UNIVERSIDAD POLITÉCNICA DE CATALUNYA

Benders Algorithm: Stochastic Optimization Problem

Ignasi Mañé and Joaquim Girbau

Professors: Javier Heredia and Jordi Castro

Contents

1 Introduction

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 to the loss of benefits by interest rates) of c euros for each euros in the ATM. The demand of 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 euros, with a technical minimum of l euros. If the demand is greater than x then the ATM has to be refilled, with a cost of q euros for each euros the demand exceeds x. The bank branch formulates the following stochastic optimization problem in extensive form:

$$
\begin{array}{lll}
P = \min\limits_{x} & cx + \sum_{i=1}^{s} p_i q y_i & \\
\text{subject to} & l \le x \le u & \\
& x + y_i \ge \xi_i, & i = 1, \ldots, s \\
& y_i \ge 0, & i = 1, \ldots, s
\end{array}
$$

Write and AMPL code to solve the above problem using Benders decomposition. Check the solution coincides with the one obtained by directly solving the extensive form as a linear optimization problem. You have to deliver:

- A report describing the Benders master and subproblem formulations; the AMPL code, including a short description of it; the computational results obtained. In particular, detail and justify the formulation of the Benders subproblem.
- A .zip file including the AMPL .mod, .dat and .run files.

$$
\begin{aligned}
c &= 0.00025 \\
q &= 0.0011 \\
l &= 21 \\
u &= 147 \\
s &= 7
\end{aligned}
$$

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

2 Benders Formulation

To solve the problem we will use Benders decomposition. The problem has to be split into two related problems. We start from the original extensive form of the problem written as:

$$\begin{aligned} P : \min_{x} \quad & cx + Q_P \\ \text{subject to} \quad & l \leq x \leq u \end{aligned}$$

Problem Q_P is defined as follows.

$$\begin{aligned} Q_P : \min_{y} \quad & \sum_{i=1}^{s} p_i q y_i \\ \text{subject to} \quad & y_i \geq \xi_i - x, \quad && i = 1, \ldots, s \\ & y_i \geq 0, && i = 1, \ldots, s \end{aligned}$$

The previous problem can be dualized to obtain problem Q_D in which x is considered a parameter. This problem is also referred as the sub problem.

$$\begin{aligned} Q_D : \max_{u} \quad & \sum_{i=1}^{s} (\xi_i - x) u_i \\ \text{subject to} \quad & u_i \leq p_i q_i, \quad && i = 1, \ldots, s \\ & u_i \geq 0, && i = 1, \ldots, s \end{aligned}$$

Finally, Problem BPr, also referred as the Master Problem, can be defined as follows.

$$\begin{aligned} BPr : \min_{x,z} \quad & cx + z \\ \text{subject to} \quad & z \geq \sum_{i=1}^{s} (\xi_i - x) u_i^k, \quad && k \in I \\ & 0 \geq \sum_{i=1}^{s} (\xi_i - x) v_i^k, && k \in J \end{aligned}$$

Notice that v_i^k and u_i^k are a subset of the set of extreme rays J and extreme points I of the dual problem Q_D respectively. When the set of extreme rays and extreme points is less than the whole set, this problem is a relaxation of the original problem P.

3 AMPL implementation

To solve the problem we used two different AMPL implementations. First, we directly solved the problem without decomposing it to get the optimal solution, the EVPI and the VSS. Second, the same problem was solved using Benders decomposition and the results were compared to those previously obtained.

3.1 Extensive Form of the Stochastic Problem

3.1.1 Sets and Parameters

The following sets and parameters were defined in the AMPL model.

```
param c; # cost per euro in ATM
param q; # cost per euro when demand exceeds
param l; # minimum capacity
param u; # maximum capacity
param s; # num of scenarios
param nCUT >= 0 integer; # num max of cuts

set SCENE := {1..s}; # set of scenarios

param p {SCENE}; # Probability of scenario i
param chi {SCENE}; # Demand of scenario i
```

Figure 1: Comon parameters and sets extensive form

3.1.2 Extensive Form of the Stochastic Problem

The following extensive formulation was used to get the optimal solution of the stochastic optimization problem.

```
#------------------------------
#STOCHASTIC PROGRAMMING PROBLEM
#------------------------------
#used to calculate RP
var x >= l <= u;
var y {SCENE} >= 0;

minimize P: c*x + sum{i in SCENE} (p[i]*q*y[i]);
subject to Exceed {i in SCENE}: x+y[i]>=chi[i];
```

Figure 2: Extensive form of the stochastic problem

3.1.3 Auxiliary Problem for WS

To compute the EVPI we had to calculate the value of the Wait and See solution. To do so we needed to know the optimal value of the objective function for every possible value $\xi \in \Xi$. So, we formulated a version of the previous problem considering ξ as known value or parameter. This variant of the original problem would also be useful to compute the EV solution, that is, the solution considering ξ a fixed value equal to its average.

```
#------------------------------
#DETERMINISTIC PROBLEM
#------------------------------
#useed to calculate WS
var Y >=0;
var X >= l <= u;
param Chi;

minimize DP: c*X + q*Y;
subject to Excees: X+Y>=Chi;
```

Figure 3: Deterministic problem when random variable ξ is known

To compute the value of the WS solution we had to compute $E_\xi\left[z(\bar{x}(\xi), \xi)\right]$. Therefore, the following code, which compute the expected value of the optimal solution for every given value of ξ, was implemented in the run file.

```
#-----------
#WS SOLUTION
#-----------
param WS_solution default 0;

for {i in SCENE} {

    let Chi := chi[i];
    solve WS;

    let WS_solution := WS_solution + p[i]*DP;

    }

printf "\nWS SOLUTION\n";
printf "--------------------|\n";
display WS_solution;
```

Figure 4: Iterative procedure for the WS solution

3.1.4 Auxiliary Problem for EEV

Similarly, to compute the EEV we formulated another auxiliary problem considering x as a known parameter named $Xpar$. Parameter $Xpar$ takes the value of the optimal solution of the problem defined in Figure 3 when $\xi = \bar{\xi}$.

```
#-----------------------------
#STOCHASTIC PROBLEM FOR FIXED X
#-----------------------------
#used to calculate EEV
param Xpar;

minimize EP: c*Xpar + sum{i in SCENE} (p[i]*q*y[i]);
subject to Exceedpar {i in SCENE}: Xpar+y[i]>=chi[i];
```

Figure 5: Stochastic problem with x as a parameter

3.2 Benders Implementation

3.2.1 Sets and Parameters

As in the previous section, the following sets and parameters were defined in the AMPL model.

```
param c; # cost per euro in ATM
param q; # cost per euro when demand exceeds
param l; # minimum capacity
param u; # maximum capacity
param s; # num of scenarios
param nCUT >= 0 integer; # num max of cuts

set SCENE := {1..s}; # set of scenarios

param p{SCENE}; # Probability of scenario i
param chi{SCENE}; # Demand of scenario i
```

Figure 6: omon parameters and sets of the extensive form

3.2.2 Problem Q_D. Sub-Problem

We formulated problem Q_D or Sub-Problem as follows. Notice that in this problem, x is a parameter named X and dual values u_i are the variables of the problem.

```
###### SUBPROBLEM Q_D(X) ######
param X;
var mu {SCENE} >= 0; # Dual variables

maximize SubP: sum{i in SCENE} (chi[i]-X)*mu[i];
subject to const1 {i in SCENE}: mu[i] <= p[i]*q;
```

Figure 7: Problem Q_D

3.2.3 Problem BPr. Master Problem

We formulated problem BPr as follows. Notice that in this problem x is the variable of the problem and the restrictions are feasibility and optimality cuts obtained solving the sub problem.

```
##### MASTERPROBLEM BPr(X) #####

var x >= l <=u; # euros in ATM on Friday
param MU {SCENE, 1..nCUT};
param cut_type {1..nCUT} symbolic within {"point","ray"};
# symbolic: els valors els hi posarem durant l'execucio
var z;

minimize MP: c*x + z;

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

Figure 8: Problem BPr

4 Results

In this section the results obtained using the previous implementations will be presented. Theoretically, both results must be equal since, the Benders decomposition algorithm eventually converges to the optimal solution of the problem.

4.1 Stochastic Solution

Using the first approach the results summarized in Table 1 were obtained. The value of the objective function at the optimal solution $x^* = 110$ is 0.03025. This problem has an advantage of 0.0030252 respect to the solution obtained using its deterministic version and $\xi = \bar{\xi}$. So, if the cost of solving the stochastic version of the optimization problem is less than this value, then its usage is justified.

RP	WS	EEV	EVPI	VSS	x^*_{RP}	x^*_{EV}
0.03025	0.021902	0.0332752	0.008348	0.0030252	110	87.2

Table 1: Solution of the Stochastic Optimization problem

Moreover, the values of the second stage decision variables are summarized in 2. With a solution of $x^* = 110$, these variable represent the future shortage of money under each of the possible scenarios.

Iter	y_i
1	40
2	10
3	0
4	0
5	0
6	0
7	0

Table 2: Second Stage variables value

4.2 Benders Results

Applying Benders decomposition the same optimal solution was reached after 6 iterations. Table 3 summarizes the progress of the algorithm at each iteration. Notice that the value of the master problem and the sub problem tend to be closer after each iteration until they converge at the optimal solution. As expected, the same values, contained in Table 4, were obtained for the second stage variables.

Iter	GAP	Upper B.	Lower B.	x
1	Inf	0.05342	-Inf	50.00
2	0.05951	0.03688	-0.02263	147.00
3	0.00885	0.03335	0.02450	86.89
4	0.00162	0.03026	0.02864	107.00
5	0.00052	0.03076	0.03024	114.74
6	-0.00000	0.03025	0.03025	110.00

Table 3: Summary of Bender's Algorithm Results

Iter	y_i
1	40
2	10
3	0
4	0
5	0
6	0
7	0

Table 4: Benders's Second Stage variables value

The convergence pattern for this particular case is also represented in Figure 9. The blue line is the value of the objective function of the sub problem at each iteration whereas the red line represents the value of the objective function of the master problem. At iteration 6, both values coincide and, therefore, the algorithm stops.

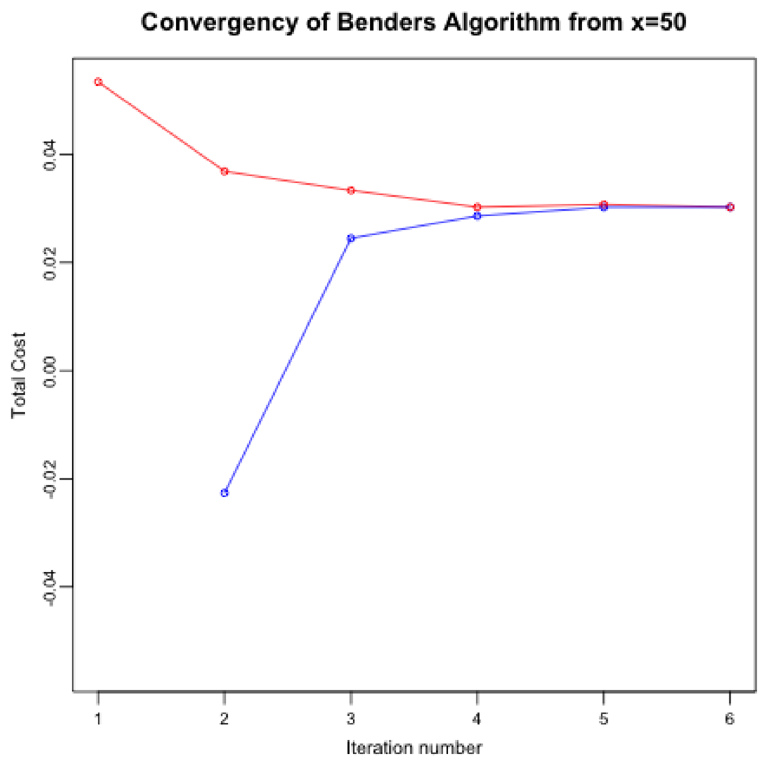

Figure 9: Convergency behaviour of the Algorithm at each iteration BPr

A AMPL Scripts

A.1 Extensive Mod Script

```
# ----------------------------------------
# THE ATM PROBLEM
# EXTENSIVE FORM
# ----------------------------------------

param c; # cost per euro in ATM
param q; # cost per euro when demand exceeds
param l; # minimum capacity
param u; # maximum capacity
param s; # num of scenarios
param nCUT >= 0 integer; # num max of cuts

set SCENE := {1..s}; # set of scenarios

param p {SCENE}; # Probability of scenario i
param chi {SCENE}; # Demand of scenario i

#-------------------------------
#STOCHASTIC PROGRAMMING PROBLEM
#-------------------------------
#used to calculate RP
var x >= l <= u;
var y {SCENE} >= 0;

minimize P: c*x + sum{i in SCENE} (p[i]*q*y[i]);
subject to Exceed {i in SCENE}: x+y[i]>=chi[i];

#-------------------------------
#STOCHASTIC PROBLEM FOR FIXED X
#-------------------------------
#used to calculate EEV
param Xpar;

minimize EP: c*Xpar + sum{i in SCENE} (p[i]*q*y[i]);
subject to Exceedpar {i in SCENE}: Xpar+y[i]>=chi[i];

#-------------------------------
#DETERMINISTIC PROBLEM
#-------------------------------
#useed to calculate WS
var Y >=0;
var X >= l <= u;
param Chi;
```

```
minimize DP: c*X + q*Y;
subject to Excees: X+Y>=Chi;

#END---------------------------------------------------
#########################################################
```

A.2 Benders Mod Script

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

param c; # cost per euro in ATM
param q; # cost per euro when demand exceeds
param l; # minimum capacity
param u; # maximum capacity
param s; # num of scenarios
param nCUT >= 0 integer; # num max of cuts

set SCENE := {1..s}; # set of scenarios

param p{SCENE}; # Probability of scenario i
param chi{SCENE}; # Demand of scenario i

###### SUBPROBLEM Q_D(X) ######
param X;
var mu {SCENE} >= 0; # Dual variables

maximize SubP: sum{i in SCENE} (chi[i]-X)*mu[i];
subject to const1 {i in SCENE}: mu[i] <= p[i]*q;

##### MASTERPROBLEM BPr(X) #####

var x >= l <=u; # euros in ATM on Friday
param MU {SCENE, 1..nCUT};
param cut_type {1..nCUT} symbolic within {"point","ray"};
# symbolic: els valors els hi posarem durant l'execucio
var z;

minimize MP: c*x + z;

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

```
#END-------------------------------------------------------
##########################################################
```

A.3 Extensive Form Run Script

```
# ----------------------------------------
# THE ATM PROBLEM
# EXTENSIVE FORM .RUN
# ----------------------------------------

reset;
# OPTIMAL SOLUTION FOR THE COMPLETE PROBLEM (using CPLEX)

model ATM.mod;
data ATM.dat;

option solver cplex;
# ----------------------------------------
# THE ATM PROBLEM .RUN
# ----------------------------------------

problem RP: P, x, y, Exceed;
problem WS: DP, X, Y,Excees;
problem ERP: EP, y, Exceedpar;

#----------------------
#SOLVING THE RP PROBLEM
#----------------------

solve RP;
option omit_zero_rows 0;

printf "\nRP SOLUTION\n";
printf "-------------------|\n";
display x;
display y;

#-----------
#WS SOLUTION
#-----------
param WS_solution default 0;

for {i in SCENE} {

let Chi := chi[i];
solve WS;
```

```
let WS_solution := WS_solution + p[i]*DP;

}

printf "\nWS SOLUTION\n";
printf "--------------------|\n";
display WS_solution;

#-----------
#EV SOLUTION
#-----------

let Chi := sum {i in SCENE} p[i]*chi[i];
solve WS;

printf "\nEV SOLUTION\n";
printf "--------------------|\n";
display X;

#------------
#EEV SOLUTION
#------------

let Xpar := X;
solve ERP;

printf "\nEEV SOLUTION\n";
printf "--------------------|\n";
display EP;
display X;

#END-----------------------------------------------------
#########################################################
```

A.4 Benders Run Script

```
# ----------------------------------------
# BENDERS DECOMPOSITION FOR
# THE ATM PROBLEM .RUN
# ----------------------------------------

reset;

model ATM_benders.mod;
data ATM.dat;

option solver cplex;
```

```
option cplex_options 'mipdisplay 2 mipinterval 100 primal';

option omit_zero_rows 0;
option display_eps .000001;

problem Master: MP, z, x, Cut_Defn;
problem Sub: mu, SubP, const1;

suffix unbdd OUT;

let nCUT := 0;
let z := -Infinity;
let X := 50; #initial solution

param GAP default Infinity;

repeat {

printf "\nITERATION %d\n", nCUT+1;
printf "-------------------------------------------------------------|\n";

printf "\nSOLVING SUB PROBLEM\n";
printf "---------------------------------------------|\n\n";

   solve Sub;
   printf "\n";

   if Sub.result = "unbounded" then { printf "UNBOUNDED\n";
      let nCUT := nCUT + 1; # we include a feasibility cut
      let cut_type[nCUT] := "ray";
      let {i in SCENE} MU[i,nCUT] := mu[i].unbdd;
      # elements of the extreme ray provided by cplex

      #Results for each iteration and starting point x=50
  printf "%3i %12.5f %12.5f %12.5f %12.5f\n",
nCUT+1,
GAP,
SubP + c*X,
MP,
X > BD_Alg_Res.txt;
  #####################################################
      }

   else {

      let GAP := min (GAP, SubP + c*X - MP);
```

```
      #Results for each iteration and starting point x=50
  printf "%3i %12.5f %12.5f %12.5f %12.5f\n",
nCUT+1,
GAP,
SubP + c*X,
MP,
X > BD_Alg_Res.txt;
  ####################################################

  if SubP <= z + 0.00001 then break;

      option display_1col 0;
      display GAP, SubP;
      let nCUT := nCUT + 1;
      let cut_type[nCUT] := "point";
      let {i in SCENE} MU[i,nCUT] := mu[i]; # Optimal extreme point
      }


printf "\nSOLVING MASTER PROBLEM\n";
printf "--------------------------------------------|\n\n";

   solve Master;
   printf "\n";
   option display_1col 20;
   display x;

   let X := x;
} until nCUT=500; #max number of iterations

printf "\nOPTIMAL SOLUTION\n";
display x;

printf "\nSECOND STGE SOLUTION\n";
display const1;

printf "\nOBJECTIVE FUNCTION\n";
display MP;

#END-----------------------------------------------------
#########################################################
```

A.5 Data

```
param c := 0.00025;
param q := 0.0011;
param l := 21;
```

```
param u := 147;
param s := 7;

param : p chi:=
1 0.04 150
2 0.09 120
3 0.10 110
4 0.21 100
5 0.27 80
6 0.23 60
7 0.06 50;
```