Implementation of Benders Decomposition for the ATM Refill Stochastic Problem

Arnau Pérez Reverte

June 1, 2025

Stochastic Optimization
MSc in Statistics and Operations Research (UPC)

Contents

1 Introduction

This report outlines the implementation of the Benders Decomposition Method to solve the ATM Refill stochastic problem using AMPL for model formulation and its bindings with Python vie AMPLpy. The document is structured as follows: Section 2 defines the stochastic optimization problem and its extensive form; Section 3 details the Benders Decomposition formulation; Section 4 explains the AMPL implementation; and Section 5 discusses the computational results and validation.

2 Stochastic Optimization Problem Formulation

The stochastic ATM cash management problem seeks to minimize the expected total cost of cash allocation and potential shortage penalties. To formulate the problem comprehensively, we consider a bank that must determine the optimal cash deposit x for an ATM before weekend operations. The problem parameters are defined as follows: let c represent the holding cost per euro deposited in the ATM, reflecting the opportunity cost of capital. The weekend demand is characterized by a discrete random variable ξ that takes values ξ_i with corresponding probabilities p_i for $i = 1, \ldots, s$ scenarios. The ATM operates under capacity constraints with lower bound l and upper bound u, representing technical and operational limitations respectively.

When demand exceeds the initial deposit x, the bank incurs a penalty cost of q per euro of shortage. Therefore, we introduce shortage variables y_i for each scenario i to capture the excess demand beyond the initial deposit. Hence, the extensive form of the stochastic optimization problem is formulated as:

$$\begin{aligned} (P) = \min \quad & cx + \sum_{i=1}^{s} p_i q y_i \\ \text{s.t.} \quad & l \le x \le u \\ & x + y_i \ge \xi_i, \quad i = 1, \ldots, s \\ & y_i \ge 0, \quad i = 1, \ldots, s \end{aligned} \tag{1}$$

This formulation represents a two-stage stochastic linear program where the first-stage decision x (cash deposit) must be made before observing the random demand ξ, while the second-stage recourse variables y_i (shortage amounts) are determined after demand realization for each scenario.

3 Benders Decomposition Approach

As established in stochastic programming theory, two-stage problems with fixed recourse can be efficiently solved using Benders Decomposition. The general approach decomposes the problem into a master problem that proposes first-stage decisions and subproblems that evaluate the optimality and feasibility of these decisions given the recourse structure.

3.1 Translating into Standard Form

As seen in class, the Benders Decomposition formulation assumes a standard form of the problem to be decomposed, which is

$$\begin{aligned} \min \quad & \mathbf{c}^T \mathbf{x} + \mathbf{q}^T \mathbf{y} \\ \text{s.t.} \quad & T\mathbf{x} + W\mathbf{y} = \mathbf{h} \\ & \mathbf{x} \in X \\ & \mathbf{y} \ge 0 \end{aligned} \tag{2}$$

where $c, x \in \mathbb{R}^{n_1}$, $q, y \in \mathbb{R}^{n_2}$, $T \in \mathbb{R}^{m \times n_1}$, $W \in \mathbb{R}^{m \times n_2}$, and $X \neq \emptyset$ is any set of constraints. The first step into translating the problem (P) into standard form is to introduce slack and surplus variables where needed in order to convert inequality contraints into equality constraints.

$$\begin{aligned} (P') = \min \quad & cx + \sum_{i=1}^{s} p_i q y_i \\ \text{s.t.} \quad & x - x^+ = l \\ & x + x^- = u \\ & x + y_i - y_i^+ = \xi_i, \quad i = 1, \ldots, s \\ & x^+, x^- \ge 0 \\ & y_i, y_i^+ \ge 0, \quad i = 1, \ldots, s \end{aligned} \tag{3}$$

Now, to convert (3) into the form assumed by Benders (2), we make the following identifications. We put

$$\mathbf{x} \coloneqq (x, x^+, x^-) \in \mathbb{R}^3$$
$$\mathbf{y} \coloneqq (y_1, y_1^+, \ldots, y_i, y_i^+, \ldots, y_s, y_s^+) \in \mathbb{R}^{2s}$$
$$X \coloneqq \left\{\mathbf{x} = (x, x^+, x^-) \in \mathbb{R}^3 : x - x^+ = l,\ x + x^- = u,\ x^+, x^- \geq 0\right\}.$$

Then, define the following

$$\mathbf{c} \coloneqq (c, 0, 0) \in \mathbb{R}^3$$
$$\mathbf{q} \coloneqq (p_1 q, 0, \ldots, p_i q, 0, \ldots, p_n q, 0) \in \mathbb{R}^{2s}$$
$$T \coloneqq \begin{pmatrix} 1 & 0 & 0 \\ \vdots & \ddots & \vdots \\ 1 & 0 & 0 \end{pmatrix} \in \mathbb{R}^{s\times 3}, \quad W \coloneqq \begin{pmatrix} 1 & -1 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 1 & -1 \end{pmatrix} \in \mathbb{R}^{s\times 2s}$$
$$\mathbf{h} \coloneqq (\xi_1, \ldots, \xi_s) \in \mathbb{R}^s.$$

It is clear that under these definitions our ATM problem is in the desired form.

3.2 Master Problem Formulation

The master problem determines the optimal cash deposit x while satisfying capacity constraints and a growing set of optimality and feasibility cuts generated from the subproblem solutions. We recall the formulation of the Master problem seen in class and we translate it into the context of our problem.

$$\begin{aligned} (BP_r) = \min \quad & z \\ \text{s.t.} \quad & z \geq \mathbf{c}^T\mathbf{x} + \mathbf{u^i}^T(\mathbf{h} - T\mathbf{x}) \quad i \in I \subseteq \{1 \ldots p'\} \\ & \mathbf{v^j}^T(\mathbf{h} - T\mathbf{x}) \leq 0 \quad j \in J \subseteq \{1 \ldots q'\} \\ & \mathbf{x} \in X \end{aligned} \quad \equiv \quad \begin{aligned} \min \quad & z \\ \text{s.t.} \quad & z \geq (c - \mathbf{u^i}^T\mathbf{e})x + \mathbf{u^i}^T\mathbf{h} \quad i \in I \subseteq \{1 \ldots p'\} \\ & \mathbf{v^j}^T\mathbf{h} - (\mathbf{v^j}^T\mathbf{e})x \leq 0 \quad j \in J \subseteq \{1 \ldots q'\} \\ & \mathbf{x} \in X \end{aligned}$$

This finally yields the Master Problem (BP_r)

$$\begin{aligned} (BP_r) = \min \quad & z \\ \text{s.t.} \quad & z \geq \left(c - \sum_{k=1}^{s} \mathbf{u^i}_k\right)x + \sum_{k=1}^{s} \mathbf{u^i}_k \xi_k \quad i \in I \subseteq \{1 \ldots p'\} \\ & \sum_{k=1}^{s} \mathbf{v^j}_k \xi_k - \left(\sum_{k=1}^{s} \mathbf{v^j}_k\right)x \leq 0 \quad j \in J \subseteq \{1 \ldots q'\} \\ & \mathbf{x} \in X. \end{aligned} \tag{4}$$

3.3 Subproblem Formulation and Dual Development

Fixing a first-stage solution $x \in X$ from the master problem, the subproblem evaluates the recourse cost as the solution to the problem

$$\begin{aligned} (Q) = \min \quad & \mathbf{q}^T\mathbf{y} \\ \text{s.t.} \quad & W\mathbf{y} = \mathbf{h} - T\mathbf{x} \\ & \mathbf{y} \geq 0 \end{aligned} \quad \equiv \quad \begin{aligned} \min \quad & \sum_{k=1}^{s} p_k q y_k \\ \text{s.t.} \quad & y_i - y_i^+ = \xi_i - x_i, \quad i = 1, \ldots, s \\ & \mathbf{y} \geq 0 \end{aligned} \tag{5}$$

To derive the Benders cuts, we formulate the dual of the previous subproblem. The dual variable $\mathbf{u}$ corresponds to the constraint $W\mathbf{y} = \mathbf{h} - T\mathbf{x}$, yielding the dual formulation:

$$\begin{aligned} (Q_D) = \max \quad & \mathbf{u}^T(\mathbf{h} - T\mathbf{x}) \\ \text{s.t.} \quad & W^T\mathbf{u} \leq \mathbf{q} \\ & \mathbf{u} \in \mathbb{R}^s \end{aligned} \quad \equiv \quad \begin{aligned} \max \quad & \sum_{k=1}^{s} \mathbf{u}_k(\xi_k - x_k) \\ \text{s.t.} \quad & \mathbf{u}_i \leq p_i q, \quad i = 1, \ldots, s \\ & \mathbf{u} \geq 0 \end{aligned} \tag{6}$$

3.4 Optimality and Feasibility Cuts Generation

If during iteration r of the Benders algorithm the dual of the subproblem finds a finite solution in a vertex $\mathbf{u}^{\mathbf{i_0}} \in \mathbb{R}^s$, we compare the current objective of the Master problem z_r^* with the one reconstructed using this finite solution via

$$|z_r^* - z^*| = \left| z_r^* - \left(c - \sum_{k=1}^{s} \mathbf{u}^{\mathbf{i_0}}{}_k \right) x + \sum_{k=1}^{s} \mathbf{u}^{\mathbf{i_0}}{}_k \xi_k \right| < \varepsilon$$

for some tolerance $\varepsilon \coloneqq 10^{-6}$. If this condition is not satisfied, we add the vertex $\mathbf{u}^{\mathbf{i_0}}$ to the set of optimality cuts.

Moreover, if the dual subproblem becomes unbounded, we extract the unbounded ray $\mathbf{v}^{\mathbf{i_0}}$ and we impose it as a feasibility cut according to the form of Equation (4).

4 AMPL Implementation

Our AMPL implementation orchestrates the Benders decomposition algorithm through the official AMPL Python API (AMPLpy), enabling dynamic problem modification and iterative solution of the master problem and subproblem evaluation. The implementation structure follows established decomposition patterns while accommodating the specific requirements of the stochastic ATM cash management problem.

4.1 Model Structure and Problem Formulation

The implementation consists of a unified AMPL model file (`atm.mod`) that defines multiple problem formulations within a single framework. This approach facilitates both extensive form solution and Benders decomposition while maintaining consistency in problem parameterization.

4.1.1 Parameter Definitions

The model defines the fundamental problem parameters as follows:

```
param s integer > 0;              # Number of scenarios
set SCENARIOS := 1..s;            # Set of scenarios
param c > 0;                      # Holding cost per unit
param p{i in SCENARIOS};          # Scenario probabilities
param xi{i in SCENARIOS};         # Demand realizations
param u > 0, l > 0;              # Capacity bounds
param q > 0;                      # Shortage penalty cost
```

For the Benders decomposition, the following additional parameters manage the dynamic cut generation:

```
param n_cut integer >= 0;         # Number of cuts generated
set CUTS := 1..n_cut;            # Set of cuts
param opt_cuts_rhs{CUTS};         # RHS of optimality cuts
param opt_cuts_coeff{CUTS, 1..3}; # Coefficients of optimality cuts
param feas_cuts_rhs{CUTS};        # RHS of feasibility cuts
param feas_cuts_coeff{CUTS, 1..3}; # Coefficients of feasibility cuts
param cuts_types{CUTS} symbolic;  # Cut type classification
```

4.1.2 Variable Structure

The model employs an extended variable formulation to accommodate capacity constraints:

```
var x{1..3} >= 0;                 # Extended first-stage variables
var z;                            # Master objective approximation
var y{SCENARIOS} >= 0;            # Second-stage shortage variables
var surplus_y{SCENARIOS} >= 0;    # Auxiliary surplus variables
var u_dual{SCENARIOS} >= 0;       # Dual variables for subproblem
```

The variable `x[1]` represents the actual cash deposit, while `x[2]` and `x[3]` serve as the auxiliary variables introduced to reformulate the capacity constraints in standard form. Note that in the case of $\mathbf{y}$, we separate into `y` and `surplus_y`.

4.2 Problem Decomposition Structure

4.2.1 Extensive Form Formulation

The extensive form problem serves as a validation benchmark and is defined as:

```
minimize Total_Cost:
    (c*x[1] + sum{i in SCENARIOS} p[i]*q*y[i]);

subj to ATM_Capacity_Upper: x[1] - x[2] = l;
subj to ATM_Capacity_Lower: x[1] + x[3] = u;
subj to ATM_Capacity_Scenarios {i in SCENARIOS}:
    y[i] - surplus_y[i] = xi[i] - x[1];
```

This formulation captures the complete stochastic problem while maintaining the auxiliary variable structure required for decomposition consistency.

4.2.2 Master Problem Formulation

The master problem determines the first-stage decisions subject to accumulated optimality and feasibility cuts:

```
minimize Master_Obj: z;

subj to ATM_Capacity_Upper: x[1] - x[2] = l;
subj to ATM_Capacity_Lower: x[1] + x[3] = u;
subj to Opt_Cuts {k in CUTS: cuts_types[k] = "point"}:
    z >= opt_cuts_rhs[k] + sum{j in 1..3} opt_cuts_coeff[k,j] * x[j];
subj to Feas_Cuts {k in CUTS: cuts_types[k] = "ray"}:
    0 >= feas_cuts_rhs[k] + sum{j in 1..3} feas_cuts_coeff[k,j] * x[j];
```

The master problem maintains the capacity constraint structure while progressively incorporating cuts generated from subproblem solutions.

4.2.3 Subproblem Formulation

Given a fixed first-stage solution $\bar{x}$, the subproblem evaluates the dual formulation to generate Benders cuts:

```
maximize Dual_Obj:
    sum{i in SCENARIOS} u_dual[i] * (xi[i] - x_fix[1]);

subj to Dual_Inequality_Constraints {i in SCENARIOS}:
    u_dual[i] <= p[i]*q;
```

This dual formulation directly provides the coefficients required for optimality cut generation while enabling detection of feasibility issues through unboundedness analysis.

4.3 AMPLpy Algorithm Implementation

The Benders decomposition iterative frameworks is orchestrated through Python using the AMPLpy interface. The implementation follows the classical Benders framework adapted for stochastic programming applications.

4.3.1 Initialization and Main Loop

The algorithm initializes with an empty cut set and iterates until convergence:

```
n_cut = 0
eps = 1e-6
max_iter = 100

ampl.getParameter("n_cut").set(n_cut)

for iteration in range(max_iter):
```

```
    # Solve master problem
    ampl.solve("Master", solver="cplex")
    master_obj = ampl.get_objective("Master_Obj").value()
    x_vals = {i: ampl.get_variable("x")[i].value() for i in range(1, 4)}
```

The AMPLpy interface facilitates direct access to solution values and objective functions, enabling seamless data transfer between master and subproblem iterations.

4.3.2 Subproblem Evaluation and Cut Generation

For each master solution, the algorithm fixes the first-stage variables and evaluates the subproblem:

```
# Fix x variables for subproblem
ampl.param['x_fix'] = x_vals

# Solve subproblem
ampl.solve("Subproblem", solver="cplex")
subproblem_status = ampl.get_value("solve_result")
```

The subproblem solution status determines the appropriate cut generation strategy. When the subproblem yields an optimal solution, an optimality cut is generated:

```
if subproblem_status == "solved":
    u_dual_values = ampl.get_variable("u_dual").get_values().to_dict()
    dual_obj = ampl.get_objective("Dual_Obj").value()

    upper_bound = c * x_vals[1] + dual_obj

    # Generate optimality cut
    rhs = sum(u_dual_values[i] * xi_values[i-1] for i in SCENARIOS)
    coeff_x1 = c - sum(u_dual_values[i] for i in SCENARIOS)

    ampl.param['cuts_types'][n_cut] = "point"
    ampl.param['opt_cuts_rhs'][n_cut] = rhs
    ampl.param['opt_cuts_coeff'][n_cut, 1] = coeff_x1
```

4.3.3 Convergence Testing and Feasibility Handling

The algorithm monitors convergence through the gap between the master objective (lower bound) and the computed upper bound:

```
if abs(master_obj - upper_bound) < eps:
    print(f"=== OPTIMAL SOLUTION FOUND ===")
    print(f"Optimal cash amount: {x_vals[1]}")
    break
```

When the subproblem becomes unbounded, indicating infeasibility of the current master solution, a feasibility cut is generated using the unbounded ray retrieved via the unbdd suffix:

```
elif "unbounded" in subproblem_status.lower():
    u_dual_ray = ampl.get_variable("u_dual").get_values("unbdd").to_dict()

    rhs = sum(u_dual_ray[i] * xi_values[i-1] for i in SCENARIOS)
    coeff_x1 = -sum(u_dual_ray[i] for i in SCENARIOS)

    ampl.param['cuts_types'][n_cut] = "ray"
    ampl.param['feas_cuts_rhs'][n_cut] = rhs
    ampl.param['feas_cuts_coeff'][n_cut, 1] = coeff_x1
```

5 Results

We conducted computational experiments using the provided problem parameters to validate our Benders decomposition implementation against the extensive form solution. The experiments were performed using AMPL with the CPLEX solver.

5.1 Problem Instance

The test instance incorporates the following values for the parameters:

- Holding cost: $c = 0.00025$ per euro
- Shortage penalty: $q = 0.0011$ per euro
- Capacity bounds: $l = 21000$, $u = 147000$ euros
- Demand random variable support:

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 (euros)	150000	120000	110000	100000	80000	60000	50000

5.2 Computational Performance

The Benders decomposition algorithm converged to optimality within 5 iterations, showing rapid convergence for this particular problem structure. The computational results are summarized as follows:

Method	**Optimal Solution (euros)**	**Objective Value (euros)**
Benders Decomposition	$x^* = 110000$	30.25
Extensive Form	$x^* = 110000$	30.25

Table 1: Comparison of solution methods for the stochastic ATM problem

6 Validation and Conclusions

Our implementation successfully demonstrates the application of Benders Decomposition to the stochastic ATM cash management problem. The numerical results confirm that both the decomposition approach and the extensive form yield identical optimal solutions, validating the correctness of our implementation.

The Benders decomposition approach offers computational advantages for larger-scale instances where the extensive form becomes computationally intractable. As we have seen, this was not the case for this problem description.