STOCHASTIC PROGRAMMING

November 30, 2023

Today:

- Algorithms for two-stage stochastic programs

ALGORITHMS FOR (LINEAR) TWO-STAGE STOCHASTIC PROGRAMS

Linear two-stage stochastic program

$$\underset{x \in \mathbb{R}^m}{\text{minimize}} \quad c^T x + \mathbb{E}_\xi [Q(x, \xi)]$$

↖ Uncertainties

$$\text{s.t.} \quad Ax = b, \; x \geq 0$$

$\xi = (q, T, W, h)$

where $Q(x, \xi) = \min_{y \in \mathbb{R}^m} q^T y$

$$\text{s.t.} \quad Tx + Wy = h, \; y \geq 0$$

Two-step model:

- Decide x (here-and-now) at time $t=0$
- Observe a realization of ξ

Recourse [

- Decide y (wait-and-see) at time $t=1$ as a function of x and the realization of ξ

↳ In general, we do not have access to the distribution of ξ and thus we cannot compute $\mathbb{E}_\xi [Q(x, \xi)]$

⇒ We approximate this expected value using scenarios (i.e. samples)

⇒ This produces a deterministic optimization problem with a lot of structure

Scenario reformulation

↳ We consider K scenarios $\{\xi_1, \dots, \xi_K\}$ where $\xi_k = (q_k, T_k, W_k, h_k)$. Every scenario comes with a probability $p_k \geq 0$ such that $\sum_{k=1}^{K} p_k = 1$.

↳ $\mathbb{E}_\xi[Q(x,\xi)] \approx \sum_{k=1}^{K} p_k Q(x, \xi_k)$

(Motivation: as $K \to \infty$, the scenario approximation should converge to $\mathbb{E}_\xi[Q(x,\xi)]$)

↳ With finitely many scenarios, we can define a wait-and-see variable y_k for every scenario k

⇒ The linear two-stage stochastic program can then be reformulated as a deterministic linear program

$$
\begin{cases}
\underset{\substack{x \in \mathbb{R}^n \\ y_1 \in \mathbb{R}^m \\ \vdots \\ y_K \in \mathbb{R}^m}}{\text{minimize}} & c^T x + \sum_{k=1}^{K} p_k q_k^T y_k \\
\text{s.t.} & Ax = b, \quad x \geq 0 \\
& T_k x + W_k y_k = h_k, \quad y_k \geq 0 \qquad \forall k = 1..K
\end{cases}
$$

⟶ Deterministic because the $\xi_k = (q_k, T_k, W_k, h_k)$ are known quantities

⟶ Can be solved by any general-purpose linear programming solver (Gurobi, COPT for commercial, HiGHS for open-source)

→ Dedicated algorithms and solvers have been proposed to take the particular structure of two-stage problems into account

Algorithm 1 : L-shaped method

↳ Motivation : The linear equality constraints in the scenario formulation can be combined into one linear system of equations

$$\begin{bmatrix} A & 0 & \cdots & 0 \\ T_1 & W_1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ T_K & 0 & \cdots & W_K \end{bmatrix} \begin{bmatrix} x \\ y_1 \\ \vdots \\ y_K \end{bmatrix} = \begin{bmatrix} b \\ h_1 \\ \vdots \\ h_K \end{bmatrix}$$

(m)

- Every y_k is "coupled" with x through linear constraints $T_k x + W_k y_k = h_k$
- But there are no constraints coupling y_i with y_j for $i \neq j$, suggesting we could compute the y_k's independently for fixed x

Idea of the L-shaped method

* Iteratively,
  1) Solve a "master problem" in x
  2) Use the value of x from 1) to solve K "subproblems" in $y_1, \dots, y_K$ independently
  3) Use the result of those subproblems to update the master problem

* Master problem / subproblems : Linear programs of smaller size than the original one

Full algorithmic description

Initialization: Set $i = 0$ (iteration index) and $x^0 \in \mathbb{R}^n$ as the solution of

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \quad c^T x \quad \text{s.t.} \quad Ax = b, \; x \geq 0$$

For $i = 0, 1, \ldots$

- For $k \in \{1, \ldots, K\}$, consider the problem

$$\underset{y_k^i \in \mathbb{R}^m}{\text{minimize}} \quad p_k q_k^T y_k^i \quad \text{s.t.} \quad W_k y_k^i = h_k - T_k x^i, \; y_k^i \geq 0$$

(x^i now a variable for that problem)

  a) If the problem has a solution, compute

$$Q_k(x^i) = \min_{y_k^i \in \mathbb{R}^m} p_k q_k^T y_k^i \quad \text{s.t.} \quad W_k y_k^i = h_k - T_k x^i, \; y_k^i \geq 0$$

  and u_k^i dual variable for the problem.

  b) If the problem does not have a solution, compute u_k^i dual variable such that

$$(u_k^i)^T (h_k - T_k x^i) \leq 0$$

- Compute x^{i+1} by solving

$$\underset{x \in \mathbb{R}^n, \; z_1 \in \mathbb{R}, \ldots, z_K \in \mathbb{R}}{\text{minimize}} \quad c^T x + \sum_{k=1}^{K} z_k \quad \text{s.t.} \quad Ax = b, \; x \geq 0$$

$$(u_k^j)^T (h_k - T_k x) \leq z_k, \quad j = 0, \ldots, i, \; k = 1, \ldots, K$$

(constraints added before iteration i; new constraints obtained by solving the subproblems at iteration i)

where the constraints added at iteration i are

$$(u_k^i)^T (T_k x^i - T_k x) + Q_k(x^i) \leq \vartheta_k$$

if the problem in y_k^i had a solution

$$(u_k^i)^T (h_k - T_k x) \leq 0$$

otherwise

and the constraints added at iteration $j < i$ are

$$(u_k^j)^T (T_k x^j - x) + Q_k(x^j) \leq \vartheta_k$$

if the problem in y_k^j had a solution

$$(u_k^j)^T (h_k - T_k x) \leq 0$$

otherwise

Convergence: After finitely many iterations, the method finds the optimal solution if it exists

Justification of the constraints/cuts added at every iteration

- "Cuts": terminology from (discrete) linear programming
  - → Inequality that you can add to the problem without changing the optimal solution but that can restrict the feasible set
  - → Classical in decomposition techniques (Dantzig-Wolfe, Benders decomposition)
  - → In our case, we construct a description of the feasible set for the first-stage problem in $\mathbb{R}^K$ inequalities at a time

Our case :

At iteration i and for scenario $k \in \{1, \dots, K\}$, we consider the linear program

$$\underset{y_k^i \in \mathbb{R}^m}{\text{minimize}} \quad p_k q_k^T y_k^i \quad \text{s.t.} \quad W_k y_k^i = h_k - T_k x^i, \quad y_k^i \geq 0$$

($W_k \in \mathbb{R}^{l \times m}$, $h_k - T_k x^i \in \mathbb{R}^l$)

The dual of this problem is the linear program

$$\underset{u_k^i \in \mathbb{R}^l}{\text{maximize}} \quad (h_k - T_k x^i)^T u_k^i \quad \text{s.t.} \quad W_k^T u_k^i \leq p_k q_k$$

(1) If the problem in y_k^i has a solution, then

$$Q_k(x^i) = \min_{y_k^i \in \mathbb{R}^m} p_k q_k^T y_k^i \quad \text{s.t.} \quad W_k y_k^i = h_k - T_k x^i, \quad y_k^i \geq 0$$

is finite and there exists $u_k^i \in \mathbb{R}^l$ such that

$$Q_k(x^i) = (h_k - T_k x^i)^T u_k^i$$

($Q_k(x^i)$: optimal value for the problem in y_k^i; $(h_k - T_k x^i)^T u_k^i$: optimal value for the dual)

$\Rightarrow$ The solution to the main problem should satisfy

$$Q_k(x) \geq (u_k^i)^T (T_k x^i - T_k x) + Q_k(x^i)$$

($(u_k^i)^T (T_k x^i - T_k x) = (h_k - T_k x)^T u_k^i - (h_k - T_k x^i)^T u_k^i$)

$\Leftrightarrow Q_k(x) \geq (h_k - T_k x)^T u_k^i$: x is feasible for the problem in y_k^i

Alternate interpretation

The function $x \mapsto Q_k(x)$ is convex

Therefore it satisfies

$$Q_k(x) \geq Q_k(y) + g^T(x-y)$$

for any vectors x and y such that $Q_k(x)$ and $Q_k(y)$ are finite, and for any g subgradient of Q_k at y.

→ In our case, it is possible to show that

$T_k^T u_k^i$ is a subgradient for Q_k at x^i

Then for any x such that $Q_k(x)$ is finite, we have

$$Q_k(x) \geq Q_k(x^i) + (T_k^T u_k^i)^T (x - x^i)$$

$$= Q_k(x^i) + (u_k^i)^T (T_k x - T_k x^i)$$

For the master problem, we replace $Q_k(x)$ by an additional variable z_k to avoid explicit calculation of $Q_k(x)$

(2) The problem in y_i has no solution

⇒ Happens because of infeasibility

⇒ In that case, the dual problem is unbounded

i.e.

$$\max_{u_k^i \in \mathbb{R}^m} (h_k - T_k x^i)^T u_k^i \quad \text{s.t.} \quad W_k^T u_k^i \leq p_k q_k$$

$$= +\infty$$

⇒ there must exist a vector u_k^i that is feasible for the dual such that

$$(h_k - T_k x^i)^T u_k^i > 0$$

The corresponding inequality for the master problem will be

$$(u_k^i)^T (h_k - T_k x) \leq 0$$

If x is a solution, then the dual should be bounded

N.B.: Most (if not all) linear programming solvers would compute the dual variables u_k^i while solving the subproblem in y_k^i

Algorithm 2: Progressive hedging algorithm (PHA)

(in the optimization community: augmented Lagrangian method)

Idea: Introduce one here-and-now decision variable per scenario and force them to be equal to one another.

Formulation

minimize

$$\sum_{k=1}^{K} p_k \left(c^T x_k + q_k^T y_k \right)$$

Decomposition along the K scenarios

$x_1, \dots, x_K \in \mathbb{R}^n$

$y_1, \dots, y_K \in \mathbb{R}^m$

s.t.

$$A x_k = b \qquad k = 1..K$$

$$T_k x_k + W_k y_k = h_k \qquad k = 1..K$$

$$x_k \geq 0 \qquad k = 1..K$$

$$y_k \geq 0 \qquad k = 1..K$$

$$x_k = \sum_{l=1}^{K} p_l x_l \qquad k = 1..K$$

$x_1 = \dots = x_K$

$$\Rightarrow \sum_{k=1}^{K} p_k c^T x_k = \sum_{k=1}^{K} p_k c^T x_1 = \left(\sum_{k=1}^{K} p_k \right) c^T x_1 = c^T x_1$$

$$x_k = \sum_{l=1}^{K} p_l x_l \quad \forall k = 1..K \iff x_1 = x_2 = \dots = x_K$$

PHA algorithm

- Start with $\lambda_1^0 \in \mathbb{R}^n, \dots, \lambda_K^0 \in \mathbb{R}^n$ such that $\sum_{l=1}^{K} p_l \lambda_l^0 = 0$, $\mu > 0$
  Define $x_1^0, \dots, x_K^0 \in \mathbb{R}^n$.
- For $i = 0, 1, \dots$
  - For $k = 1, \dots, K$

    Compute (x_k^{i+1}, y_k^{i+1}) by solving

    $$\begin{aligned} \text{minimize}_{x,y} \quad & p_k (c^T x + q_k^T y) + \lambda_k^{i\,T} x + \frac{\mu}{2} \|x - \sum_{l=1}^{K} p_l x_l^i\|^2 \\ \text{s.t.} \quad & Ax = b, \ x \geq 0 \\ & T_k x + W_k y = h_k, \ y \geq 0 \end{aligned}$$

  - Set $\lambda_k^{i+1} = \lambda_k^i + \mu (x_k^{i+1} - \sum_{l=1}^{K} p_l x_l^{i+1}) \quad \forall k = 1..K$

→ Augmented Lagrangian method: Put a constraint into the objective function to compute points as solutions of the problem with a modified objective but without that constraint

$$\begin{aligned} \text{minimize}_{x_k, y_k} \quad & p_k (c^T x_k + q_k^T y_k) \quad \text{s.t.} \quad Ax_k = b \\ & T_k x_k + W_k y_k = h_k \\ & x_k \geq 0, \ y_k \geq 0 \end{aligned}$$

Requires to know the other x_ls → $x_k = \sum_{l=1}^{K} p_l x_l$

$$\underset{x_k, y_k}{\text{minimize}} \quad p_k\left(c^T x_k + q_k^T y_k\right) + \lambda_k^T\left(x_k - \sum_{\ell=1}^{K} p_\ell x_\ell\right) + \frac{\mu}{2}\left\| x_k - \sum_{\ell=1}^{K} p_\ell x_\ell \right\|^2$$

↑ Augmented Lagrangian w.r.t. the constraint $x_k = \sum_{\ell=1}^{K} p_\ell x_\ell$

$$\text{s.t.} \quad A x_k = b, \; x_k \geq 0$$
$$T_k x_k + W_k y_k = h_k, \; y_k \geq 0$$

By replacing $\sum_{\ell=1}^{K} p_\ell x_\ell$ with $\sum_{\ell=1}^{K} p_\ell \bar{x}_\ell$ where $\bar{x}_\ell$ were computed at the previous iteration, we obtain a problem that only involves x_k and y_k

$\Rightarrow$ Overall, we obtain K independent subproblems

$\rightarrow$ The term $\lambda_k^T\left(x_k - \sum_{\ell=1}^{K} p_\ell x_\ell\right)$ can be simplified to $\lambda_k^T x_k$ by choosing λ such that $\sum_{\ell=1}^{K} p_\ell \lambda_\ell = 0$

$$\sum_{k=1}^{K} \lambda_k^T\left(x_k - \sum_{\ell=1}^{K} p_\ell x_\ell\right) = \sum_{k=1}^{K} \lambda_k^T x_k - \sum_{k=1}^{K}\sum_{\ell=1}^{K} \lambda_k^T p_\ell x_\ell$$
$$= \sum_{k=1}^{K} \lambda_k^T x_k - \sum_{\ell=1}^{K}\left(\underbrace{\sum_{k=1}^{K} p_k \lambda_k}_{=0}\right)^T x_\ell$$
$$= \sum_{k=1}^{K} \lambda_k^T x_k$$