

Stochastic

Aayush Bajaj

July 10, 2026

Contents

1	two-stage programs with recourse	1
1.1	formulation	1
1.2	scenario trees	2
1.3	the deterministic equivalent	3
2	sample average approximation	3
3	chance constraints	4
3.1	when they convexify	4
3.2	when they do not	5
4	the newsvendor	5
4.1	deriving the critical fractile	5
4.2	numerical verification	6
5	progressive hedging	7
6	see also	7

a **linear program** assumes you know the data. stochastic programming admits that you do not — some coefficients are random — but insists you know their *distribution*, and asks for the decision that is best **on average**. the structural insight that makes this a paradigm rather than a hack: split the decision in two. commit to x now, before the coin is flipped; after uncertainty resolves, take a corrective *recourse* action y that adapts to whatever happened. the objective charges you for both, weighting the second stage by expectation.¹

dantzig again: his 1955 paper founded the field eight years after he invented simplex. beale published the same idea the same year

the “distribution is known” clause is a genuine modelling commitment — those probabilities have to come from somewhere (data, physics, or a prior you are prepared to defend (Jaynes, Edwin T., 2003)). when you can only bound the uncertainty set and refuse to weight it, you are doing **robust optimisation** instead: worst case rather than average case. the two paradigms bracket most practical uncertainty modelling.

1 two-stage programs with recourse

1.1 formulation

the canonical two-stage stochastic linear program:

¹the standard modern textbook is birge and louveaux, *introduction to stochastic programming*, 2nd ed., springer, 2011. dantzig’s founding paper is *linear programming under uncertainty*, management science 1(3-4), 1955.

$$\begin{aligned} & \text{minimise} && c^\top x + \mathbb{E}_\xi[Q(x, \xi)] \\ & \text{subject to} && Ax = b, \quad x \geq 0, \end{aligned}$$

where the **second-stage value function** $Q(x, \xi)$ is itself the optimal value of a linear program, solved *after* observing the random element $\xi = (q, T, W, h)$:

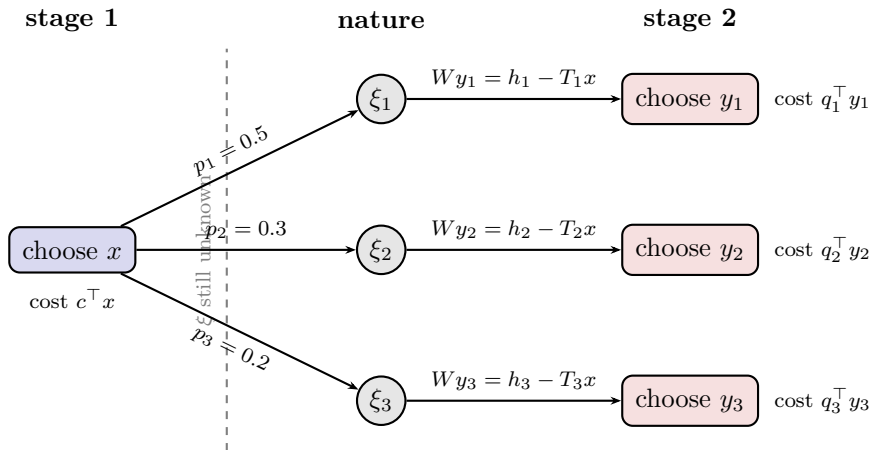
$$\begin{aligned} Q(x, \xi) = \min_y & \quad q^\top y \\ \text{subject to} & \quad Wy = h - Tx, \quad y \geq 0. \end{aligned}$$

vocabulary worth pinning down:

- **here-and-now** decision x : chosen under ignorance, shared by every future.
- **recourse** decision y : chosen with hindsight, one per realisation of ξ . T (the *technology matrix*) says how first-stage choices constrain the second stage; W (the *recourse matrix*) says what corrections are available.
- **fixed recourse**: W deterministic. **complete recourse**: $\{Wy : y \geq 0\} = \mathbb{R}^m$, so the second stage is feasible whatever you did — every mistake is patchable at a price. *relatively complete* recourse asks this only for first-stage-feasible x .
- $Q(x) = \mathbb{E}_\xi[Q(x, \xi)]$, the **expected recourse cost**, is convex in x (a minimum of an LP is convex in its right-hand side, and expectation preserves convexity); with finitely many scenarios and fixed recourse it is piecewise linear. so the whole problem is a convex program — large, but convex.

1.2 scenario trees

continuous ξ makes $Q(x)$ an integral of LP values — usually hopeless to evaluate exactly. the standard move is to discretise: approximate the distribution by finitely many **scenarios** $\xi_1, \dots, \xi_S$ with probabilities $p_1, \dots, p_S$. the resulting structure is a tree: one root where x lives, one branch per scenario, one recourse decision per leaf.



multistage versions branch again at every stage and the tree grows exponentially — the curse that keeps most practice at two stages

Figure 1: a two-stage scenario tree. the here-and-now decision x is a single node — that is nonanticipativity. nature then selects scenario s with probability p_s , and a scenario-specific recourse y_s cleans up.

1.3 the deterministic equivalent

with finitely many scenarios the expectation is a weighted sum, and the whole thing collapses into one ordinary — if enormous — LP, the **deterministic equivalent**:

$$\begin{aligned} \text{minimise} \quad & c^\top x + \sum_{s=1}^S p_s q_s^\top y_s \\ \text{subject to} \quad & Ax = b \\ & T_s x + W y_s = h_s \quad s = 1, \dots, S \\ & x \geq 0, \quad y_s \geq 0 \quad s = 1, \dots, S. \end{aligned}$$

its constraint matrix has the signature **dual block-angular** shape — a spine of first-stage columns coupling S otherwise-independent diagonal blocks:

$$\begin{bmatrix} A & & & & \\ T_1 & W & & & \\ T_2 & & W & & \\ \vdots & & & \ddots & \\ T_S & & & & W \end{bmatrix} \begin{bmatrix} x \\ y_1 \\ y_2 \\ \vdots \\ y_S \end{bmatrix} = \begin{bmatrix} b \\ h_1 \\ h_2 \\ \vdots \\ h_S \end{bmatrix} \quad (1)$$

the structure is the algorithm. fix x and the problem shatters into S small independent second-stage LPs — one per scenario, embarrassingly parallel. the **L-shaped method** (benders decomposition specialised to this matrix, named after the shape of the nonzero pattern) exploits exactly this: a master problem proposes x , the scenario subproblems return dual prices, and those duals become *optimality cuts* — supporting hyperplanes of the convex function $Q(x)$ — plus *feasibility cuts* when a proposal leaves some scenario unpatchable. the master re-solves against its growing outer approximation of Q until the bounds close.² with 10 first-stage variables and a million scenarios, the deterministic equivalent is a million-row LP, but each subproblem stays tiny.

2 sample average approximation

when scenarios are not given but the distribution can be *sampled*, replace the expectation with an empirical average over N iid draws $\xi^1, \dots, \xi^N$:

$$\min_{x \in X} \hat{f}_N(x) = c^\top x + \frac{1}{N} \sum_{i=1}^N Q(x, \xi^i), \quad (2)$$

and solve *that* — it is just a deterministic equivalent with equal weights $1/N$. this is **sample average approximation** (SAA): monte carlo on the objective, then honest deterministic optimisation.

why it works: for each fixed x the law of large numbers gives $\hat{f}_N(x) \rightarrow f(x)$ almost surely, and under mild conditions (compact X , integrable and well-behaved Q) the convergence is *uniform* in x — the same machinery that pushes the empirical cdf

²van slyke and wets, *l-shaped linear programs with applications to optimal control and stochastic programming*, siam journal on applied mathematics 17(4), 1969. benders' general partitioning scheme predates it (1962); the L-shaped method is benders pointed at the block structure above.

uniformly onto the true cdf (Wasserman, Larry, 2010). uniform convergence of objectives drags the minimisers along: the SAA optimal value $\hat{v}_N$ converges to the true v^* and SAA solutions accumulate on the true solution set.³

the honest caveats:

- **the in-sample value flatters you.** $\mathbb{E}[\hat{v}_N] \leq v^*$: minimising over the sample overfits the sample, so the reported optimum is biased optimistic (the bias shrinks monotonically in N). always re-evaluate the candidate $\hat{x}_N$ on a *fresh*, much larger sample to estimate its true cost.
- **rare events break it.** if the expensive scenarios have probability 10^{-4} , a thousand samples will likely contain none of them and SAA will confidently recommend a decision that has never seen a bad day.
- **no error bars for free.** one SAA run is a point estimate; confidence intervals need replications (solve M independent SAA instances) or the optimality-gap estimators built from the bias inequality above.
- **sample size is problem-dependent.** accuracy in the *solution* (not just the value) can demand far larger N than the central-limit heuristic suggests, especially with flat objectives near the optimum.

3 chance constraints

sometimes recourse is the wrong metaphor — some constraints must simply *hold with high probability*, no cleanup available. a **chance constraint** (charnes and cooper, 1959):

$$\mathbb{P}\left(a(\xi)^\top x \leq b\right) \geq 1 - \alpha, \quad (3)$$

for a small tolerance α : a bridge that carries the load on 99.9 percent of days, a portfolio that breaches its risk limit at most one day in twenty.

3.1 when they convexify

the gaussian case is the classic success. suppose $a \sim \mathcal{N}(\bar{a}, \Sigma)$. then $a^\top x \sim \mathcal{N}(\bar{a}^\top x, x^\top \Sigma x)$, so

$$\mathbb{P}\left(a^\top x \leq b\right) = \Phi\left(\frac{b - \bar{a}^\top x}{\sqrt{x^\top \Sigma x}}\right) \geq 1 - \alpha \iff \bar{a}^\top x + \Phi^{-1}(1 - \alpha) \|\Sigma^{1/2} x\|_2 \leq b, \quad (4)$$

where Φ is the standard normal cdf. for $\alpha \leq 1/2$ the quantile $\Phi^{-1}(1 - \alpha) \geq 0$ and this is a **second-order cone constraint** — convex, and solvable at scale by the interior-point machinery of conic programming. the probabilistic statement has been traded for a deterministic one: back off from the mean constraint by a safety margin of $\Phi^{-1}(1 - \alpha)$ standard deviations. squint and it is robust optimisation with an ellipsoidal uncertainty set — the two paradigms meeting in the middle.

³the SAA convergence theory, optimality-gap estimators, and the cvar approximation of chance constraints are all developed in shapiro, dentcheva and ruszczynski, *lectures on stochastic programming*, siam, 2009.

3.2 when they do not

- **the wrong tolerance.** $\alpha > 1/2$ flips the sign of the quantile and the same set turns nonconvex (a reverse cone). rarely wanted, but it shows convexity is a property of the *confidence level*, not just the distribution.
- **general distributions.** for arbitrary ξ the feasible set $\{x : \mathbb{P}(g(x, \xi) \leq 0) \geq 1 - \alpha\}$ can be nonconvex and even disconnected — probability mass can sit in awkward lumps, and unioning “mostly-feasible” regions does not preserve convexity.
- **finite scenarios go combinatorial.** with S scenarios, “hold in at least $(1 - \alpha)S$ of them” needs a binary indicator per scenario and big-M constraints: $a_s^\top x \leq b + Mz_s, \sum_s z_s \leq \alpha S$. that is an **integer program** — you are literally choosing *which* scenarios to sacrifice.
- **joint chance constraints.** asking several inequalities to hold *simultaneously* with probability $1 - \alpha$ is harder still: even in the gaussian case the multivariate normal cdf has no usable closed form. the workhorse fixes are conservative convex surrogates — replace the probability by a cvar (expected-shortfall) constraint, which is convex and implies the chance constraint, at the price of some feasibility headroom.³

4 the newsvendor

the fruit-fly of stochastic programming: every result in the field can be demonstrated on it, and it is the *only* interesting stochastic program with a closed-form solution.

*airline seats, hotel rooms,
fashion stock, vaccine orders,
cloud capacity — all
newsvendor problems in trench
coats*

a vendor orders q papers before knowing demand $D \sim F$. each unit of unmet demand costs c_u (foregone margin); each unsold paper costs c_o (writedown). this *is* a two-stage program: first stage “order q ”, second stage “observe D , absorb the mismatch” — with the recourse LP so trivial it evaluates in closed form:

$$C(q) = \mathbb{E} [c_o(q - D)^+ + c_u(D - q)^+] = c_o \int_0^q (q - t) f(t) dt + c_u \int_q^\infty (t - q) f(t) dt. \quad (5)$$

4.1 deriving the critical fractile

differentiate under the integral sign (leibniz rule — the boundary terms vanish because the integrands are zero at $t = q$):

$$\begin{aligned} C'(q) &= c_o \left[(q - t)f(t) \Big|_{t=q} + \int_0^q f(t) dt \right] + c_u \left[-(t - q)f(t) \Big|_{t=q} - \int_q^\infty f(t) dt \right] \\ &= c_o F(q) - c_u (1 - F(q)). \end{aligned}$$

so the marginal paper is worth ordering exactly while $c_u \mathbb{P}(D > q)$, the expected gain from covering one more unit of demand, exceeds $c_o \mathbb{P}(D \leq q)$, the expected loss from one more leftover. moreover $C''(q) = (c_o + c_u)f(q) \geq 0$, so C is convex and the first-order condition is the global minimum:

$$F(q^*) = \frac{c_u}{c_u + c_o} \iff q^* = F^{-1}\left(\frac{c_u}{c_u + c_o}\right) \quad (6)$$

the **critical fractile**. read it as a service level: stock to cover demand with probability $c_u/(c_u + c_o)$. expensive shortages push q^* into the right tail; expensive leftovers pull it back. for discrete or non-invertible F , take the generalised inverse (smallest q with $F(q) \geq \text{fractile}$) — which is exactly why the SAA solution below is a sample quantile.

4.2 numerical verification

exponential demand with mean μ keeps everything analytic: $F(q) = 1 - e^{-q/\mu}$, so $q^* = \mu \ln((c_u + c_o)/c_o)$. with $c_u = 3$, $c_o = 1.5$ the fractile is $2/3$ and $q^* = \mu \ln 3 \approx 109.86$. four independent routes to the same number — closed form, SAA (which for the newsvendor *is* the empirical $2/3$ -quantile), a brute-force monte-carlo sweep over q , and an honest two-stage deterministic-equivalent LP with 2000 scenarios and 4001 variables:

```

1  import numpy as np
2  from scipy.optimize import linprog
3
4  rng = np.random.default_rng(7)
5
6  # newsvendor: order q before demand D is seen.
7  # underage cost cu per unit short, overage cost co per unit left over.
8  cu, co = 3.0, 1.5                                # lost margin vs writedown
9  mu = 100.0                                         # exponential demand, mean mu
10 frac = cu / (cu + co)                             # critical fractile = 2/3
11
12 # 1. closed form: F(q) = 1 - exp(-q/mu) => q* = -mu ln(1 - frac)
13 q_closed = -mu * np.log(1.0 - frac)
14
15 # 2. SAA: empirical quantile of a demand sample
16 d = rng.exponential(mu, size=200_000)
17 q_saa = np.quantile(d, frac)
18
19 # 3. brute force: monte-carlo expected cost on a grid of q
20 qs = np.linspace(60, 160, 401)
21 cost = np.array([(co * np.maximum(q - d, 0) + cu * np.maximum(d - q, 0)).mean()
22                  for q in qs])
23 q_brute = qs[cost.argmin()]
24
25 # 4. two-stage deterministic equivalent LP on S scenarios:
26 #   min sum_s p_s (co*u_s + cu*v_s)  s.t.  u_s >= q - d_s,  v_s >= d_s - q
27 S = 2_000
28 ds = rng.exponential(mu, size=S)
29 p = np.full(S, 1.0 / S)
30 # variables: [q, u_1..u_S, v_1..v_S]
31 c_lp = np.concatenate([0.0, co * p, cu * p])
32 # -q + u_s >= -d_s  ->  q - u_s <= d_s
33 A1 = np.hstack([np.ones((S, 1)), -np.eye(S), np.zeros((S, S))])
34 #  q + v_s >= d_s  -> -q - v_s <= -d_s
35 A2 = np.hstack([-np.ones((S, 1)), np.zeros((S, S)), -np.eye(S)])
36 res = linprog(c_lp, A_ub=np.vstack([A1, A2]),
37               b_ub=np.concatenate([ds, -ds]),
38               bounds=[(0, None)] * (1 + 2 * S), method="highs")
39 q_lp = res.x[0]
40
41 print(f"critical fractile    cu/(cu+co) = {frac:.4f}")
42 print(f"closed form          q* = mu ln 3      = {q_closed:.3f}")
43 print(f"SAA quantile           (n = 200000)      = {q_saa:.3f}")

```

```

44 print(f"brute-force sweep (grid, same MC) = {q_brute:.3f}")
45 print(f"det-equivalent LP (S = 2000) = {q_lp:.3f}")
46 print(f"expected cost at q* (MC) = {np.interp(q_closed, qs,
    ↪ cost):.3f}")

```

critical fractile	$cu/(cu+co) = 0.6667$
closed form	$q^* = \mu \ln 3 = 109.861$
SAA quantile	$(n = 200000) = 109.682$
brute-force sweep	$(\text{grid, same MC}) = 109.750$
det-equivalent LP	$(S = 2000) = 111.678$
expected cost at q^*	$(\text{MC}) = 164.447$

everything lines up. the 200k-sample estimates land within 0.2 of the truth; the LP, fed only 2000 scenarios, lands within 2 — and its error is pure SAA sampling error (its exact optimum *is* the 2/3 sample quantile of its own 2000 draws), a live demonstration of the caveats above. note also how flat the objective is near q^* : convex expected costs forgive small ordering errors, which is why the crude 0.25-wide grid still gets three digits.

5 progressive hedging

the L-shaped method decomposes by *stage*; **progressive hedging** decomposes by *scenario*. give every scenario its own full copy of the first-stage decision, x_s , and solve each scenario's deterministic problem independently — trivially parallel, but now scenario 3's plan assumes it knows scenario 3 is coming, which is cheating. the fix is to state the no-cheating rule explicitly as a constraint, **nonanticipativity**: $x_s = \bar{x}$ for all s , where $\bar{x} = \sum_s p_s x_s$ is the consensus decision. progressive hedging enforces it softly via an augmented lagrangian: at each iteration, solve the per-scenario problems with an added penalty $w_s^\top x_s + \frac{\rho}{2} \|x_s - \bar{x}\|^2$, recompute the consensus $\bar{x}$, and update the multipliers $w_s \leftarrow w_s + \rho (x_s - \bar{x})$. rockafellar and wets proved convergence for convex problems (the scheme is a proximal point algorithm in disguise);⁴ in practice it is used far beyond that warranty, as a remarkably effective heuristic when the scenario subproblems are mixed-integer and the L-shaped method's convexity assumptions have left the building.

6 see also

- **linear programming** — the deterministic equivalent is one of these, wearing a very wide coat
- **robust optimisation** — the same uncertainty without the probabilities: worst case, not average case
- **integer programming** — where scenario-counting chance constraints and integer recourse end up
- **non-linear programming** — the general machinery behind the conic and cvar reformulations

⁴rockafellar and wets, *scenarios and policy aggregation in optimization under uncertainty*, mathematics of operations research 16(1), 1991.

references

Jaynes, Edwin T. (2003). *Probability Theory: The Logic of Science*, Cambridge University Press.

Wasserman, Larry (2010). *All of Statistics: A Concise Course in Statistical Inference*, Springer.