Chapter 6
Parametric Stochastic Programming with One Chance Constraint: Gaining Insights from Response Space Analysis

Harvey J. Greenberg, Jean-Paul Watson, and David L. Woodruff

Abstract We consider stochastic programs with discrete scenario probabilities where scenario-specific constraints must hold with some probability, which we vary parametrically. We thus obtain minimum cost as a function of constraint-satisfaction probability. We characterize this trade-off using Everett's response space and introduce an efficient construction of the response space frontier based on tangential approximation, a method introduced for one specified right-hand side. Generated points in the response space are optimal for a finite set of probabilities, with Lagrangian bounds equal to the piece-wise linear functional value. We apply our procedures to a number of illustrative stochastic mixed-integer programming models, emphasizing insights obtained and tactics for gaining more information about the trade-off between solution cost and probability of scenario satisfaction. Our code is an extension of the PySP stochastic programming library, included with the Pyomo (*Python Optimization Modeling Objects*) open-source optimization library.

Electronic Supplementary Material: The online version of this chapter (https://doi.org/10.1007/978-3-030-56429-2_6) contains supplementary material, which is available to authorized users.

The author "Harvey J. Greenberg" is deceased at the time of publication.

H. J. Greenberg
Mathematics Department, University of Colorado, Denver, CO, USA

J.-P. Watson (✉)
Center for Applied and Scientific Computing and Global Security Directorate, Lawrence Livermore National Laboratory, Livermore, CA, USA
e-mail: jeanpaulwatson@llnl.gov

D. L. Woodruff
Graduate School of Management, University of California Davis, Davis, CA, USA
e-mail: dlwoodruff@ucdavis.edu

A. Holder (ed.), *Harvey J. Greenberg*, International Series in Operations Research & Management Science 295, https://doi.org/10.1007/978-3-030-56429-2_6

6.1 Introduction and Background

We consider stochastic programs with discrete probabilities where one or more constraints must hold jointly with some probability, β, which we vary parametrically between zero and one. We refer to the probability requirement as a *chance constraint* [20, 25]. Chance constraints make sense in many settings for various reasons, among them: (1) when constraints represent adherence to policies rather than laws of physics, it may be deemed too expensive to comply with all constraints under all circumstances; (2) when the discrete probabilities are the result of sampling from continuous distributions or from simulation realizations, it is simply a form of false advertising to claim that constraints will hold with probability 1, so it may make sense to relax away from 1 under the control of a parameter. The usefulness of chance constraints has led to a large body of research directed at solving these sorts of problems for a given value of β (see, e.g., [1, 20, 21, 24, 30, 31]).

Let $S = \{1, \ldots, N_S\}$ represent the set of scenario indexes. Each of the N_S scenarios gives a full set of the data for a constrained minimization problem, and we associate the symbol p_s with the probability that scenario $s \in S$ will be realized, where $\sum_{s \in S} p_s = 1$. Following [28], we assume that the problem formulation includes N_S binary variables, δ, that take the value one if there must be compliance with scenario-s constraints. Although $\delta_s = 0$ allows violation (i.e., non-compliance), we discuss a model extension to allow the converse: $\delta_s = 0$ only if some scenario-s constraint is violated (see Sect. 6.6.2).

There are different formulations that fit under this rubric. For example, consider a two-stage, chance-constrained, stochastic program where the first-stage variables, x, are constrained to be in a set X. The second-stage variables, $\{y_s\}_{s \in S}$, are constrained by $y_s \in Y_s(x, \delta)$. In particular, suppose the function to be minimized is $c(x) + \sum_{s \in S} p_s h_s(x, y_s)$, where c and $\{h_s\}_{s \in S}$ are functionals and

$$Y_s(x, \delta) = \{y_s \in \mathcal{Y}^s \colon A_s x + B_s y_s \geq \delta_s d_s - (1 - \delta_s) M_s\}, \text{ for } x \in X, \tag{6.1}$$

where M_s is sufficiently large to render scenario-s constraints redundant for $\delta_s = 0$; $\mathcal{Y}^s$ may be simply $\mathbb{R}^{m_s}$ or it may constrain some variables to be integer-valued.

Only a proper subset of the constraints form the joint chance constraint in some applications. In order to capture a wide range of chance-constrained models, we express the general idea by using $z^*(\delta)$ to represent the result of solving the extended minimization problem with an indicator vector, δ. We thus define the chance-constrained problem as:

$$\text{CC}: \ \min z^*(\delta) : p\delta \geq \beta, \ \delta \in \{0, 1\}^n, \tag{6.2}$$

where $p\delta \overset{\text{def}}{=} \sum_{s \in S} p_s \delta_s$.

We think of CC computationally as a decomposition with an outer problem to select scenarios by setting their corresponding $\delta_s = 1$; the inner problem is what defines z^*. Specifically, the two-stage stochastic program with joint chance constraints uses Y_s as defined in (6.1) to obtain

$$z^*(\delta) = \min\left\{c(x) + \sum_{s\in S} p_s h_s(x, y_s) : x \in X,\ y_s \in Y_s(x, \delta),\ \forall s \in S\right\}.$$

To compute solutions under parametric variation of β, we form the Lagrangian of CC:

$$L^*(\lambda) \overset{\text{def}}{=} \min\left\{z^*(\delta) - \lambda p\delta : \delta \in \{0,\ 1\}^n\right\}. \tag{6.3}$$

Each Lagrangian gives a lower bound on the minimum cost:

$$f^*(\beta) \overset{\text{def}}{=} \min\{z^*(\delta) : p\delta \geq \beta,\ \delta \in \{0,\ 1\}^n\} \ \geq\ L^*(\lambda) + \lambda\beta. \tag{6.4}$$

The optimal multiplier, λ^*, gives the tightest bound:

$$L^*(\lambda^*) + \lambda^*\beta = \max_{\lambda\geq 0}\{L^*(\lambda) + \lambda\beta\},$$

which is the weak Lagrangian dual. The Lagrangian gap is the difference in optimal objective values:

$$G(\beta) \overset{\text{def}}{=} f^*(\beta) - (L^*(\lambda^*) + \lambda^*\beta).$$

Let $\delta^* \in \operatorname{argmin}\{z^*(\delta) - p\delta : \delta \in \{0,\ 1\}^n\}$. We have $G(\beta) = 0$ if, and only if, complementary slackness holds: $\lambda^* > 0 \Rightarrow p\delta^* = \beta$. This follows from $f^*(\beta) = z^*(\delta^*)$, and hence $G(\beta) = \lambda^*(p\delta^* - \beta)$.

If $\beta = 0$, no scenarios need to be selected, so $\delta = 0$ is optimal and $\lambda = 0$ is an optimal multiplier. Otherwise, if the optimal solution satisfies $p\delta^* = \beta$, then it solves the original problem (6.2). In the more typical cases, either the probabilities are such that there is no vector $\delta \in \{0, 1\}^n$ for which $p\delta = \beta$, or such vectors are suboptimal. There are two alternative Lagrangian optima in these cases, δ^L and δ^U, such that $b^L = p\delta^L < \beta < p\delta^U = b^U$. The interval (b^L, b^U) is called the *gap region*.

The best feasible solution corresponds to b^U, with min-cost $z^U = z^*(\delta^U)$. The Lagrangian duality gap is bounded by

$$G(\beta) = f^*(\beta) - \left(L^*(\lambda^*) + \lambda^*\beta\right) = f^*(\beta) - \left(z^U - \lambda^* b^U + \lambda^*\beta\right) \leq \lambda^*\left(b^U - \beta\right),$$

where the last inequality follows from the fact that $\beta < b^U \Rightarrow f^*(\beta) \leq f^*(b^U) = z^U$. If we think of λ^* as a unit price, then the bound value is the total cost of the discrepancy, $b^U - \beta$. We use a dimensionless measure of solution quality, called the *relative Lagrangian gap*:

$$g(\beta) = \frac{\lambda^* \left(b^U - \beta\right)}{z^U}. \tag{6.5}$$

While our main goal is to use a chance-constraint stochastic programming model in support of decision-making, we go beyond the model and algorithm descriptions by emphasizing a maxim of good decision support: *The purpose of mathematical programming is insight, not numbers*[6]. We envision an environment where the mathematical program without the chance constraint is computationally difficult, so a best algorithm is one that needs the fewest Lagrangian solutions. Furthermore, we see the user as an analyst who wants to see a broad range of the efficient frontier, f^*, but not necessarily those points that add significant computational difficulty. Thus, seeing the convex envelope, F^*, presents a useful graph in its own right. Besides the generated points, where $f^* = F^*$, we provide a visual of how close the cost is for some particular β. The user can then choose regions for which the gap, $f^* - F^*$, needs to be tightened. The restricted flipping heuristic offers a framework for doing this, and the analyst could specify regions of search or use our automatic search based on uncertainty measured by the length of the gap interval, $b^U - b^L$.

There are cases where a probability is (or appears to be) specified. For example, consider the case of a government regulation on sulfur emissions. A company may want parametric analysis to substantiate a challenge based on how much the regulation costs, particularly if a small relaxation of the regulation costs much less. The government may want to analyze consideration of a tax that incentivizes compliance with the impact of keeping emissions and cost low. The Lagrange multipliers provide bounds on a tax that associates cost with compliance probability. (See LP Myth 23 in [12] to avoid seeing the tax as equivalent to the optimal multiplier.)

The rest of this chapter is organized as follows. The response space in which trade-offs are displayed is defined in Sect. 6.2. An algorithm that finds the optimal Lagrange multiplier is described in Sect. 6.3. Some of our computational search can be mitigated by the pre-processing methods in Sect. 6.4, and we emphasize the insight that tells us when a scenario must be selected. Examples based on instances of three models are given in the Supplementary Material for this chapter (https://github.com/DLWoodruff/GWW). These are used to illustrate methods for finding additional points in the response space in Sect. 6.5. Section 6.6 provides information about details that arise when implementing algorithms that map the trade-offs between probability and cost. The chapter closes with a summary and conclusions. The methods described in this chapter have been implemented as an extension to the PySP stochastic programming library [32], which is distributed as part of the Pyomo [16, 17] algebraic modeling language.

6.2 Response Space Analysis

Everett's seminal paper [5] introduced the Payoff-Resource (PR) space, which is the range of the objective and constraint functions. His mathematical program was a maximization of a payoff subject to resource limits. Our model is a minimization of cost subject to a probability of scenario satisfaction, so we call it more simply the response space (RS). (See *Mathematical Programming Glossary* [18].) He also introduced the term "gap," which is now entrenched in our vocabulary, to mean the difference between the primal and dual objective values. There was a stream of foundational papers that deepened our knowledge of general (non-convex) duals based on Everett's Generalized Lagrange Multiplier method (GLM)—see [2, 4, 7–9, 15, 29, 33]. We present the main concepts focused on one joint chance constraint with uncertainties that can be involved in both the left-hand side matrix and the right-hand side vector. Our purpose is to elucidate the results, particularly the search for an optimal Lagrange multiplier and the source of a Lagrangian duality gap, to gain insight.

The set of feasible right-hand sides for the chance-constraint problem is $B = [0, P^{\max}]$. For now, assume $P^{\max} = 1$. Since scenarios may compete for common resources it may not be possible to achieve $P^{\max} = 1$, so it is important to consider $P^{\max} < 1$, and it may be the reason for a chance-constraint model. However, in the interest of clarity, we defer this point until after we present the main results.

The response space compares the range of probability to cost over scenario-selection values, δ:

$$\mathrm{RS} = \{(b, z) \colon b = p\delta,\ z = z^*(\delta) \text{ for some } \delta \in \{0,\ 1\}^n\}. \tag{6.6}$$

It is helpful to realize that each Lagrangian contour in response space is a line, regardless of the structure of decision space and objective function. Furthermore, the transition from decision space to response space makes evident that the maximum-Lagrangian is the *convex envelope function*, F^* (also called the second convex conjugate of f^*) [14]:

$$F^*(\beta) = \max_{\lambda \geq 0} \min_{b \in B} \{f^*(b) - \lambda b + \lambda\beta\}. \tag{6.7}$$

The epigraph of $[F^*, B]$ is geometrically the closed convex hull of the epigraph of (f^*, B), denoted by

$$\mathrm{epi}(F^*, B) = \mathrm{convh}(\mathrm{epi}(f^*, B)). \tag{6.8}$$

Figure 6.1 illustrates this epigraph, where each point is the probability, $b = p\delta$, and cost, $z = z^*(\delta)$. Each line supports its epigraph:

$$\mathrm{epi}(F^*, B) = \{(b, z) \colon b \in B,\ z \geq F^*(b)\}. \tag{6.9}$$

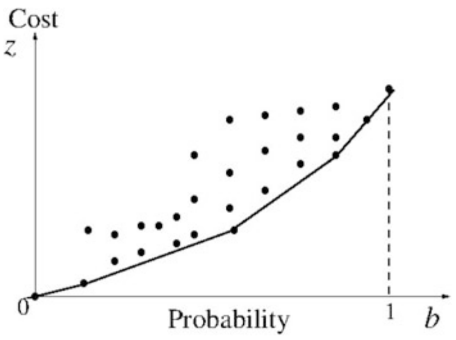


Fig. 6.1 Response Space as the range of $\delta \to \big(b = p\delta,\ z = z^*(\delta)\big)$

At each change in slope, the (b, z) point corresponds to an integer optimum that defines endpoints of the line segment whose slope is the Lagrange multiplier that produces the support for epi(f^*, B). If $\beta^* \in (b^L, b^U)$ (i.e., not one of the endpoints), then it is theoretically possible to find it, but to do so requires an enumeration of alternative optimal δ values. Only the endpoints are generated because they have alternative optimal multipliers. (Note that it is possible that an initial solution happens to obtain $(\beta, f^*(\beta))$, but once the iterations begin, only the endpoints are generated.) Thus, every $\beta \in (b^L, b^U)$ is essentially in a gap even though the gap value may be zero.

Because the Lagrangian approach provides a decomposition of scenarios, we can fit it into the PySP framework by simply adjusting the stage-two objective function to include the Lagrangian penalty cost. Luedtke [22] takes an alternative decomposition approach designed to obtain points on $[f^*, B]$, the efficient frontier of cost versus probability. Our Lagrangian approach focuses on computational efficiency by first obtaining points on $[F^*, B]$, followed by exploratory analysis of RS that includes sub-optimal solutions.

Here is a summary of the main points about response space.

- Each point in decision space, $\delta \in \{0,\ 1\}^{N_S}$, maps to a point in response space, $(b, z) \in$ RS.
- A Lagrangian contour in RS is a line with slope = λ.
- The bound, $f^*(\beta) \geq L^*(\lambda) + \lambda\beta$, is the support-line value at $b = \beta$.
- The Lagrangian dual gives the tightest Lagrangian bound, $\lambda^* \in \text{argmax}\{L^*(\lambda) + \lambda\beta\}$.
- The optimal multiplier, λ^*, is unique if, and only if, β is in a gap, in which case $\beta \in (b^L, b^U)$ and $\lambda^* = \dfrac{z^U - z^L}{b^U - b^L}$.

6.3 Multiplier Search

We now review the method of tangential approximation [10] to find an optimal Lagrange multiplier and then extend it to find the entire envelope function.

6.3.1 Search for One Optimal Multiplier

There are several ways to search for one optimal Lagrange multiplier, but tangential approximation was proposed as an efficient scheme [10]. For CC, it converges finitely to λ^* whether β is in a gap or not.

The general class of interval reduction algorithms includes bisection and linear interpolation, analyzed in [10]. Unlike tangential approximation, they are not guaranteed to converge finitely although it is possible to construct numerical examples for which they converge immediately. For example, suppose the initial interval of the multiplier search is $\lambda \in (0, \lambda^{\max})$ and $\lambda^* = \lambda^{\max}/2$. If we assume β is in a gap, which is likely in our binary model, then the optimal multiplier is unique—only extreme values of (b, z) yield a range, $\lambda^* \in [\lambda^L, \lambda^U]$ for $\beta \in (b^L, b^U)$. The multipliers are the left and right derivatives of F^*, respectively:

$$\lambda^L = \frac{\partial^- F^*(\beta)}{\partial \beta} \leq \frac{\partial^+ F^*(\beta)}{\partial \beta} = \lambda^U. \tag{6.10}$$

One optimal search for λ^* is Fibonacci, which minimizes the maximum number of functional evaluations (i.e., Lagrange solutions). One problem is with initialization: setting $\lambda^U = \infty$ (some big number). Another problem is getting close to λ^* but not converging finitely, in which case the computed gap region could be much wider than the actual value.

The tangential approximation search for one optimal multiplier, $\lambda^*(\beta)$, begins with the search intervals $(0, z^*(\vec{0}))$ and $(1, z^*(\vec{1}))$. These are obtained by $\delta \overset{\text{fix}}{=} \vec{0}$ and $\delta \overset{\text{fix}}{=} \vec{1}$, respectively. (We address the case where $\delta \overset{\text{fix}}{=} \vec{1}$ is infeasible in Sect. 6.6.3.) At a general iteration we have $(b^L, z^L), (b^U, z^U) \in \text{RS}$ such that $b^L < \beta < b^U$, $z^L = f^*(b^L) < f^*(b^U) = z^U$. We set λ equal to the slope of the line segment joining these two points:

$$\lambda = \frac{z^U - z^L}{b^U - b^L}. \tag{6.11}$$

Computing $L^*(\lambda)$ yields the point on the support: $(b = p\delta^*,\ z = z^*(\delta^*)) \in \text{RS}$ so that $b \in [b^L, b^U]$. If $b = \beta$, then we are done and λ is an optimal multiplier, and the chance-constraint instance is solved. If $b = b^L$ or $b = b^U$, then we terminate with the gap region, (b^L, b^U), which contains β. We otherwise shrink the interval of search by replacing (b^L, z^L) or (b^U, z^U) according to whether $b < \beta$ or $b > \beta$,

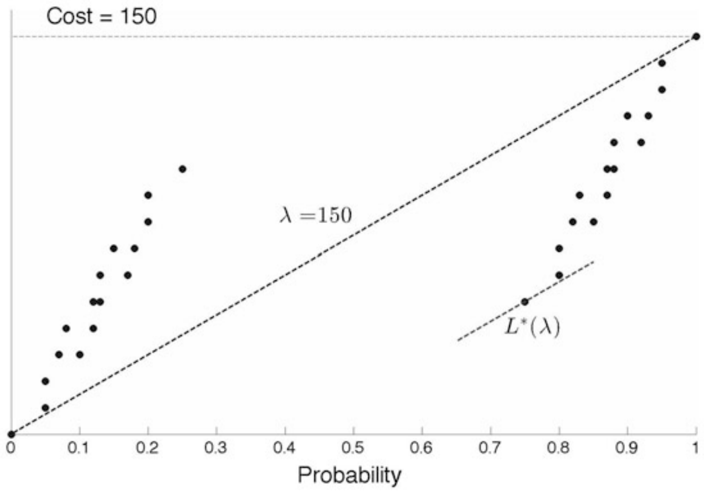


Fig. 6.2 Complete Response Space for Example 6.1 (32 points)

respectively. Because RS is finite, this must converge in a finite number of iterations, and our experiments indicate that it requires very few iterations.

Example 6.1 Suppose $z^*(\delta) = c\delta$ and we have the following five scenarios:

	Scenario				
	1	2	3	4	5
Probability (p)	0.05	0.05	0.07	0.08	0.75
Cost (c)	10	20	30	40	50

Figure 6.2 shows the complete response space, which has 32 points, corresponding to the 2^5 subsets of selections.

The slope of the line segment joining (0, 0) and (1, 150) is the initial Lagrange multiplier, $\lambda = 150$. Minimizing $L(\delta, \lambda) = z^*(\delta) - \lambda p\delta$ moves the line down (parallel) to become the support of $\text{epi}(f^*, B)$ and of $\text{epi}(F^*, B)$ at $(b, z) = (0.75, 50)$. We would terminate with the exact solution (no gap) if $\beta = 0.75$. Otherwise, the left point is replaced and the interval becomes [0.75, 1] if $\beta > 0.75$; the right point is replaced and the interval becomes [0, 0.75] if $\beta < 0.75$.

6.3.2 Parametric Search Algorithm

We extend tangential approximation to parametric analysis of $\beta \in [\beta^{\min}, \beta^{\max}] \subseteq [0, 1]$. The trade-off between cost and probability is a decision support tool that helps a policy analyst understand impacts, notably the proverbial: *What if* I loosen/tighten the probability? *How* does it affect cost? The analyst may also want to explore: *Why* was this scenario selected and that one not?

The Lagrangian approach uses multiplier values as a trade-off between cost and compliance probability. Varying λ generates $\beta^1, \ldots, \beta^K$, such that $f^*(\beta^k) = F^*(\beta^k) = L^*(\lambda^k) + \lambda^k\beta^k$, thus creating points on the efficient frontier of the bi-objective problem, Pareto-min$\{-b, z\}$: $(z, b) \in$ RS; see [26, 27] for alternative approaches. MOP Myth 2 in [12] also shows how a Lagrangian duality gap relates to Pareto-frontier generation. The difference with our Lagrangian approach is that it can be done efficiently and can provide additional perspectives of the multiplier values—viz., each break point on the piece-wise linear convex envelope has a range of multiplier values. See, e.g., [3, 34] for early connections between parametric linear programming and multiple objectives.

One approach is to specify a sample of target probabilities. This may be adequate if the Lagrangian problem is solved within a few minutes. Our applications, however, require many minutes (sometimes more than an hour) to solve one Lagrangian problem, so our PySP extension is designed to obtain the convex envelope of the response function for more computer-intensive reference models.

For a specified probability, tangential approximation is efficient among interval reduction methods [10], but it is not dominant. Shen [30] uses bisection, which may obtain an optimal multiplier in just one iteration, once there are two initial solutions with $b^L < \beta < b^U$. It may be (due to the problem instance) that $\lambda^* = \frac{1}{2}(\lambda^L + \lambda^U)$. In a worst case, however, bisection may not generate any new RS point, and it may not confirm the region as the gap region for β. The reason is that if $\beta \in (b^L, b^U)$ is in a gap and $|\lambda_i - \lambda^*|$ is sufficiently small, but $\lambda_i \neq \lambda^*$ for any (finite) i, then $\lambda_i < \lambda^* \to b_i = b^L$ and $\lambda_i > \lambda^* \to b_i = b^U$. Only tangential approximation is guaranteed to set $\lambda_i = \lambda^* = (z^U - z^L)/(b^U - b^L)$ once b^L and b^U are generated, thus terminating with the confirmation that β is in the gap region, (b^L, b^U).

Our method is an extension of tangential approximation that computes the minimum number of Lagrangian solutions to obtain the breakpoints in the piece-wise linear envelope. Other methods may compute solutions that provide no new information, for example, by generating a point on the convex envelope already generated by another Lagrange multiplier. This occurs if the probabilities are in the same gap region. None of the target probabilities are known to be on the convex envelope except for $\beta = 0$ and $\beta = 1$, so choosing a sparse set of targets could provide little information to the analyst.

Initialization Set $\lambda = 0$, fix $\delta_s = 0$ for all $s \in S$, and solve the Lagrangian problem (6.3). If the Lagrangian is infeasible, so is CC problem (6.2) for all β. Otherwise, the solution yields the point $(0, z_0) \in$ RS.

Next, fix $\delta_s = 1$ for all $s \in S$ and solve the Lagrangian with $\lambda = 0$, making $L^*(\lambda)$ to be the cost. If the Lagrangian is unbounded, then so is CC problem (6.2) for all β. If it is infeasible, then set λ to some large value and solve to obtain the maximum probability attainable (see Sect. 6.6.3). The solution otherwise yields $(1, z_1) \in \text{RS}$. Initialization ends with two points in RS : $(0, z_0)$ and $(1, z_1)$. Set $\mathcal{I} = \{[0, 1]\}$ and $L_1 = 0$.

Fathoming Gap Intervals At a general iteration we have a sequence of intervals, $\mathcal{I} = \{[b_0, b_1], [b_1, b_2], \dots, [b_{n-1}, b_n]\}$, with associated min-costs, $\{z_i\}_0^n$, and truth labels, $\{L_i\}_1^n \in \{0, 1\}$. $L_i = 1$ indicates the ith interval is fathomed, meaning that it is the gap region for $\beta \in (b_{i-1}, b_i)$. Otherwise, the associated interval needs to be searched if $L_i = 0$.

Choose an interval that is not fathomed. There are tactical selections such as choosing an interval with the greatest Lagrangian gap value. Such tactics are important if each Lagrangian minimization takes so much time that termination may need to occur before the parametric solution is complete. Set λ as one iteration of tangential approximation:

$$\lambda = \frac{z_i - z_{i-1}}{b_i - b_{i-1}}.$$

Solve the Lagrangian to obtain the response space point (b, z), where $b \in [b_{i-1}, b_i]$. If $b = b_{i-1}$ or $b = b_i$, set $L_i = 1$ and $\lambda_i = \lambda$. Otherwise, do one of the following:

Case 1: $b < \beta^{\min}$ (must have selected the interval $[b_0, b_1]$). Replace $b_0 = b$.
Case 2: $b > \beta^{\max}$ (must have selected the interval $[b_{n-1}, b_n]$). Replace $b_n = b$.
Case 3: $\beta^{\min} \le b_{i-1} < b < b_i \le \beta^{\max}$. Split the interval into $[b_{i-1}, b]$ and $[b, b_i]$. Re-index to maintain $b_0 < b_1 < \dots < b_n$.

This update maintains $b_0 \le \beta^{\min} \le b_1 < \dots < b_{n-1} \le \beta^{\max} \le b_n$. We are done when all intervals are fathomed. The scheme terminates in a finite number of iterations since there is a finite number of gap regions, each detected by tangential approximation of its endpoints.

The result is the sequence of successive points in the response space, $\{(b_i, z_i)\}_0^n$, and their associated, optimal multipliers, $\{\lambda_i\}_0^n$:

$$\lambda_0 = 0, \ \lambda_i = \frac{z_i - z_{i-1}}{b_i - b_{i-1}} \text{ for } i = 1, \dots, n.$$

We provide a function that computes the Lagrangian bound and best feasible solution for each $\beta \in [\beta^{\min}, \beta^{\max}]$ from the algorithm's terminal information. Specifically, find the interval that contains β: $b_{i-1} \le \beta \le b_i$. Then, (b_i, z_i) is the best feasible solution, and the Lagrangian bound is $F^*(\beta) = L^*(\lambda_i) + \lambda_i\beta = z_i + \lambda_i(\beta - b_i)$. The relative Lagrangian gap is thus $g(\beta) = 1 - F^*(\beta)/z_i \in (0, 1]$. Note that $g(\beta) > 0$ because $F^*(\beta) < z_i$ for $\beta < b_i$. We now have the following property.

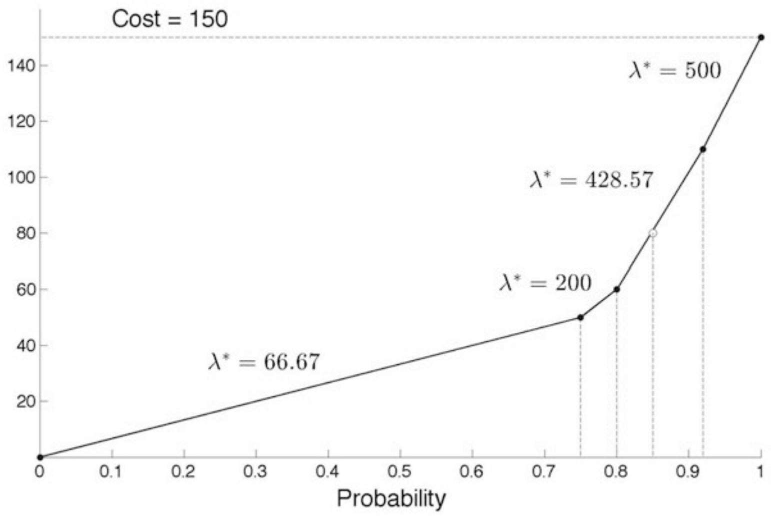


Fig. 6.3 Result of parametric multiplier search for Example 6.1 (c.f., Fig. 6.2)

Property 6.1 *Parametric Tangential Approximation solves the complete parametric CC model with the minimum number of Lagrangian optimizations.*

Our parametric algorithm reduces to the method of tangential approximation for one β. This follows because $\beta^{\min} = \beta^{\max}$ implies we always have either Case 1 or Case 2, thus shrinking the one interval of search and never splitting the interval. At the other extreme, if $\beta^{\min} = 0$ and $\beta^{\max} = 1$, Case 3 always applies and the interval is split. Hence, the number of Lagrangian minimizations is equal to the number of gap regions.

Figure 6.3 shows the result of parametric search for $\beta \in [0, 1]$. The only points generated are the endpoints of each gap interval. The point on F^* at $b = 0.85$ is an alternative optimum to the Lagrangian defined by the slope of the line segment, and this point is not generated. In Sect. 6.5 we describe techniques for generating additional points.

The computational time is dominated by the time it takes to minimize the Lagrangian to obtain z^*. Initialization requires two computations, but all δ values are fixed, so it is the time needed to solve the instance without the chance constraint ($\delta = \vec{0}$) plus the time needed to solve the complete extended form with all scenarios being satisfied ($\delta = \vec{1}$). Each subsequent iteration solves the original instance with the N_S additional binary variables, δ, plus all scenario constraints present with associated δ to indicate whether to require their satisfaction. Each Lagrangian solution yields a point on the envelope, so the total time is the initialization time plus the average time to solve the model instance multiplied by the number of points generated.

We emphasize the novelty of parametric tangential approximation. First, there are no superfluous computations like those of other methods. Each Lagrangian solution either generates a new point on the envelope function or it fathoms a gap region. Our parametric tangential approximation algorithm is optimal in the sense that it requires the minimum number of Lagrangian optimizations to generate the complete convex envelope. Second, there is no a priori specification of target probabilities except for $\beta = 0$ and $\beta = 1$ and all envelope points are generated a posteriori.

6.4 Pre-processing

Connections between chance constraints and knapsack constraints have been exploited by numerous authors (e.g., [19, 23, 28]) and there are knapsack properties that can be used for our application. We found the following property useful in reducing the number of indicator variables when solving the CC problem (6.2).

Property 6.2 *If $p_s > 1 - \beta$, then $\delta_s = 1$ in every feasible solution.*

A proof is straightforward. If $\delta_s = 0$, then the probability is at most $\sum_{i \neq s} p_i$, which equals $1 - p_s$. We thus require $1 - p_s \geq \beta$, which is equivalent to $p_s \leq 1 - \beta$. We let $\alpha \overset{\text{def}}{=} 1 - \beta$ for notational convenience in the remainder of this section.

If the scenarios are equally likely, then Property 6.2 yields an all-or-nothing situation. If $\alpha < \frac{1}{N_S}$, then all scenarios are forced to be selected; otherwise, no scenario is forced. In practice, the distribution is generally not uniform and there are scenarios that must be selected for sufficiently large β. For example, if there are only 20 scenarios (maybe during model development), then some $p_s \geq 0.05$—in which case the scenario must be selected for $\beta > 0.95$.

Pre-processing with a specified probability includes fixing $\delta_s = 1$ for all forced selections, i.e., for $p_s > \alpha$. Figure 6.4 shows the reduced response space for Example 6.1 with $\beta = 0.5$. The response space has only 16 of the 32 points, and the left endpoint is $(0.75, 60)$, corresponding to setting $\delta_5 = 1$.

In some cases forced selections solve the problem using the following property.

Property 6.3 *Let $\widehat{S}$ be a set of scenarios for which $\delta_s = 1$ for all $s \in \widehat{S}$. Suppose $P(\widehat{S}) = \sum_{s \in \widehat{S}} p_s \geq \beta$. Then, we can fix $\delta_s = 0$ for all $s \notin \widehat{S}$ without loss in optimality.*

We can use these two properties to limit the intervals over which we must search. Let the scenarios be sorted by non-decreasing probability, and suppose $\widehat{S}$ contains all s for which $p_s > \alpha$. Further suppose that k is the smallest index in the set (so $p_{k-1} \leq \alpha$). Combining Properties 6.2 and 6.3, we find that the chance-constraint instance is solved for $\alpha \in [1 - P(\widehat{S}),\ p_k)$. We use this solution to find probability intervals that solve the chance-constraint instance with forced selections. Let $\mathcal{I}_s = \left[\sum_{i=1}^{s-1} p_i,\ p_s\right)$. We have $\mathcal{I}_1 = [0, p_1) \neq \emptyset$ (assuming $p > 0$). Let $\mathcal{A} = \cup_s \mathcal{I}_s$,

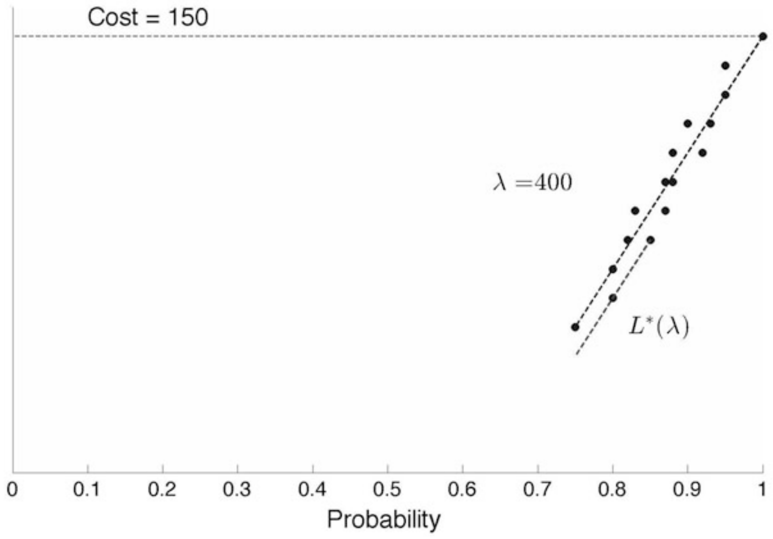


Fig. 6.4 Reduced response space for Example 6.1, fixing the selection of scenario 5

so that the chance-constraint instance is solved by forced selections if, and only if, $\alpha \in \mathcal{A}$.

Example 6.2 This is to demonstrate that solved intervals can be separated by empty ones.

s	$\sum_{i=1}^{s-1} p_i$	p_s	$\mathcal{I}_s$
1	0	0.05	$\neq\emptyset$
2	0.05	0.10	$\neq\emptyset$
3	0.15	0.10	$=\emptyset$
4	0.25	0.20	$=\emptyset$
5	0.45	0.55	$\neq\emptyset$

Thus, $\mathcal{A} = [0, 0.10) \cup [0.45, 0.55)$.

In summary, we find $\mathcal{A}$ for parametric processing, which is a non-empty union of intervals, and we intersect it with

$$[\alpha^{\min} = 1 - \beta^{\max},\ \alpha^{\max} = 1 - \beta^{\min}].$$

This process is used in the parametric version of the Lagrange multiplier search by fathoming intervals contained in the forced-selection interval. We can force at the outset selections for scenarios such that $p_s > \alpha^{\max}$. Although the conditions are simple to establish, they can have a significant impact.

We emphasize that our algorithmic goal is to provide an advanced understanding of the chance-constraint model. An analyst needs to know *why* some scenarios are selected while others are not—is it due to economic benefit or are they restricted by other constraints? Can the analyst deduce some scenario dependence—e.g., $\delta_s = 1 \rightarrow \delta_t = 0$. Such analysis could occur during a debugging stage or during data development, but in a mature model our analysis could add clarity concerning what the scenario constraints mean and how they relate to the rest of the model.

6.5 Gap Closing

The procedures of the previous section provide the lower convex envelope for RS, denoted F^*; however, analysts may benefit from seeing more points in the space even if they are not on this frontier. We seek additional information about solutions in gap regions by fixing δ, thus providing points above the envelope function. It is natural for a good analyst to ask, "How close are suboptimal solutions?" (which may have other favorable properties to present options for management).

Consider a gap region $[b^L, b^U]$ with $\beta \in (b^L, b^U)$ and $g(\beta) > \tau^{\text{gap}}$ (a tolerance). We present some heuristics to search for a feasible solution, (b, z), where b is in the interior of the gap region—i.e., $b \in [\beta, b^U)$. Let δ^L and δ^U be optimal selection values associated with the endpoints, and define the partition of scenarios:

$$\begin{aligned} S^{00} &= \{s : \delta_s^L = 0,\ \delta_s^U = 0\} \\ S^{01} &= \{s : \delta_s^L = 0,\ \delta_s^U = 1\} \\ S^{10} &= \{s : \delta_s^L = 1,\ \delta_s^U = 0\} \\ S^{11} &= \{s : \delta_s^L = 1,\ \delta_s^U = 1\}. \end{aligned}$$

We must have $\left|S^{01} \cup S^{10}\right| > 0$ because the two solutions differ. Our first heuristic is called *restricted flipping* and it fixes values in $S^{00} \cup S^{11}$ and flip values in $S^{01} \cup S^{10}$, moving from b^L to β and/or moving from b^U to β.

If $\left|S^{01}\right| = 1$, restricted flipping takes us from b^L to b^U, so suppose $\left|S^{01}\right| > 1$. We then select a sequence to flip until the total probability, b, is at least β. If $b = b^U$, then this flipping sequence fails, and we order the sequence by probability, leaving the minimum value for last. If that last flip is necessary to reach β—i.e., if $\sum_{s \in S} p_s < \beta$ for all $S \underset{\neq}{\subset} S^{01}$, restricted flipping fails. We otherwise fix $\delta_s = 1$ for those flipped. Those not flipped are fixed at 0, their current value. This gives us a new point in the response space, $(b, z^*(\delta))$.

Initialize $z^{\text{Best}} = z^U$ and $b^{\text{Best}} = b^U$. If $z^*(\delta) < z^{\text{Best}}$, then update $z^{\text{Best}} = z^*(\delta)$ and $b^{\text{Best}} = p\delta$. Test for termination using a gap tolerance, $g(\beta) \leq \tau^{\text{gap}}$, and a probability tolerance: $\left|b^{\text{Best}} - \beta\right| \leq \tau^{\text{prob}}$. If we do not terminate, flip from b^U, fixing $\delta_s = 0$ for a sequence of $s \in S^{01}$, ordered by probability, until $p\delta < \beta$. Let b be the probability just before reaching this condition. As above, we must have

Table 6.1 Points in the response space and their associated scenario selections

	Subset selected	Probability	Cost
	S	$\sum_{s\in S} p_s$	$\sum_{s\in S} c_s$
1	{5}	0.750	50
2	{1, 5}	0.800	60
3	{2, 5}	0.800	70
4	{3, 5}	0.820	80
5	{4, 5}	0.830	90
6	{1, 2, 5}	0.850	80
7	{1, 3, 5}	0.870	90
8	{2, 3, 5}	0.870	100
9	{1, 4, 5}	0.880	100
10	{2, 4, 5}	0.880	110
11	{3, 4, 5}	0.900	120
12	{1, 2, 3, 5}	0.920	110
13	{1, 2, 4, 5}	0.930	120
14	{1, 3, 4, 5}	0.950	130
15	{2, 3, 4, 5}	0.950	140
16	{1, 2, 3, 4, 5}	1.000	150

Table 6.2 Illustration of calculations for points in the interval (0.85, 0.92)

b	z^*	S	
0.85	80	{1, 2, 5}	$S^{10} = \emptyset,\ S^{11} = \{1, 2, 5\}$
0.87	90	{1, 3, 5}	$L^*(\lambda^*) + \lambda^* b = 90$
0.88	100	{1, 4, 5}	$L^*(\lambda^*) + \lambda^* b = 95$
0.90	120	N/A[a]	$L^*(\lambda^*) + \lambda^* b = 105$
0.92	110	{1, 2, 4, 5}	$S^{01} = \{4\},\ S^{00} = \{3\}$

[a] $f^*(0.90) = f^*(0.92)$

$b > b^L$ to obtain a new point in the response space, and if that is the case, then compute $z^*(\delta)$ and apply the same tests to update z^{Best} and terminate.

Table 6.1 enumerates the 16 points of the reduced response space from Fig. 6.4, plotted in Fig. 6.5 (spread out to see the points more distinctly). The envelope function, F^*, is the piece-wise linear function, with $B = [0.75, 1]$. We can restrict $\beta \in B$ because the parametric range is $\alpha \in [0.05, 0.25)$.

Suppose we want to close the gap in the interval $(0.85, 0.92)$, with $\lambda^* = (110 - 80)/(0.92 - 0.85) = 428.57$. The two circled points are the only non-dominated, feasible points with a better solution than $z^U = 110$ as documented in Table 6.2.

Restricted flipping fails because once we fix the common selections, $\delta_1 = \delta_2 = \delta_5 = 1$, only $\delta_4 = 1$ flips from b^L, which gets us to b^U; and, flipping $\delta_4 = 0$ from b^U gets us to b^L. However, if we relax fixing all common selections, we can reach (0.88, 100) from b^U by flipping δ_2, resulting in $\delta = (1, 0, 0, 1, 1)$. This is the optimal value, but all we can confirm is that the best feasible solution, with $z = 100$, has relative gap value $g(0.88) = 1 - 95/100 = 0.05$. This is a significant

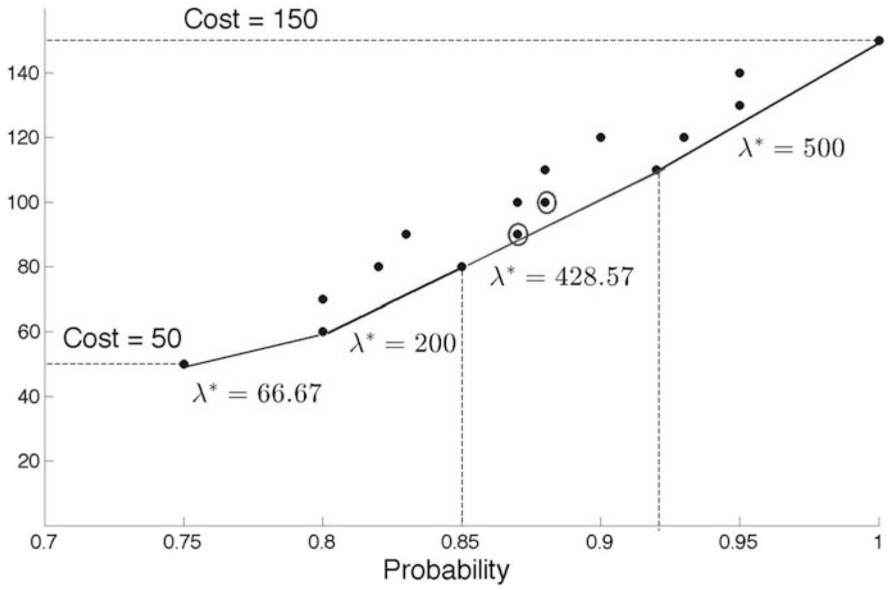


Fig. 6.5 Reduced response space of Example 6.1 (c.f., Fig. 6.4) after our parametric search algorithm finds $F^*(\beta)$ for $\beta \geq 0.75$

improvement over the original value $g(0.88) = 1 - 95/110 = 0.1364$, and it is the best we can do.

We also cannot reach $(0.87, 90)$ by restricted flipping because $S = \{1, 3, 5\} \not\subset S^U = S^{01} \cup S^{11}$ implies that we cannot flip from (b^U, z^U). Similarly, $S \not\subset S^L = S^{10} \cup S^{11}$ means we cannot flip from (b^L, z^L). However, if we relax fixing common exclusions, we can then flip $\delta_3 = 1$ and consider flipping others in S^L. Heuristics that relax fixing common exclusions remain as future research.

It is in general inexpensive and potentially valuable to consider flipping only scenarios that are selected by one endpoint and not the other. Contrary to the particular example, common selections may be a form of evidence, and there is little computational cost to try it first. That is, we need not solve any new minimization problem to discover if this flipping generates a new probability; we simply loop through a sorted list of probabilities. If this fails, then relaxed flipping is tried, which may generate a new feasible response space point, (b, z) with $b \geq \beta$ and $z < z^U$. If this is the case, then we decrease the gap by setting $z^{\text{Best}} = z$.

Suppose restricted flipping fails to yield an acceptable solution—i.e., the best solution is not within tolerances: $\left|b^{\text{Best}} - \beta\right| > \tau^{\text{prob}}$ or $g(\beta) > \tau^{\text{gap}}$. We then begin to enlarge the space of candidates to flip. For parametric chance constraint we use gap-closing heuristics to generate additional points in RS. The purpose is to learn about the cost-probability trade-offs.

We applied our methods to the three models as described in the Supplementary Material attached to the electronic version of this chapter. Two of the models capture

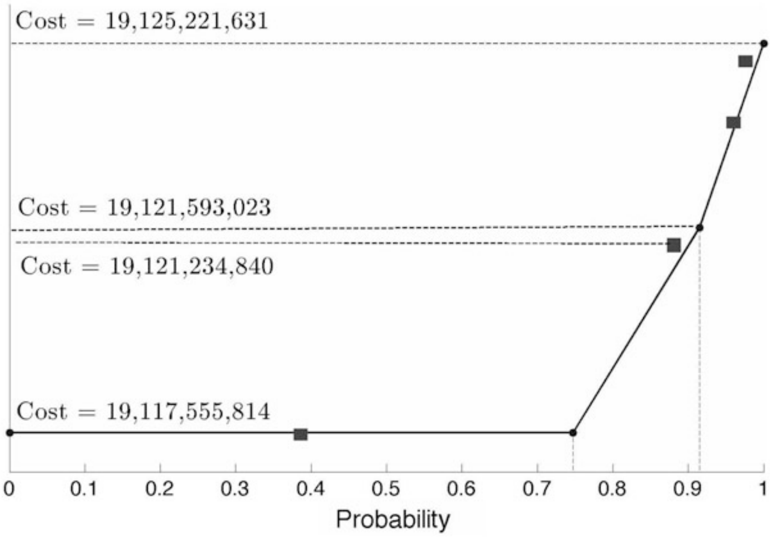


Fig. 6.6 Adding response space points to the 10-scenario Midwest GEP model

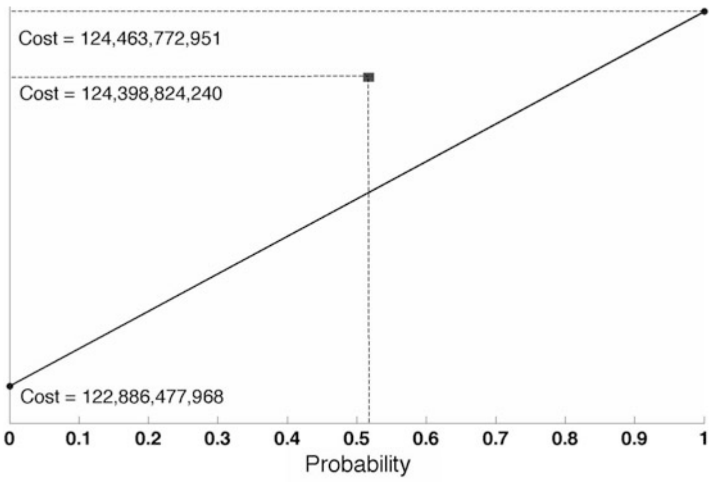


Fig. 6.7 Adding to the response space for the 70-scenario Korean GEP model

features of an electricity generation expansion planning (GEP) problem, and the third model is a network flow, capacity-planning model. We refer to the models as Midwest GEP, Korean GEP, and Network Flow, respectively. Figures 6.6, 6.7, and 6.8 show response space points added to each of the instances by flipping selections of each upper endpoint ($\delta_s^U = 1$) not selected in the interval's lower endpoint ($\delta_s^L = 0$).

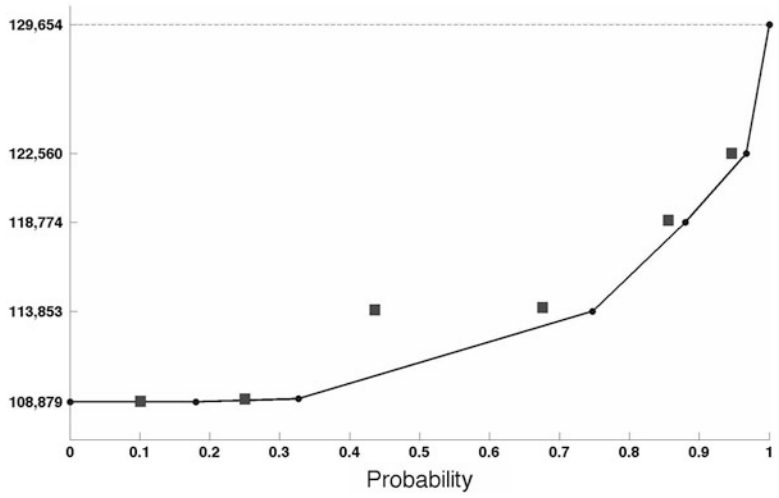


Fig. 6.8 Adding response space points to the 10-scenario network flow model

The one point added to RS in the Korean model (Fig. 6.7) tells us $f^*(0.512) \leq 124,398,824,240$, which is a slight improvement over the upper endpoint, $z^U = 124,463,772,951$. The relative gap is reduced by an order of magnitude to 0.000111 (from 0.006237).

Here is the algorithm to generate additional points after F^* is constructed. First, form the list of gap intervals, $\{(i, w_i, m_i)\}$, where $w_i = b_i - b_{i-1}$ is the width, and $m_i = \frac{1}{2}(b_i + b_{i-1})$ is the midpoint. Sort this list by width and drop intervals with $w_i \leq 2\tau^{\text{prob}}$.

If the number of (sufficiently wide) intervals is greater than a specified maximum, we simply drop the last few intervals. If we have fewer than the specified number of intervals, then we use the sort-order to split the first (i.e., widest) interval:

$$(i, w, m) \rightarrow \left(i,\ \frac{1}{2}w,\ m - \frac{1}{4}w\right),\ \left(i,\ \frac{1}{2}w,\ m + \frac{1}{4}w\right).$$

Note that the original index is retained when splitting. We then re-sort until we either reach the maximum number of points specified or the split would make the width too small—i.e., stop once $w \leq 4\tau^{\text{prob}}$.

We have in the end abscissa points, $\{m_k\}$, plus associated widths and gap-region indexes, for $k = 1, \ldots, K$, where K is within the specified maximum and $w_k > 2\tau^{\text{prob}}$. For each k, initialize selections from z_{i_k}, the upper endpoint of the i_k-th gap region, and flip $s_1, s_2, \ldots$ (in probability-order) until reaching $b = \sum_{j=1}^{\nu} p_{s_j} \geq m_k$ and $b - p_{s_\nu} < m_k$. If this is reached before $b^L = b_{i_k - 1}$, we then compute $z^*(\delta)$ to obtain the new RS point, $(b, z^*(\delta))$. Otherwise, we simply go to the next interval.

We can combine the gap intervals with pre-processing intervals of the form $\cup_k(\bar{p}_{s_k}, \bar{a}_k]$, where $\bar{v} \overset{\text{def}}{=} 1 - v$ for any $v \in [0, 1]$. We know $b = \frac{1}{2}(\bar{a}_k + \bar{p}_{s_k})$ is solved by a forced selection (that fixes δ). If the forced selection has $b = \bar{a}_k$ and the selection is already an endpoint of a gap interval, then the solver regenerates b_i and we do not obtain a new point. However, if $\min_i |\bar{a}_k - b_i| > \tau^{\text{prob}}$, we then compute $(b = p\delta,\ z^*(\delta))$ for δ corresponding to the forced selections by solving for $z^*(\delta)$.

A major advantage of doing this is that $f^*(b) = z^*(\delta)$. This optimality cannot be guaranteed with a gap-closing heuristic, like restricted flipping. On the other hand, an advantage of using gap intervals to determine the abscissa values is that we have a more distributed collection of response space points, which gives a sense of how the chance constraint affects the solution. Further experimentation with this avenue of solution insights from a response space is warranted.

6.6 Some Pitfalls to Consider

Our implementation has identified pitfalls that merit some attention. For convenience, assume $z > 0$ for all $(b, z) \in \text{RS}$, so relative cost values can be used without absolute values.

6.6.1 Tolerance Relations

We can increase the optimality tolerance, τ^{opt}, to reduce the time to minimize the Lagrangian. The effect of this change depends on the solver [11] and relates to two tolerances that can be set as options in our Python program:

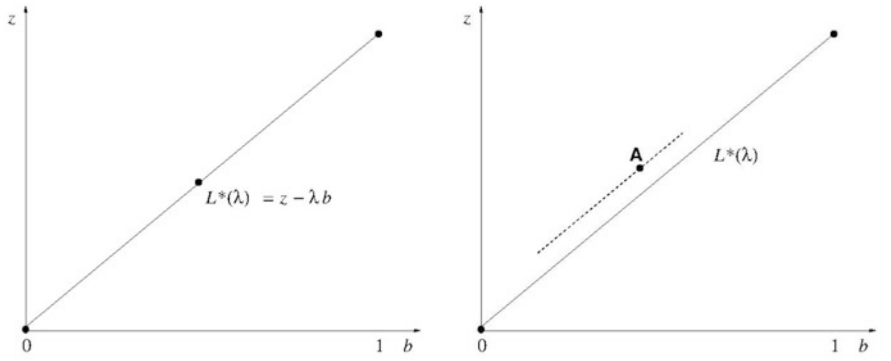
- τ^{prob}: two probabilities, b and b', are equal if $|b - b'| \le \tau^{\text{prob}}$.
- τ^{gap}: (b, z) is acceptable (i.e., z is sufficiently close to $f^*(b)$) if the gap between z and the Lagrangian bound, $F^*(b) = L^*(\lambda) + \lambda b$, satisfies $1 - F^*(b)/z \le \tau^{\text{gap}}$. Recall we use this when exploring gap regions with $z = z^U$.

We cannot be sure exactly what near-optimality means, but we can suppose $\mathcal{L}$ is a lower bound on the (unknown) optimum because $\mathcal{L}(\lambda) \le L^*(\lambda) \le z - \lambda b$. The solver terminates if

$$\frac{z - \lambda b - \mathcal{L}(\lambda)}{\mathcal{L}(\lambda)} \le \tau^{\text{opt}}.$$

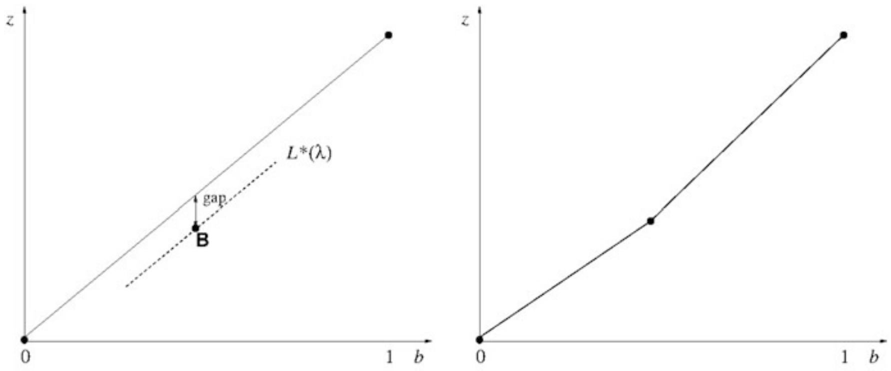
Equivalently (as implemented), $z - \lambda b \le L^*(\lambda)(1 + \tau^{\text{opt}})$.

Figure 6.9a shows an alternative optimum for the Lagrangian with $\lambda = z_1 - z_0$, which is the slope of line segment joining the two initial points, $(0, z_0)$ and $(1, z_1)$. A solver should begin by checking the optimality of the endpoint that is still resident, but some will begin anew. That is the only way the alternative solution



(a) Alternative Optimum of $L^*(\lambda)$ (b) Near-Optimal Solution of Lagrangian

Fig. 6.9 Inexact alternative Lagrange optimum in $(b^L + \tau^{\text{prob}},\ b^U - \tau^{\text{prob}})$. (**a**) Alternative optimum of $L^*(\lambda)$. (**b**) Near-optimal solution of Lagrangian



(a) $z < z^L + \lambda(b - b^L)$ (b) Update

Fig. 6.10 Accepting an inexact alternative Lagrange optimum as a new RS point. (**a**) $z < z^L + \lambda(b - b^L)$. (**b**) Update

would be reached. If this occurs, then we save the generated RS point because b is not within tolerance of either endpoint—i.e., $b \in (b^L + \tau^{\text{prob}},\ b^U - \tau^{\text{prob}})$. However, the interval is fathomed because there are no points below the line.

Figure 6.9b shows a situation where a new point is generated by being within (relative) tolerance of optimality: $z - \lambda b \leq L^*(\lambda)(1 + \tau^{\text{opt}})$. With cost above the line (i.e., $z > z^L + \lambda(b - b^L)$), depicted as point A, we save (b, z), but we fathom the interval, as in the case of the exact optimum.

Figure 6.10a shows the near-optimum of the Lagrangian below the line, labeled point B. We consider B to be a new point if $|b - b^L| > \tau^{\text{prob}}$, $|b - b^U| > \tau^{\text{prob}}$, and the Lagrangian gap exceeds tolerance: $1 - (z^L + \lambda(b - b^L))/2 > \tau^{\text{gap}}$. We otherwise treat B the same as if (b, z) is above the line.

The value τ^{opt} pertains to the Lagrangian optimum, whereas τ^{gap} and τ^{prob} pertain to the cost and probability, respectively. The action to treat (b, z) as a new point (splitting the interval, rather than fathoming it) depends on how these tolerances relate. Their meanings are different and can cause anomalous behavior, even if $\tau^{\text{opt}} = \tau^{\text{gap}}$. See [11] for elaboration and examples of how tolerances can interact. A pitfall to avoid is setting τ^{opt} in a way that is inconsistent with other tolerances. More analysis of the interaction among tolerances, including some solver tolerances, is an avenue for further research.

6.6.2 Measuring Probability

Using the indicator variable in (6.1), we find that it is possible to have $\delta_s = 0$ but still have scenario-s constraints satisfied. Therefore, $p\delta$ is not an exact measure of the probability of being feasible. Letting Q_s denote the feasible values of (x, y_s), the real chance constraint is $\sum_s \Pr\big((x, y_s) \in Q_s \mid s\big)\, p_s \geq \beta$. Our probability value is thus an underestimate:

$$p\delta = \sum_{s=1}^{N_S} p_s \delta_s \leq \sum_{s=1}^{N_S} \Pr\big((x, y_s) \in Q_s \mid s)\big)\, p_s.$$

To see this, consider x restricted to satisfy scenario s. That is the case if $\delta_s = 1$, and we have in this case that $\Pr\big((x, y_s) \in Q_s \mid s)\big) = 1$. However, x can satisfy the constraint when $\delta_s = 0$, so in general $\delta_s \leq \Pr\big((x, y_s) \in Q_s \mid s)\big)$. Being an underestimate means that the chance-constraint model is conservative because $p\delta \geq \beta$ implies that the true chance constraint is satisfied.

A modeler can add violation variables to measure actual violation and enforce the converse: $\delta_s = 0$ implies scenario s is violated. Let the auxiliary variable v_i^s measure violation of the ith constraint in scenario s:

$$\textstyle\sum_j a_{ij}^s x_j^s - b_i^s \geq -v_i^s, \qquad\qquad 0 \leq v_i^s \leq (1 - \delta_s)M.$$

The scenario constraint is satisfied if $v^s = 0$, which is forced by $\delta_s = 1$ (as in first model).

For $\delta_s = 0$, the solver could produce a solution with $v^s \neq 0$ even if there is no violation as long as the cost is not greater than the minimum. In fact, if an interior solution is computed, then both the surplus variable and v_i^s are positive if the ith constraint is over satisfied in *some* optimal solution. To ensure $v \neq 0$ except when necessary, define a nuisance cost, $\varepsilon > 0$, and add $\varepsilon \sum_{i,s} v_i^s$ to the objective. If there are alternative optima, then favor is given to $v = 0$. Notice that ε must be small enough to preserve minimality of the original cost. Then, an optimal solution will have $v_i^s = \max\big\{0,\ b_i^s - \sum_j a_{ij}^s x_j^s\big\}$, which equals the amount of violation of the ith constraint.

For some models, like our Network Flow, setting $\delta_s = 0$ has no effect on the objective function, but in other models, this could mislead an analyst who uses scenario violation to support one decision over another. Moreover, the exact form is important to answer questions like, "What is the impact of having a chance constraint?" The level of violation may be of interest, which is not obtained in the first (underestimate) model. Further analysis of the level of constraint violation can be supported by taking a large number of additional samples for the purpose of a better estimate of the actual probability of violation (see, e.g., [24]).

6.6.3 When It Is Infeasible to Select All Scenarios

We have assumed for notational convenience that it is feasible to select all scenarios—i.e., $\delta_s = 1$ for all $s \in S$. This may not be the case, and we might need to find

$$P^{\max} = \max_{\delta \in \{0,1\}^{N_S}} \{p\delta : \exists x \in X \ni Y_s(x, \delta) \neq \emptyset\}. \qquad (6.12)$$

If we seek a solution for $\beta > P^{\max}$, then the specified chance-constraint instance is infeasible. We otherwise need to find λ such that $L^*(\lambda)$ yields the RS point, $(P^{\max}, z^*(\delta))$ for some optimal δ (not necessarily the selections computed if we only maximized $p\delta$ without regard for cost). Tangential approximation is initialized with this point to bracket the search in this case.

Here is how we find such a λ. Let Z be the cost for the computed solution of $P^{\max}$, and consider $\lambda > Z/\tau \geq z^*(\delta)/\tau$, where τ is sufficiently small to ensure that $L^*(\lambda) = z^*(\delta) - \lambda p\delta \Rightarrow p\delta \geq P^{\max} - \tau^{\text{prob}}$. To help intuition, consider $z^*(\delta) = c\delta$. Then, $c_s - \lambda p_s < 0$ for all s for $\lambda > \max_s \{c_s/p_s\}$. This means $\delta_s = 1$ unless it is not feasible to select scenario s. Minimization of the Lagrangian takes care of the trade-off, making $p\delta$ a maximum over all feasible selections.

6.6.4 Low Probabilities

The general range for the parametric tangential approximation algorithm is $[\beta^{\min}, \beta^{\max}]$. If $\beta^{\min} = 0$ is infeasible, then our code terminates, as this means the original model instance is infeasible without the scenario constraints. One usually imagines low values of β as being of little interest, but we assert that this is a pitfall because low values of β can also provide some information, starting with $\beta = 0$:

- What is my minimum cost with no scenario compliance?

- How much computational time is due to adding the joint chance constraint? (That is, what is a baseline for how much computational time to expect as we search for optimal multiplier values, which adds N_S binary variables to the model?)

The full range is useful for debugging a model and testing its validity even before analysis support.

6.7 Summary and Conclusions

Each Lagrangian solution generates a point that is an exact optimum for the Lagrangian problem (6.3) and its associated parametric program. The piece-wise linear function connecting those points is the envelope function that yields the Lagrangian bound. We have shown that the complete parametric tangential approximation minimizes the number of Lagrangian solutions to generate the convex envelope. Each iteration of parametric tangential approximation yields a new RS point that either confirms a gap region (immediately, as the resident solution is optimal) or causes it to shrink or split. The terminal succession of RS points, $\{(b_i, z_i)\}_{i=0}^N$, covers $[0, P^{\max}]$ with $b_0 = 0 < b_1 < \cdots < b_N = P^{\max}$.

Tangential approximation for a single β was introduced decades ago, and it is not necessarily an optimal algorithm. For example, bisection could reach the one optimal multiplier faster. However, our extension to a complete parametric search is optimal in that it minimizes the number of Lagrangian solutions needed for complete parametrization.

Each interval (b_{i-1}, b_i) contains Lagrangian duality gaps, where there may be a solution above (or on) the convex envelope function for which $z < z_U$. Gap intervals can be explored by a variety of heuristics. We presented one approach, called restricted flipping, that seeks a better feasible solution for $\beta \in (b_{i-1}, b_i)$ than z_i by flipping optimal values of δ that differ between the endpoint solutions. Once δ is specified, we compute the RS point, $(p\delta,\ z^*(\delta))$. We presented a heuristic for choosing probability values, based on the midpoints of intervals that have been sorted by $b_i - b_{i-1}$.

Each Lagrangian solution to a general problem, $L(x) = f(x) - \lambda g(x)$, solves two programs:

$$\min\ f(x) \colon g(x) \geq b \overset{\text{def}}{=} g(x^*) \text{ and } \max\ g(x) \colon f(x) \leq c \overset{\text{def}}{=} f(x^*)$$

for any $x^* \in \operatorname{argmin} L(x)$. Varying $\lambda \in [0, \infty)$ yields parametric solutions $\{b, f^*(b)\}_b$ and $\{g^*(c), c\}_c$. They are precisely the same set in RS. These are also equivalent to using a weighted sum of the bi-criteria program:

$$\min\ \alpha f(x) + (1 - \alpha)(-g(x)).$$

Varying $\alpha \in (0, 1)$ generates Pareto-optimal points. Each α is equivalent to the Lagrangian with $\lambda = 1/(1 - \alpha)$ (so $\min\ L = f - \lambda g \leftrightarrow \min\ \alpha f + (1 - \alpha)g$). This generates the same portion of the efficient frontier. The weighted sum fails to generate some Pareto-optimal points—viz., non-convex segments of the frontier. This is precisely the Lagrangian duality gap.

One reason to point this out is that there has been a vast literature on Lagrangian duality and bi-criteria programming (separately and jointly) in the last several decades. Most of it assumes a special structure, notably convexity or separability, which we do not. Moreover, we go beyond the algorithmics, focusing on the use of the convex envelope to support analysis. Our gap-resolution method demonstrates an effective computational approach not only to reduce the gap, but also to provide a better understanding of the cost-probability trade-off, including sub-optimal solutions.

Most importantly, Everett's Generalized Lagrange Multiplier Method advances analysis with efficient computation implemented as an open-source extension of PySP. Output includes tables of RS points, which can be used by software to provide graphical support. The goal is insight into the trade-off between cost and scenario-satisfaction probability. It is for that reason that we provide additional code to explore RS, not only for near-optimal solutions, but also for gaining information about alternative (sub-optimal) solutions. A motive for such exploration is to consider alternative solutions that have properties not represented in the model—e.g., ease of policy implementation.

Our current and future research extends our work to multiple chance constraints. Our foundation is Everett's paper and its derivatives, notably [2] and [13, 14].

Acknowledgments The research in this article was supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Applied Mathematics program under contract number KJ0401000 through the Project "Multifaceted Mathematics for Complex Energy Systems".

References

1. S. Ahmed, A. Shapiro, Chapter 12: solving chance-constrained stochastic programs via sampling and integer programming, in *TutORials in Operations Research*, ed. by Z.-L. Chen, S. Raghavan (INFORMS, Catonsville, 2008), pp. 261–269
2. R. Brooks, A. Geoffrion, Finding Everett's Lagrange multipliers by linear programming. Oper. Res. **14**(6), 1149–1153 (1966)
3. A. Charnes, W.W. Cooper, Systems evaluation and repricing theorems. Manag. Sci. **9**(1), 33–49 (1962)
4. J.P. Evans, F.J. Gould, S.M. Howe, A note on extended GLM. Oper. Res. **19**(4), 1079–1080 (1971)
5. H. Everett, III, Generalized Lagrange multiplier method for solving problems of optimum allocation of resources. Oper. Res. **11**(3), 399–417 (1963)
6. A.M. Geoffrion, The purpose of mathematical programming is insight, not numbers. Interfaces **7**(1), 81–92 (1976)
7. F.J. Gould, Extensions of Lagrange multipliers in nonlinear programming. SIAM J. Appl. Math. **17**(6), 1280–1297 (1969)
8. H.J. Greenberg, Lagrangian duality gaps: Their source and resolution. Technical Report CP-69005, Southern Methodist University, Dallas (1969). http://math.ucdenver.edu/~hgreenbe/pubs.shtml
9. H.J. Greenberg, Bounding nonconvex programs by conjugates. Oper. Res. **21**(1), 346–348 (1973)
10. H.J. Greenberg, The one dimensional generalized Lagrange multiplier problem. Oper. Res. **25**(2), 338–345 (1977)
11. H.J. Greenberg, Supplement: tolerances, in [Holder, A. (ed.), *Mathematical Programming Glossary*. INFORMS Comput. Soc. (2014)]. Posted 2003. Also appears at *Optimization Online*. http://www.optimization-online.org/DB_HTML/2012/05/3486.html
12. H.J. Greenberg, Supplement: myths and counterexamples in mathematical programming, in [A. Holder (ed.), *Mathematical Programming Glossary*. INFORMS Comput. Soc. (2014)]. Posted 2010
13. H.J. Greenberg, Supplement: Lagrangian saddle point equivalence, in [A. Holder (ed.), *Mathematical Programming Glossary*. INFORMS Comput. Soc. (2014)]. Transcribed from 1969 Course Notes
14. H.J. Greenberg, Supplement: response space, in [A. Holder (ed.), *Mathematical Programming Glossary*. INFORMS Comput. Soc. (2014)]. Transcribed from 1969 Course Notes
15. H.J. Greenberg, T. Robbins, Finding Everett's Lagrange multipliers by generalized linear programming. Technical Report CP-70008, Southern Methodist University, Dallas, 1970. http://math.ucdenver.edu/~hgreenbe/pubs.shtml
16. W.E. Hart, J.P. Watson, D.L. Woodruff, Python optimization modeling objects (Pyomo). Math. Program. Comput. **3**(3), 219–260 (2011)
17. W.E. Hart, C. Laird, J.-P. Watson, D.L. Woodruff, *Pyomo—Optimization Modeling in Python* (Springer, Berlin, 2012)
18. A. Holder (ed.), *Mathematical Programming Glossary*. INFORMS Comput. Soc. (2014). http://glossary.computing.society.informs.org
19. S. Küçukyavuz, On mixing sets arising in chance-constrained programming. Math. Program. **132**(1–2), 31–56 (2012). ISSN 0025-5610. https://doi.org/10.1007/s10107-010-0385-3
20. M.A. Lejeune, S. Shen, Multi-objective probabilistically constrained programming with variable risk: new models and applications. Eur. J. Oper. Res. **252**(2), 522–539 (2016)
21. J. Luedtke, An integer programming and decomposition approach to general chance-constrained mathematical programs, in *Integer Programming and Combinatorial Optimization*, ed. by F. Eisenbrand, F. Shepherd. Lecture Notes in Computer Science, vol. 6080 (Springer, Berlin, 2010), pp. 271–284. ISBN: 978-3-642-13035-9

22. J. Luedtke, A branch-and-cut decomposition algorithm for solving chance-constrained mathematical programs with finite support. Math. Program. A **146**, 219–244 (2014)
23. J. Luedtke, S. Ahmed, G.L. Nemhauser, An integer programming approach for linear programs with probabilistic constraints. Math. Program. **122**(2), 247–272 (2010). ISSN: 0025-5610. https://doi.org/10.1007/s10107-008-0247-4
24. A. Nemirovski, A. Shapiro, Convex approximations of chance constrained programs. SIAM J. Optim. **17**(4), 969–996 (2006)
25. A. Prekopa, Probabilistic programming, in *Handbooks in Operations Research and Management Science, Volume 10: Stochastic Programming*, ed. by A. Ruszczyński, A. Shapiro (Elsevier, Amsterdam, 2003)
26. T. Rengarajan, D. P. Morton, Estimating the efficient frontier of a probabilistic bicriteria model, in *Proceedings of the 2009 Winter Simulation Conference*, ed. by M.D. Rossetti, R.R. Hill, B. Johansson, A. Dunkin, R.G. Ingalls (2009), pp. 494–504
27. T. Rengarajan, N. Dimitrov, D.P. Morton, Convex approximations of a probabilistic bicriteria model with disruptions. INFORMS J. Comput. **25**(1), 147–160 (2013)
28. A. Ruszczyński, Probabilistic programming with discrete distributions and precedence constrained knapsack polyhedra. Math. Program. **93**(2), 195–215 (2002)
29. J.F. Shapiro, Generalized Lagrange multipliers in integer programming. Oper. Res. **19**(1), 68–76 (1971)
30. S. Shen, Using integer programming for balancing return and risk in problems with individual chance constraints. Comput. Oper. Res. **49**, 59–70 (2014)
31. J.-P. Watson, R.J.-B. Wets, D.L. Woodruff, Scalable heuristics for a class of chance-constrained stochastic programs. INFORMS J. Comput. **22**(4), 543–554 (2010). ISSN: 1526-5528. https://doi.org/10.1287/ijoc.1090.0372
32. J.-P. Watson, D.L. Woodruff, W.E. Hart, Modeling and solving stochastic programs in Python. Math. Program. Comput. **4**(2), 109–149 (2012)
33. W.B. Widhelm, Geometric interpretation of generalized Lagrangian multiplier search procedures in the payoff space. Oper. Res. **28**(3), 822–827 (1980)
34. P.L. Yu, M. Zeleny, Linear multiparametric programming by multicriteria simplex method. Manag. Sci. **23**(2), 159–170 (1976)