # constraint of subproblem

### MASTER PROBLEM ###

param nCUT >= 0 integer;
param cut_type {1..nCUT} symbolic within {"point","ray"};
param c > 0; # cost of money

param l > 0;
param u > 0;

param Pi {i in CASE, 1..nCUT}; # dual variable in the MASTER

var x >= 0; # money to get backs
var z; # auxiliar variable

minimize cost: c*x + z;
   # sum {i in ORIG} fix_cost[i] * Build[i] + Max_Ship_Cost;

subject to Cut_Defn {k in 1..nCUT}:
   if cut_type[k] = "point" then z >= sum {i in CASE} Pi[i, k]*(chi[i] - x);

subject to r1: x >= l;
subject to r2: x <= u;
```

Appendix B. AMPL code: ATM.run

```
# ----------------------------------------
# BENDERS DECOMPOSITION FOR
# ATM PROBLEM
# ----------------------------------------

reset;

model ATM.mod;
data ATM.dat;

# option solver osl;
# option osl_options
#   'bbdisplay 2 bbdispfreq 100 dspace 500000 pretype 2 simplex 1';

option solver cplex; #cplexamp
option cplex_options 'mipdisplay 2 mipinterval 100 primal';

option omit_zero_rows 1;
option display_eps .000001;

# we must define the variables and equations only
problem Master: x, z, cost, Cut_Defn, r1, r2;
problem Sub: pi, dual_cost, bound;

suffix unbdd OUT;

let nCUT := 0;
let z := 0;
let x_dual := 1; # a feasible value for variable

param GAP default Infinity;

repeat { printf "\nITERATION %d\n\n", nCUT+1;

   solve Sub; # we solve de subproblem (dual form)
   printf "\n";

   # if the subproblem is unbounded we have to add a feasibility cut
   # to the master problem (remember that if the dual is unbounded
   # the primal is infeasible)
   if Sub.result = "unbounded" then { printf "UNBOUNDED\n";
      let nCUT := nCUT + 1;
      # implies a feasibility cut
      let cut_type[nCUT] := "ray";
      # we transform the Pi variable to the ray that
      # makes infeasible the dual variable
      let {i in CASE} Pi[i, nCUT] := pi[i].unbdd;
      }

   # otherwise (i.e. we have a bounded problem) we add an optimality cut
   else {
      # stop criteria
      if dual_cost == cost or nCUT = 10 then break; # + 0.00001

      # Updating the GAP
```

```
      let GAP := min (GAP, dual_cost - z);
      option display_1col 0;
      display GAP, cost;
      let nCUT := nCUT + 1;

      # implies a optimality cut
      let cut_type[nCUT] := "point";
      # we add the (dual) point at which the subproblem has a unbounded
      # solution
      let {i in CASE} Pi[i, nCUT] := pi[i];
      }

   printf "\nRE-SOLVING MASTER PROBLEM\n\n";

   # once we have the kind of cut added, we solve master problem
   solve Master;
   printf "\n";
   option display_1col 20;
   display x, z;

   display nCUT > solucion.txt;
   display cost, dual_cost, x, pi > solucion.txt;

   # We have to update the dual variable as the solution of this
   # obtained in the master problem
   let x_dual := x;
};
```

Appendix C. AMPL code: ATMnomal.mod

```
# ----------------------------------------
# ATM PROBLEM
# ----------------------------------------

### SUBPROBLEM ###
set CASE := {1 .. 7}; # scenario

param p {CASE} > 0; # probability of scenario
param chi {CASE} > 0; # random number from random variable
param q_fix > 0; # fixed cost for money reposition
param c > 0;
param l > 0;
param u > 0;

var y {CASE} >= 0;
var x >= 0;

minimize cost: c*x + sum{i in CASE} p[i]*q_fix*y[i];

subject to r1: x >= l;
subject to r2: x <= u;
subject to r3 {i in CASE}: x + y[i] >= chi[i];
```