Noname manuscript No.
(will be inserted by the editor)

Statistical performance of subgradient step-size update rules in Lagrangian relaxations of chance-constrained optimization models

Charlotte Ritter · Bismark Singh*

Received: date / Accepted: date

Abstract Lagrangian relaxation schemes, coupled with a subgradient procedure, are frequently employed to solve chance-constrained optimization models. The subgradient procedure typically relies on a step-size update rule. Although there is extensive research on the properties of these step-size update rules, there is little consensus on which rules are most suited in practice. This is especially so when the underlying model is a computationally challenging instance of a chance-constrained program. To close this gap, we seek to determine whether a single step-size rule can be statistically guaranteed to perform better than others. We couple the Lagrangian procedure with three strategies to identify lower bounds for two-stage chance-constrained programs. To this end, we consider two instances of such models that differ in the presence of binary variables in the second-stage. With a series of computational experiments, we demonstrate—in marked contrast to existing theoretical results—that no significant statistical differences in terms of optimality gaps can be detected between six well-known step-size update rules. Despite this, our results demonstrate that a Lagrangian procedure does provide computational benefit over a naive solution method—regardless of the underlying step-size update rule.

Keywords Chance constraints · Stochastic optimization · Lagrangian decomposition · Subgradient · Clustering

Notation

Indices and Sets

$\omega \in \Omega$	set of scenarios; $\{\omega_1, ..., \omega_{\lvert\Omega\rvert}\}$
$t \in T$	set of time periods; $\{1, 2, \ldots, \lvert T\rvert\}$

Parameters

$R_t \in \mathbb{R}^+$	reward received at time t
$C_t^\omega \in \mathbb{R}^+$	cost incurred at time t in scenario ω
$p^\omega \in [0, 1]$	probability of scenario ω
$u_t^\omega \in \mathbb{R}^+$	a realization of uncertainty revealed at time t in scenario ω
$x_t \in \mathbb{R}$	first-stage decision that provides a reward at time t
$y_t^\omega \in \mathbb{R}$	second-stage decision that results in a cost at time t in scenario ω
$\lambda \in \mathbb{R}^+$	a Lagrange multiplier
$\Delta \in \mathbb{R}^+$	a step size to update λ
$\gamma \in \mathbb{R}$	a subgradient

C. Ritter, B. Singh*
Department of Mathematics; Department of Data Science
Friedrich-Alexander-Universität Erlangen-Nürnberg
Erlangen, 91058 Germany
* corresponding author
E-mail: bismark.singh@fau.de

$\zeta \in \mathbb{R}^+$	difference between successive λ values in Algorithm 1
$\psi \in \mathbb{R}^+$	a threshold for the optimality gap in Algorithm 1
$\iota \in \mathbb{R}^+$	a threshold for ζ in Algorithm 1
iter	maximum number of iterations in Algorithm 1-3
time	maximum time limit for Algorithm 1-3 [seconds]
$\rho \in \mathbb{R}^+$	regularization parameter in model (9)
$m \in \mathbb{N}$	number of scenarios used in model (9)
$\hat{x}_t \in \mathbb{R}$	proximal term at time t in model (9)
κ	rate of growth of sample size m in Algorithm 2 and Algorithm 3
$\mathcal{Z}$	optimal objective function value for model (1)
$\underline{\mathcal{Z}}$	a lower bound for model (1)
$\mathcal{Z}_D$	an upper bound for model (1)
$\mathcal{Z}_D(\lambda)$	optimal objective function value for model (3) for given λ
$\underline{\mathcal{Z}_D}(\lambda), \overline{\mathcal{Z}_D}(\lambda)$	lower and upper bounds for model (3) for given λ
$\overline{\overline{\mathcal{Z}}}$	optimal objective function value for model (6)
$\delta \in \mathbb{R}^+$	optimality gap between $\mathcal{Z}_D$ and $\underline{\mathcal{Z}}$; $\delta = \frac{\mathcal{Z}_D - \underline{\mathcal{Z}}}{\mathcal{Z}_D}$.

1 Introduction

Consider the following formulation of a generic two-stage stochastic linear program with chance constraints:

$$\mathcal{Z} = \max_{x,y} \sum_{t \in T} R_t x_t - \mathbb{E}\big[C_t^\omega y_t^\omega\big] \tag{1a}$$

$$\text{s.t. } \mathbb{P}\big(x_t \leq y_t^\omega + u_t^\omega, \forall t \in T\big) \geq 1 - \varepsilon \tag{1b}$$

$$x_t \geq 0, y^\omega \in Y^\omega, \forall t \in T, \omega \in \Omega. \tag{1c}$$

Here, x_t is a first-stage decision that provides a reward R_t, $\forall t \in T$. Following this decision, the uncertainty is realized via a scenario $\omega \in \Omega$. Next, we make a so-called recourse, or second-stage, decision y_t^ω while paying a cost C_t^ω, respectively $\forall t \in T, \omega \in \Omega$. Then, the objective function in equation (1a) seeks to maximize the overall profit by rewarding first-stage decisions and penalizing the expected cost of second-stage decisions. Constraint (1b) is a typical joint chance constraint (JCC); here, u_t^ω is data that is revealed after the uncertainty ω is realized. The JCC ensures that the joint probability of satisfying this inequality is no less than $1 - \varepsilon$, where $\varepsilon > 0$ is a risk threshold. In other words, the JCC ensures that the inequality is satisfied with a high probability for *all* time periods $t \in T$. Constraint (1c) enforces a non-negativity restriction for the x variables, and that the y variables belong to some set Y, $\forall \omega \in \Omega$.

Chance constraints were first studied by Charnes and Cooper [8], and have been actively investigated in the past few decades. One-stage chance-constrained programs, that allow only non-adaptive decisions, are relatively easier to solve as strong integer programming reformulations resulting from strong linear programming (LP) relaxations are available, see, e.g., [3, 15]. Two-stage chance-constrained programs, where the y variables adapt to the scenario that is realized, require specialized algorithms for their solution since direct methods are often computationally intractable. Examples of such methods include decomposition methods [17], heuristics [34], or approximations [30]. Model (1) is an example of a typical two-stage chance-constrained model that requires the use of similarly tailored solution methods. Further, the set Y can include integer variables, and coupling this set with a JCC leads to an even greater requirement of computational effort [29].

The above description of model (1) is generic and includes a few modeling assumptions; these are also reflected in our work. First, we assume a discrete representation of the uncertainty is available via finitely many scenarios in the set Ω. This is possible via a Monte Carlo sampling procedure, or via a sample average approximation (SAA) of the expectation and the chance constraint [18]. For a detailed analysis of this assumption, see [33]. Second, we assume that the entire spectrum of uncertainty is known after the first-stage decision is made. This is a standard representation of two-stage stochastic programs;

in contrast, a multi-stage stochastic model allows the uncertainty to unfold at successive time periods. Multi-stage stochastic models are significantly more difficult to solve than two-stage models, see, e.g., [1]. Third, we assume that the x variables are not directly related across time periods, except via the JCC. This assumption is relatively mild and can be relaxed; however, we maintain it for simplicity of exposition.

The JCC in constraint (1b) can be reformulated as follows.

$$x_t - y_t^\omega \leq u_t^\omega + M_t^\omega z^\omega, \forall t \in T, \omega \in \Omega \tag{2a}$$

$$\sum_{\omega \in \Omega} p^\omega z^\omega \leq \varepsilon \tag{2b}$$

$$z^\omega \in \{0,1\}, \forall \omega \in \Omega. \tag{2c}$$

Here, M_t^ω is large enough such that equation (2a) is vacuous when $z^\omega = 1$. Constraint (2b) is a *knapsack* constraint. For equally likely scenarios, this constraint reduces to $\sum_{\omega \in \Omega} z^\omega \leq \lfloor |\Omega|\varepsilon \rfloor$, where $\lfloor \cdot \rfloor$ rounds its argument down to the nearest integer. With the reformulation of the JCC as in equation (2), model (1) can be directly solved using a commercial mixed-integer programming (MIP) solver; however, the computational effort can be prohibitive.

The aim of this work is to investigate the computational performance of a series of procedures that provide lower and upper bounds for model (1) with the JCC represented as equation (2). For lower bounds of model (1), we summarize, extend, and compare three methods based on regularization, aggregation, and sorting techniques, respectively. For upper bounds of model (1), we use a Lagrangian relaxation procedure that we solve with a subgradient algorithm using six step-size update rules. The theory of these procedures is largely developed; however, as we show later in this work, this theory is not always reflected consistently in computational practice. Specifically, differences arise since in practice we solve models with a given time limit; and, the theoretical "best" bounds are not always achieved in this time limit. To this end, we are interested in the following two questions:

- Q1 Given a computationally intractable chance-constrained program, is there any merit for a bounding procedure that is also computationally challenging?
- Q2 Which bounding procedures are the most effective in practice and when?

Questions Q1 and Q2 are fundamental when facing a choice of bounding procedures for chance-constrained optimization models. We provide a set of guidelines that assist in this choice. With this background, the following are the key contributions of this work.

(i) We bridge the gap between theory and computational practice of six well-known step-size rules for the subgradient method of a Lagrangian relaxation. Our results demonstrate that although there is theoretical evidence for some rules performing better than others, this is not always reflected in practice. Indeed, we show that within a given time limit—even when the time limit is sufficiently large—a single rule cannot be statistically validated as the best performer for a chance-constrained optimization model.

(ii) For instances of model (1) with significant computational challenges, only conservative upper bounds can be derived from a Lagrangian procedure. However, our results demonstrate that even then there is improvement over a naive solution method no matter which step-size rule is employed. For example, with one choice of a step-size rule, over the 32 instances that we consider, the Lagrangian procedure, coupled with a lower bounding scheme, provides an average reduction of 20.9% in optimality gaps.

(iii) For lower bounds of model (1), we investigate three methods based on regularization, aggregation, and sorting techniques, respectively. Over the 32 instances that we consider, the three methods provide an average improvement over the naive solution method in terms of the best feasible solution by 16.2%, 0.5%, and 10.3%, respectively, even when using only half or less of the time consumed by the naive solution method.

(iv) We demonstrate that regularization with aggregated scenarios does not always benefit over regularization with independent identically distributed (i.i.d) scenarios. In the absence of customized aggregation procedures, choosing i.i.d. scenarios for the SAA of model (1) is advisable.

(v) Finally, our results show that good lower and upper bounds used together—even when the bounding procedures are terminated prematurely—result in smaller optimality gaps than those obtained naively. Over the 32 instances that we consider, the average gap between the best lower and upper bounds improves to 30.3% from 37.0% obtained naively.

The structure of the rest of this article is as follows. In Section 2, we study upper bounds using the Lagrangian relaxation procedure with the subgradient algorithm investigating six step-size rules. We compare the six bounds statistically using the best available upper bounds in a time limit. In Section 3, we summarize three lower bounding procedures using methods of regularization, aggregation, and quantiles, respectively. In Section 4, we briefly introduce two existing models that serve as our computational case studies in Section 5. We analyze the lower and upper bounds in Section 5.3 and Section 5.4, respectively; further in Section 5.4, we statistically compare the step-size rules. Finally, we perform an additional set of statistical comparisons—between the naive solution method and the Lagrangian procedure integrated with the best lower bounds—in Section 5.5. We conclude with a discussion in Section 6.

2 Upper bounds for model (1)

2.1 Lagrangian relaxation

Relaxing constraint (1b) via a Lagrangian dual procedure results in the following model that we call as the dual model for model (1).

$$\mathcal{Z}_D(\lambda) = \max_{x,y,z} \sum_{t\in T} R_t x_t - \mathbb{E}[C_t^\omega y_t^\omega] + \lambda\big(\varepsilon - \sum_{\omega\in\Omega} p^\omega z^\omega\big) \tag{3a}$$

$$\text{s.t.} \quad (1c), (2a), (2c). \tag{3b}$$

We have the following basic bounding principle (BBP) for the Lagrangian dual model:

$$\mathcal{Z} \le \mathcal{Z}_D(\lambda), \forall \lambda \ge 0.$$

The BBP follows from the fact that an optimal solution for model (1) is feasible for model (3), and with a non-negative λ the objective function value in equation (3a) is more than that of equation (1a); see, e.g., [7]. Given an optimal solution (x, y, z) of model (3) for a $\lambda \ge 0$, the BBP holds with equality—$\mathcal{Z} = \mathcal{Z}_D(\lambda)$—if the following two sufficient conditions are met:

(i) $\varepsilon \ge \sum_{\omega\in\Omega} p^\omega z^\omega$, and
(ii) $\lambda\big(\varepsilon - \sum_{\omega\in\Omega} p^\omega z^\omega\big) = 0$.

Then, the Lagrangian dual solves model (1). However, even for LP models, these conditions rarely hold true in practice. For a numerical solution of MIPs—such as model (1) and model (3)—we require the use of an optimization solver, e.g., CPLEX or Gurobi, that works by branching. Then, the solutions provided by the solver are either provably optimal to a small tolerance, or are the best known feasible solutions in a given amount of time.

In this time limit, if the dual problem is solved to optimality, the BBP holds (although not necessarily with equality). Further, this is true in the presence of integer variables as well. If we have a lower bound for model (1) available, we can construct a range for the optimal objective function value of model (1). This is true even if the optimization models for the lower bounding procedure are solved suboptimally. However, if the dual problem is solved suboptimally, only a weak, or conservative, bounding procedure results. Let $\underline{\mathcal{Z}_D}(\lambda)$ and $\overline{\mathcal{Z}_D}(\lambda)$ denote the lower and upper bounds of the dual model reported by the optimization solver. The following simple example serves to illustrate the point of this discussion.

Example 1 Consider the following MIP:

$$\begin{aligned} \mathcal{Z} = \max_{x,y} \quad & 2x + 3y \\ \text{s.t.} \quad & x + y \le 1 \\ & 3x + y \le 3 \\ & x, y \in \{0, 1\}. \end{aligned}$$

The optimal solution is $x = 0, y = 1$ with $\mathcal{Z} = 3$. Consider the following Lagrangian dual model:

$$\begin{aligned}\mathcal{Z}_D(\lambda) = \max_{x,y}\ & 2x + 3y + \lambda(1 - x - y)\\ \text{s.t.}\quad & 3x + y \leq 3\\ & x, y \in \{0, 1\}.\end{aligned}$$

Assume that the branch-and-bound solver terminates with the following feasible solution for the Lagrangian dual model: $x = 1, y = 0$ that provides $\underline{\mathcal{Z}_D}(\lambda) = 2, \forall \lambda \geq 0$. We note that this solution is feasible for the true model. A corresponding upper bound is available from $x = 0, y = 1$ with $\overline{\mathcal{Z}_D}(\lambda) = 3, \forall \lambda \geq 0$. Since $\underline{\mathcal{Z}_D}(\lambda) < \mathcal{Z}$, the feasible solution reported by the solver cannot be used for the BBP. Further, since the precise value of $\mathcal{Z}_D(\lambda) \in [\underline{\mathcal{Z}_D}(\lambda), \overline{\mathcal{Z}_D}(\lambda)]$ is unknown, the BBP is of no value. □

Example 1 demonstrates that, unfortunately, in practice we cannot provide a strong optimality certificate as that given by the theoretical bound in the BBP. Thus, we modify the BBP conservatively as

$$\mathcal{Z} \leq \overline{\mathcal{Z}_D}(\lambda), \forall \lambda \geq 0,$$

and denote it as the modified basic bounding procedure (mBBP). It follows from the mBBP that smaller values of $\overline{\mathcal{Z}_D}(\lambda)$, or $\mathcal{Z}_D(\lambda)$, are of particular interest to us. Solving the optimization problem

$$\overline{\mathcal{Z}} = \min_{\lambda:\lambda\geq 0} \mathcal{Z}_D(\lambda) \tag{6}$$

achieves this, and we do so via the classical subgradient algorithm of Shor [27]. To this end, there are two key quantities of interest.

(i) First, the one-dimensional quantity $\varepsilon - \sum_{\omega\in\Omega} p^\omega z^\omega$ is a *subgradient* of the function $\mathcal{Z}_D(\lambda)$ for any $\lambda \geq 0$ at which z is an optimal solution for model (3). The subgradient essentially determines the direction of descent towards the optimal λ. We denote this subgradient by γ.

(ii) The second quantity is the *step size* that determines the magnitude of decrease in the descent direction. A number of strategies to determine good step sizes have been designed, see, e.g., a series of works by Polyak [21, 22]. We denote the step size by Δ.

Then, at iteration k we update λ by taking a step in a direction opposite to the subgradient: $\lambda_{k+1} \leftarrow \max\{\lambda_k - \Delta_k\gamma_k, 0\}$.

This straightforward Lagrangian relaxation procedure of a chance-constrained optimization model is well-studied in various contexts. The progressive hedging algorithm of Rockafellar and Wets [24], as further developed in [34], is also based on this scheme as follows. In the absence of the first-stage variables x, model (3) separates across the scenarios. Thus, we include a so-called non-anticipativity constraint that creates copies of the x variables as follows: $x_t = \bar{x_t}^\omega, \forall t \in T, \omega \in \Omega$. Then, relaxing both the non-anticipativity and the knapsack constraints completely splits the problem across the scenarios; these separable problems are solvable in parallel to further reduce computational effort. Similar ideas are employed in work on decomposition algorithms for chance-constrained optimization models [4, 10]. Such schemes find particular application in the power systems literature. Ozturk et al. [20] study a Lagrangian relaxation procedure of a stochastic unit commitment problem with chance constraints. Singh and Knueven [28] study a similar procedure for a hybrid storage system with chance constraints.

The above cited works bound chance-constrained, or general stochastic, optimization models with the following motivation. The original optimization model—such as model (1)—is difficult to solve naively. Relaxing a set of constraints by a dual procedure results in a relatively easier optimization model—such as model (3)—that is computationally more tractable. Then, these works seek to iteratively determine the best bounds from the relaxation by solving model (6). Even when $\arg\min_\lambda \mathcal{Z}_D(\lambda)$ cannot be computed, such schemes provide significant value. This value is especially evident when the Lagrangian models are completely [34] or nearly-completely [31] decomposed into independent subproblems, such as in the progressive hedging algorithm. Further, some error bounds given a suboptimal λ are also known, see, e.g., [6].

As we mention in Section 1, we are interested in a different setting than the works cited above. Specifically, we study the situation where the relaxed model is *also* computationally challenging to solve, such that not only $\arg\min_\lambda \mathcal{Z}_D(\lambda)$ is difficult to compute but also the individual models in the sequence

of Lagrangian problems are only solved suboptimally. To this end, our work on upper bounds specifically focuses on the gap between theory and practice in relation to the parameter Δ for the two-stage chance-constrained program given by model (1). Below are six well-known strategies to update this step-size parameter that we consider in this work.

2.2 Subgradient methods

We consider the following six strategies to update the step-size parameter Δ.

(i) First, we consider a constant step size; i.e., $\Delta_k = \Delta > 0$. In Section 5.4, based on a preliminary analysis we use $\Delta_k = 0.58$. We denote this rule as Rule I.

(ii) Second, we consider a constant *step length*. The quantity $|\gamma_k|\Delta_k$ provides the step length or the total amount we change λ by. We then have $\Delta_k = \frac{\alpha}{|\gamma_k|}$, with $\alpha > 0$. In Section 5.4, we use $\Delta_k = \frac{200}{|\gamma_k|}$. We denote this rule as Rule II.

(iii) Third, we consider a sequence Δ_k where the sum diverges, but the squared sum is finite; i.e.,

$$\Delta_k \geq 0, \quad \text{with} \quad \sum_{k=1}^{\infty} \Delta_k^2 < \infty, \quad \sum_{k=1}^{\infty} \Delta_k = \infty.$$

An example of a sequence that satisfies this criterion is $\Delta_k = \frac{a}{b+k}$, where $a, b > 0$. In Section 5.4, we use $\Delta_k = \frac{3}{2+k}$. We denote this rule as Rule III.

(iv) Fourth, we consider a sequence Δ_k where the sum diverges as in Rule III, however the step size shrinks to zero; i.e.,

$$\Delta_k \geq 0, \quad \text{with} \quad \lim_{k\to\infty} \Delta_k = 0, \quad \sum_{k=1}^{\infty} \Delta_k = \infty.$$

An example of a sequence that satisfies this criterion is $\Delta_k = \frac{a}{k^n}$, where $a > 0$ and $0 < n \leq 1$. In Section 5.4, we use $\Delta_k = \frac{1}{\sqrt{k}}$. We denote this rule as Rule IV.

(v) Fifth, analogous to Rule IV, we consider step lengths that diminish but are not summable; i.e.,

$$\Delta_k = \frac{\alpha_k}{|\gamma_k|}, \quad \text{with} \quad \alpha_k \geq 0, \quad \lim_{k\to\infty} \alpha_k = 0, \quad \sum_{k=1}^{\infty} \alpha_k = \infty.$$

In Section 5.4, we use $\Delta_k = \frac{300/k}{|\gamma_k|}$. We denote this rule as Rule V.

(vi) Finally, we consider the step size proposed by Polyak [23, Chapter 5]:

$$\Delta_k = \theta_k \frac{(\overline{\mathcal{Z}_D}_k - \underline{\mathcal{Z}})}{\gamma_k^2},$$

where $\overline{\mathcal{Z}_D}_k$ is an upper bound for the optimal objective function value for model (3) at iteration k given a $\lambda \geq 0$, $\underline{\mathcal{Z}}$ is any lower bound for model (1), and $\theta_k > 0$ is a scalar. From the `mBBP` we are assured that $\Delta_k \geq 0$ for any lower bound $\underline{\mathcal{Z}}$. In Section 3, we provide three ways to compute the lower bounds, $\underline{\mathcal{Z}}$. Following the suggestion by Held et al. [11], in Section 5.4 we use $\theta_1 = 2$ and progressively half this value if there is no improvement in $\mathcal{Z}_D(\lambda)$ in two iterations. We denote this rule as Rule VI.

We summarize the above discussion in Algorithm 1. In Step 1, we begin with the LP relaxation of model (1) to provide an initial upper bound for model (1). We terminate the algorithm in Step 9 if one of the following happens: (i) the gap between the estimated upper bound and the input lower bound is less than a threshold, δ, (ii) a time limit, *time*, is reached, or (iii) the values of λ do not change by more than a threshold, ζ, between two iterations. In the last two cases, the algorithm converges without finding the optimal objective function value $\overline{\mathcal{Z}}$ of model (6). Then, we report the available optimality gap and the conservative upper bound Z_D. As we mention in Section 2.1, we are especially interested in the second of these three cases.

Algorithm 1 Lagrangian relaxation procedure of model (1)

Input: *iter*; $\underline{\mathcal{Z}}$; *time*; θ; ψ; ι; step-size rule r.
Output: $\mathcal{Z}_D$; δ.
1: Solve LP relaxation of model (1); $\mathcal{Z}_D \leftarrow$ optimal objective function value; $\lambda \leftarrow$ optimal dual of constraint (2b).
2: $k \leftarrow 1$.
3: **while** $k \leq iter$ **do**
4: Solve model (3); $\mathcal{Z}_D \leftarrow \min\{\mathcal{Z}_D, \overline{\mathcal{Z}_D}(\lambda)\}$; $\gamma \leftarrow \varepsilon - \sum_{\omega\in\Omega} p^\omega z^\omega$ with optimal z^ω.
5: If no change in $\mathcal{Z}_D$ between current and previous two iterations, $\theta \leftarrow \frac{\theta}{2}$.
6: Update Δ according to step-size rule r.
7: $\lambda_{\text{new}} \leftarrow \max\{0, \lambda - \Delta\gamma\}$; $\zeta \leftarrow |\lambda_{\text{new}} - \lambda|$; $\lambda \leftarrow \lambda_{\text{new}}$
8: $\delta \leftarrow \frac{\mathcal{Z}_D - \underline{\mathcal{Z}}}{\mathcal{Z}_D}$.
9: If $\delta \leq \psi$ or time $\geq$ *time* or $\zeta < \iota$, *STOP*.
10: $k \leftarrow k + 1$; update time to the cumulative wall-clock time.

2.3 Statistical validation

In practice, high quality bounds that are achieved quickly are of greater interest than provably optimal bounds that require significant computational effort. For example, although Polyak's works [21,22] provide strategies that are guaranteed to converge to the optimal λ for convex optimization models, they often require improvisation in practice [11]. Model (3) is non-convex due to the presence of the binary variables z; see, also, [6].

In this work, we seek to identify step-size rules that perform "better" than others in practice. Further, we are motivated by the setting where the individual problems in the sequence of Lagrangian models can only be solved suboptimally. To this end, we employ 20 batches of two computationally expensive instances of chance-constrained programs from existing works to develop statistical guarantees; i.e., for both instances of model (1), given the parameters, we sample the uncertainties 20 times each. Both of these instances could not be solved to optimality in the existing works. In this section, we present a hypothesis test that we use to validate the performance of Algorithm 1 statistically, and we present the results of this test on the two instances in Section 5.4.

Let μ_r denote the expected gap computed by Algorithm 1 for step-size rule r. Consider the following null and alternate hypotheses:

$$H_0 : \mu_r \geq \mu_s, \qquad H_1 : \mu_r < \mu_s. \tag{7}$$

We employ a Welch t-test with a test statistic T that has a Student-t distribution with ν degrees of freedom as follows:

$$T = \frac{\overline{r} - \overline{s}}{\sqrt{\frac{s_r^2}{20} + \frac{s_s^2}{20}}}, \quad \text{where } T \sim t_\nu, \quad \text{with } \nu = \left\lfloor \frac{\left(\frac{s_r^2}{20} + \frac{s_s^2}{20}\right)^2}{\frac{1}{19}\left(\frac{s_r^2}{20}\right)^2 + \frac{1}{19}\left(\frac{s_s^2}{20}\right)^2} \right\rfloor. \tag{8}$$

In equation (8), $\overline{r}$ and $\overline{s}$ denote the average gap over the 20 batches for step-size rules r and s, respectively. Analogously, s_r and s_s are the corresponding sample standard deviations of the gaps. For a 5% level of significance, we reject the null hypothesis if $T < t_{0.05,\nu}$, where $t_{0.05,\nu}$ denotes the 5-percentile quantile of the t-distribution with ν degrees of freedom. Rejecting the null hypothesis suggests rule r performs better than rule s.

3 Lower bounds for model (1)

A corresponding lower bound, $\underline{\mathcal{Z}}$, for model (1) is available via any feasible solution. Good quality feasible solutions that provide large values of the objective function are of interest to us for at least two reasons. First, the presence of a lower bound, in addition to the upper bounds, from Section 2 helps determine an optimality gap for model (1). Second, step-size Rule VI requires a lower bound. In this section, we summarize three methods to determine lower bounds for model (1). Each of the three methods has their own merits as well as shortcomings that we acknowledge below.

3.1 An iterative regularization bound

In [29], the authors provide an iterative regularization-based (IR) heuristic that achieves statistically validated lower bounds for a chance-constrained optimization model. This regularization method is motivated by classical descent algorithms in the continuous optimization literature (see, e.g., [7]), with a notable difference of the use of independent samples of scenarios at each iteration. Traditional regularization methods work by modifying the objective function via a proximal term, see, e.g., [32]. Singh et al. [29] exploit this idea to break symmetries in MIP models and achieve good quality feasible solutions quickly. Although the authors do not provide any guarantees of convergence, they achieve lower bounds that are nearly double of those obtained via a naive solution method on their test instances [29]. Thus, the method is promising; however, to the best of our knowledge, it has not been tested on other instances of chance-constrained models. We test the computational performance of this method in Section 5.3.

Algorithm 2 summarizes the IR procedure of [29], where the authors assume equally likely scenarios. Model (9) is the regularized model, where $\rho > 0$ is a regularization parameter and $\hat{x}$ is a proximal term. The IR iteratively solves SAAs of model (9) of progressively increasing size, m; i.e., all the considered scenarios have a probability of $1/m$.

$$\max_{x,y} \sum_{t\in T} \left(R_t x_t - \mathbb{E}[C_t^\omega y_t^\omega]\right) - \sum_{t\in T} \rho|x_t - \hat{x}_t| \tag{9a}$$

$$\text{s.t.} \quad (1c), (2a) - (2c). \tag{9b}$$

Algorithm 2 Iterative regularization bound of [29] for model (1)

Input: m; κ; *time*; *iter*; ρ; Oracle to generate independent scenarios for model (1); instance of model (1) with $|\Omega|$ scenarios.
Output: $\underline{\mathcal{Z}}$.

$\underline{\mathcal{Z}} \leftarrow 0$; $k \leftarrow 1$.
while time $\leq$ *time* and $k \leq$ *iter* **do**
  if $k = 1$ **then**
    Generate m independent scenarios from Oracle; solve SAA of model (1) with these m scenarios; $\hat{x} \leftarrow$ optimal x.
  else
    Generate m independent scenarios from Oracle; solve SAA of model (9) with these m scenarios; $\hat{x} \leftarrow$ optimal x.
  Solve input instance of model (1) with x fixed to $\hat{x}$; $z_m \leftarrow$ objective function value.
  $\underline{\mathcal{Z}} \leftarrow \max\{z_m, \underline{\mathcal{Z}}\}$.
  $m \leftarrow \lceil (1+\kappa)m \rceil$.
  $k \leftarrow k + 1$; update time to wall-clock time.

3.2 An aggregation bound

A large number of scenarios in the set Ω lead to computational difficulties in solving model (1). Choosing a new, smaller, set of scenarios that "represents" Ω is one way to handle this computational difficulty. This idea is again related to the classical scenario aggregation, or progressive hedging, technique proposed in [24]; although clustering techniques for general data points are well-studied in other disciplines, see, e.g., [12, 25]. In this section, we develop a second lower bound by integrating a clustering procedure within the IR bound.

To this end, we begin with a large pool of scenarios that we aggregate (or, cluster) into a set of a few representative scenarios. Several methods are available to determine these representative scenarios, see, e.g., [14] for a comparison of these methods and their respective performance on energy system models. We then solve this relatively smaller problem to obtain an optimal solution for the first-stage x variables. Next, we solve the original problem with the x variables fixed to this value; the resulting objective function value provides a lower bound for $\mathcal{Z}$. Ideally, both of these problems are solved quickly. However, the procedure fails if the second problem is infeasible. For models with relatively complete recourse—where the second-stage problem is feasible for any choice of the first-stage variables—this procedure guarantees a feasible solution.

The scheme we describe above can be further improved, and we do so in Algorithm 3. To this end, we use the `IR` procedure of Algorithm 2 but instead of sampling m identically distributed samples in Step 8, we use the above-described clustering procedure to generate m samples. In Section 5.3, we use the so-called hierarchical aggregation method presented by Hoffmann et al. [13] for the procedure `cluster`(M, m). Given a pool of samples of size M, this procedure returns m aggregated scenarios with respective weights for each scenario. These weights determine the probability, or relevance, of each of these m scenarios; i.e., the scenarios are no longer equally likely. We denote this lower bound as the aggregation procedure (`AP`) bound.

Algorithm 3 Aggregation bound for model (1)

Input: m; κ; *time*; *iter*; ρ; p; Oracle to generate independent scenarios for model (1); procedure `cluster`(a, b) that aggregates set of a scenarios into set of b scenarios and their respective weights, $a > b$; instance of model (1) with $|\Omega|$ scenarios.
Output: $\underline{\mathcal{Z}}$.
1: Generate $p >> |\Omega|$ independent scenarios from Oracle.
2: $\underline{\mathcal{Z}} \leftarrow 0$; $k \leftarrow 1$.
3: **while** time $\leq$ *time* and $k \leq$ *iter* **do**
4: **if** k=1 **then**
5: Use `cluster`(p, m); solve SAA of model (1) with these m scenarios and their respective weights; $\hat{x} \leftarrow$ optimal x.
6: **else**
7: Use `cluster`(p, m); solve SAA of model (9) with these m scenarios and their respective weights; $\hat{x} \leftarrow$ optimal x.
8: Solve input instance of model (1) with x fixed to $\hat{x}$; let z_m denote the objective function value.
9: $\underline{\mathcal{Z}} \leftarrow \max\{z_m, \underline{\mathcal{Z}}\}$.
10: $m \leftarrow \lceil (1 + \kappa)m \rceil$.
11: $k \leftarrow k + 1$; update time to wall-clock time.

3.3 A quantile bound

Another lower bound is available from a procedure described and implemented in [28]. Here, Singh and Knueven [28] modify the quantile-based bound of Ahmed et al. [2]—originally used to generate upper bounds—to generate lower bounds from feasible solutions. We first note that a feasible solution for the z variables for model (1) is available by choosing an arbitrary number, k, of scenarios ω as 1 and the rest as 0, where k is the cardinality of the set $\{\omega : \sum_{\omega \in \Omega} p^\omega \leq \varepsilon\}$. For equally likely scenarios, we have $k = \lfloor |\Omega|\varepsilon \rfloor$.

In [28], the authors consider equally likely scenarios and first solve model (1) separately for each scenario ω by fixing $z^\omega = 0$; i.e., a set of $|\Omega|$ subproblems. Similar to the progressive hedging subproblems that we discuss in Section 2, each of these subproblems are computationally cheap to solve and can also be solved in parallel. Then, the procedure chooses $z^\omega = 1$ for the worst performing $\lfloor |\Omega|\varepsilon \rfloor$ scenarios; i.e., those with the lowest objective function values. Solving model (1) for this fixed value of z provides a lower bound, $\underline{\mathcal{Z}}$. On the test instances defined in [28], the authors are able to complete this procedure for $|\Omega| = 1200$ in about half a minute. However, if this final problem is computationally intensive—despite the z variables being fixed—this heuristic is not effective. As we demonstrate later in our work, this can indeed happen. We denote this bound as the quantile procedure (`QP`) bound, and summarize this discussion in Algorithm 4.

Algorithm 4 Quantile bound of [28] and [2] for equally likely scenarios for model (1)

Input: instance of model (1) with $|\Omega|$ scenarios.
Output: $\underline{\mathcal{Z}}$.
1: Solve model (1) separately for each ω in the input instance with $z^\omega \leftarrow 0$.
2: Sort corresponding objective function values in ascending order; $z^\omega \leftarrow 1$ for first $\lfloor |\Omega|\varepsilon \rfloor$ scenarios, $z^\omega \leftarrow 0$ for rest.
3: Solve input instance of model (1) with z fixed from Step 2.
4: $\underline{\mathcal{Z}} \leftarrow$ objective value.

4 Two instances of model (1)

4.1 A hybrid solar-battery storage system

We first consider the chance-constrained optimization model presented in [28]. The authors consider equally likely scenarios, and in what follows we do so as well. This model seeks to maximize the profit from a day-ahead promise of power minus the cost of operating a battery. Energy is produced only via a photovoltaic power station, and excess production is stored in the battery. The JCC ensures highly reliable operations despite the limited knowledge of future solar power forecasts. The set Y determines the standard operating regime for the battery — ramping rates and storage limits. This set does not contain any binary variables. We denote this instance of model (1) as Model I.

4.2 A hybrid wind-conventional generator system

Next, we consider the adaptive chance-constrained optimization model presented in [29]. This model has equally likely scenarios as well. The model seeks to maximize profit given by revenue from a day-ahead bid for power minus the expected generation costs. Energy is generated via a natural gas generator. The JCC again ensures highly reliable operations for the day-ahead market. The set Y consists of nearly-standard operating constraints for a generator — ramping, startup and shutdown rates, and minimum and maximum power generation limits. These constraints form part of so-called unit commitment models. The set Y now contains binary variables; i.e., model (1) contains binary variables additional to the binary z variables of equation (2). Thus, this instance of model (1) is more challenging than Model I. We denote this instance as Model II.

5 Computational results

5.1 Computational setup

All computational experiments are carried out on two high performance computing clusters at the Regionales Rechenzentrum Erlangen with Intel Xeon E3-1240 v5 processors with 32 GB of RAM with GAMS version 24.8.5. We use Gurobi version 7.0.2 for solving all the optimization models. We solve model (1) naively to an optimality gap of 0.1% using a time limit of 4200 seconds. We solve each iteration of the Lagrangian relaxation of model (3) to an optimality gap of 0.01% using a time limit of 2100 seconds per iteration. We allow at most ten iterations for the Lagrangian relaxation procedure with a total time limit of 4200 seconds. We stop the procedure if the total time after completion of any iteration exceeds 2250 seconds; else, the procedure could exceed the total time limit of 4200 seconds. In this way, our results for the Lagrangian relaxation are conservative. Thus, in Algorithm 1, we use $iter = 10$, $time = 2250$, $\theta = 2$, $\psi = 0.001$ and $\iota = 0.0001$.

For the `IR` and `AP` bounds in Algorithm 2 and Algorithm 3, respectively, we use $m = 20$, $\kappa = 0.1$, $time = 1800$, $iter = 30$, $\rho = 0.1$ and $p = 7200$. In Algorithm 2 and Algorithm 3, we solve the SAAs of model (1) to an optimality gap of 4% with a time limit of 2100 seconds, the SAAs of model (9) to an optimality gap of 0.05% with a time limit of 2100 seconds, and the models with the x variables fixed to an optimality gap of 0% with a time limit of 1800 seconds. For the `QP` bound in Algorithm 4, we solve all the subproblems to an optimality gap of 0.01% using a time limit of 2100 seconds; although the iterations in Step 1 of Algorithm 4 are solved very quickly.

We use four sets of scenario-size regimes for both Model I and Model II: $|\Omega| = \{100, 600, 900, 1500\}$. For each of the regimes, we sample scenarios independently using procedures detailed in [29] and [28] for Model I and Model II, respectively. The parameters in the set Y are unchanged from those in the original works. We use four regimes of the reliability threshold: $\varepsilon = \{0.01, 0.03, 0.05, 0.07\}$; i.e., we have 16 instances for both Model I and Model II.

5.2 Analysis: naive solve

First, we present results using a direct naive solution method of Model I and Model II. Table 1 summarizes our computational experiments. Naturally, instances with a larger number of scenarios are computationally more challenging to solve. Increasing the reliability threshold, ε, leads to a greater number of combinatorics that is further reflected in an increased computational effort. Except for the smallest instances of Model I with $|\Omega| = 100$, none of the instances are solved to optimality. As we mention before, Model II is computationally more challenging to solve than Model I. For each of the 16 instances we consider, the optimality gap—see fifth column of Table 1—is larger for Model II as compared to Model I. The average gap for the naive solution method over the 16 instances for Model I and Model II is 12.9% and 61.0%, respectively.

$\|\Omega\|$	ε	Objective	Time	Gap
100	0.01	278.54	53	-
	0.03	280.05	284	-
	0.05	280.98	401	-
	0.07	283.99	615	-
600	0.01	246.57	**T**	6.85%
	0.03	232.13	**T**	16.47%
	0.05	232.30	**T**	19.20%
	0.07	246.98	**T**	15.73%
900	0.01	224.91	**T**	16.14%
	0.03	224.83	**T**	19.83%
	0.05	225.12	**T**	21.15%
	0.07	224.83	**T**	23.29%
1500	0.01	238.57	**T**	10.37%
	0.03	230.36	**T**	17.37%
	0.05	233.60	**T**	18.70%
	0.07	230.33	**T**	21.53%

(a) Model I

$\|\Omega\|$	ε	Objective	Time	Gap
100	0.01	6,052.86	**T**	29.51%
	0.03	6,337.38	**T**	37.08%
	0.05	6,605.23	**T**	43.46%
	0.07	6,946.04	**T**	47.83%
600	0.01	4,441.43	**T**	53.30%
	0.03	4,150.03	**T**	62.99%
	0.05	4,234.92	**T**	66.95%
	0.07	4,326.37	**T**	70.06%
900	0.01	3,823.28	**T**	60.78%
	0.03	3,133.00	**T**	72.81%
	0.05	2,921.09	**T**	77.85%
	0.07	3,254.70	**T**	78.05%
1500	0.01	3,745.73	**T**	61.84%
	0.03	3,849.86	**T**	66.37%
	0.05	3,550.13	**T**	72.92%
	0.07	3,696.89	**T**	74.96%

(b) Model II

Table 1: Computational results for a naive solution method on instances of model (1). An entry of "**T**" denotes the instance could not be solved to optimality in the time limit, while an entry of "−" denotes the instance solved to optimality. For details, see Section 5.2.

The set of scenarios we use is different than those used in the previous works [28, 29], and our gaps in Table 1 are larger than those reported previously. Further, our computational setup is different than that in [28, 29] as well. Thus, to further validate the trend we report, we repeat this experiment using 20 independent batches of scenarios for both Model I and Model II. We do this for all four scenario-size regimes with $\varepsilon = 0.05$. Table 2 presents the 95% confidence intervals for the objective function value, the

time, and the optimality gap. The observations we report above again follow, thereby suggesting that the computational difficulty is not biased by the particular scenarios we sample.

$\|\Omega\|$	Objective CI	Time CI	Gap CI [%]
100	(280.94, 282.29)	(387, 453)	(0.01, 0.04)
600	(239.28, 244.64)	**T**	(14.77, 16.49)
900	(234.08, 240.19)	**T**	(49.19, 86.23)
1500	(233.42, 237.98)	**T**	(14.16, 37.89)

(a) Model I

$\|\Omega\|$	Objective CI	Time CI	Gap CI [%]
100	(6,385.63, 6,529.70)	**T**	(42.69, 43,53)
600	(4,588.53, 4,755.71)	**T**	(62.13, 63.38)
900	(4,296.65, 4,417.90)	**T**	(65.62, 66.55)
1500	(2,879.22, 3,308.35)	**T**	(74.92, 78.17)

(b) Model II

Table 2: Computational results for a naive solution method on 20 instances of model (1) for $\varepsilon = 0.05$. CI denotes the 95% confidence interval. For details, see Section 5.2.

The large optimality gaps for both the instances further warrant the need for heuristics and approximations for such chance-constrained programs. In the proceeding sections, we present our computational experiments to reduce the optimality gaps for both the instances. We begin with lower bounds.

5.3 Analysis: lower bounds

In this section, we compare the three lower bounding heuristics of Section 3 for Model I and Model II. Table 3 summarizes our results. In the practical implementation, we deviate slightly from Algorithm 2 and 3 for the `IR` and `AP` bounds, respectively. We add the time required for Step 7 and 8 in Algorithm 2 and 3, respectively, after the completion of the algorithms. This affects only the last row of Table 3(a) where the total time slightly exceeds the limit of 2100 seconds; for further details on our implementation, see our code that is publicly available at `https://github.com/charlotteritter/ArticleSubgradient`. In Table 3, entries marked with a ✗ denote the instance is infeasible for the corresponding heuristic; i.e., the heuristic fails. The "Improvement" columns denote the relative improvement of the heuristic with respect to the best known feasible solution obtained by the naive solution method (Table 1); i.e, $100\frac{\text{heuristic - naive}}{\text{naive}}\%$, where heuristic and naive are the objective function values in Table 3 and Table 1, respectively.

Similar to the naive solution method, the `IR` and `AP` bounds require greater computational effort for instances with a larger number of scenarios and regimes with a higher tolerance threshold. This is because Step 7 of Algorithm 2 and Step 8 in Algorithm 3, require a solution of model (1) which can be challenging despite the fixed first-stage variables. The largest instance—see last row of Table 3(a)—requires over 2100 seconds for the `IR` bound. Still, even for this particular instance, this computational effort is half of what we use as the time limit for the naive solution, and, yet, we achieve an improvement over 10%. Further, for the `IR` bound, 10 and 12 of the 16 instances obtain bounds at least as good as that obtained naively in Table 1 for Model I and Model II, respectively. The average improvement over the 16 instances, for Model I and Model II using the `IR` bound is 2.8% and 29.6%, respectively. The larger instances of model (1) are particularly suited for the `IR` bound. For the eight instances of the $|\Omega| = 900$ and $|\Omega| = 1500$ regimes, the average improvements for Model I and Model II are 7.4% and 54.5%, respectively. We emphasize that this improvement is achieved in less than half the time required by the naive solution method.

In terms of the computational effort required, the `AP` bound behaves similar to the `IR` bound. However, as we observe from Table 3, the `AP` bound is often unsuccessful in obtaining a feasible solution. For Model I, the `AP` bound is only successful for four of the 16 instances and is always worse than the naively

		IR			AP			QP		
$\|\Omega\|$	ε	Objective	Time	Improvement	Objective	Time	Improvement	Objective	Time	Improvement
100	0.01	253.92	123	-8.84%	232.26	127	-16.62%	278.54	53	0.00%
	0.03	258.16	191	-7.82%	232.54	451	-16.96%	279.16	61	-0.32%
	0.05	265.16	245	-5.63%	232.79	429	-17.15%	280.43	57	-0.20%
	0.07	265.43	289	-6.54%	277.26	693	-2.37%	283.71	58	-0.10%
600	0.01	238.39	669	-3.32%	✗	244		248.50	13	0.78%
	0.03	240.03	768	3.40%	✗	552		251.93	16	8.53%
	0.05	254.61	829	9.60%	✗	583		252.48	18	8.69%
	0.07	258.43	873	4.64%	✗	846		253.08	21	2.47%
900	0.01	234.38	1024	4.21%	✗	267		244.89	30	8.88%
	0.03	239.71	1115	6.62%	✗	630		245.71	46	9.29%
	0.05	253.70	1194	12.70%	✗	573		245.97	254	9.26%
	0.07	255.05	1246	13.44%	✗	948		246.22	194	9.51%
1500	0.01	235.36	1815	-1.35%	✗	429		233.37	69	-2.18%
	0.03	239.46	1970	3.95%	✗	836		235.44	61	2.21%
	0.05	253.49	2083	8.51%	✗	739		235.63	59	0.87%
	0.07	254.93	2137	10.68%	✗	1260		239.96	56	4.18%

(a) Model I

		IR			AP			QP		
$\|\Omega\|$	ε	Objective	Time	Improvement	Objective	Time	Improvement	Objective	Time	Improvement
100	0.01	4,951.30	**T**	-18.20%	4,578.96	**T**	-24.35%	6,051.36	**T**	-0.02%
	0.03	5,304.93	**T**	-16.29%	4,694.28	**T**	-25.93%	6,648.47	**T**	4.91%
	0.05	5,624.33	**T**	-14.85%	4,850.88	**T**	-26.56%	7,122.64	**T**	7.83%
	0.07	6,113.24	**T**	-11.99%	✗	**T**		7,620.29	**T**	9.71%
600	0.01	4,602.68	**T**	3.63%	4,431.45	**T**	-0.22%	4,371.01	**T**	-1.59%
	0.03	4,980.82	**T**	20.02%	4,725.91	**T**	13.88%	4,777.36	**T**	15.12%
	0.05	5,633.56	**T**	33.03%	5,138.94	**T**	21.35%	4,648.14	**T**	9.76%
	0.07	6,144.55	**T**	42.03%	✗	**T**		4,949.73	**T**	14.41%
900	0.01	4,547.32	**T**	18.94%	4,376.98	**T**	14.48%	4,267.61	**T**	11.62%
	0.03	5,083.08	**T**	62.24%	4,553.03	**T**	45.32%	4,534.42	**T**	44.73%
	0.05	5,601.94	**T**	91.78%	4,941.90	**T**	69.18%	4,477.66	**T**	53.29%
	0.07	6,113.63	**T**	87.84%	✗	**T**		4,660.55	**T**	43.19%
1500	0.01	4,603.75	**T**	22.91%	4,470.61	**T**	19.35%	4,069.86	**T**	8.65%
	0.03	5,081.49	**T**	31.99%	4,683.39	**T**	21.65%	4,104.78	**T**	6.62%
	0.05	5,543.66	**T**	56.15%	5,104.65	**T**	43.79%	4,058.15	**T**	14.31%
	0.07	6,056.66	**T**	63.83%	✗	**T**		4,589.12	**T**	24.13%

(b) Model II

Table 3: Computational results for three lower bounding heuristics on instances of model (1). A ✗ indicates the instance is infeasible for the heuristic. For details, see Section 3.

obtained objective function value. Yet, analogous to the IR bound, the AP bound is useful for the more challenging instances of Model II. Although four of the 16 instances are infeasible within the give time limit, the average improvement over the other 12 is 14.3%. For the six feasible instances of the $|\Omega| = 900$ and $|\Omega| = 1500$ regimes, the average improvements for Model II increase to 35.6%.

The QP bound is obtained fast for Model I. For Model I, 12 of the 16 instances obtain bounds at least as good as that obtained naively in Table 1(a). Except for two instances, all instances are solved in less than about a minute. Although the QP bound is not obtained as quickly as for Model I for Model II, it still manages to achieve solutions that are comparable or better than that obtained by the naive solution method. For Model II, 14 of the 16 instances obtain bounds at least as good as that obtained naively in Table 1(b). The average improvement over the 16 instances for Model I and Model II using the QP bound is 3.9% and 16.7%, respectively.

To get more representative results, we repeat the calculation of the three different lower bounds on the same 20 independent batches of scenarios that we use in Section 5.2 for all four scenario-size regimes for Model I and Model II. We do this for $\varepsilon = 0.05$. Table 4 presents the 95% confidence intervals for the objective function value, the time, and the improvement in relation to the naive solution of the three lower bounding techniques. Here, the trend we report in Table 3 is further validated. For Model I, the QP bound is computed very quickly; however, compared to the IR bound over the larger instances, the

	IR			AP			QP		
$\|\Omega\|$	Objective CI	Time CI	Improvement CI [%]	Objective CI	Time CI	Improvement CI [%]	Objective CI	Time CI	Improvement CI [%]
100	(261.70, 263.43)	(244, 245)	(-7.08, -6.35)	(232.76, 232.97)	(430, 431)	(-17.45, -17.08)	(279.93, 281.63)	(4, 4)	(-0.44, -0.05)
600	(253.65, 254.17)	(822, 827)	(3.80, 6.20)	✗	✗	✗	(246.37, 249.76)	(44, 48)	(1.43, 3.72)
900	(253.61, 254.10)	(1194, 1198)	(5.39, 8.37)	✗	✗	✗	(244.88, 247.96)	(73, 81)	(2.13, 5.37)
1500	(253.72, 254.29)	(2087, 2096)	(6.73, 8.87)	✗	✗	✗	(239.06, 243.83)	(165, 184)	(1.60, 3.28)

(a) Model I

	IR			AP			QP		
$\|\Omega\|$	Objective CI	Time CI	Improvement CI [%]	Objective CI	Time CI	Improvement CI [%]	Objective CI	Time CI	Improvement CI [%]
100	(5,650.76, 5,871.67)	**T**	(-12.45, -9.07)	(5,110.87, 5,342.39)	**T**	(-20.98, -17.07)	(6,729.79, 6,852.93)	**T**	(4.60, 5.77)
600	(5,494.22, 5,612.70)	**T**	(17.37, 20.57)	(5,061.44, 5,184.64)	**T**	(7.66, 11.92)	(4,700.14, 4,896.32)	**T**	(-0.21, 5.96)
900	(5,545.61, 5,621.69)	**T**	(26.29, 30.22)	(5,144.17, 5,241.68)	**T**	(17.57, 20.95)	(4,344.65, 4,563.58)	**T**	(-0.60, 5.23)
1500	(5,574.85, 5,638.68)	**T**	(69.50, 103.09)	(5,148.49, 5,215.68)	**T**	(56.35, 88.31)	(3,184.35, 3,477.76)	**T**	(-0.92, 23.07)

(b) Model II

Table 4: Computational results for three lower bounding heuristics on 20 instances of model (1) for $\varepsilon = 0.05$. CI denotes the 95% confidence interval. For details, see Section 5.3.

improvements for QP are much smaller. For example, consider the $|\Omega| = 1500$ regime for Model I. The QP bound has an improvement of at most 3.3%, while the IR bound has an improvement that is at least double (6.7%). We further observe that the AP bound is practically useless for Model I. For all instances, in each of the 20 batches with $|\Omega| \geq 600$, we fail to obtain a feasible solution with the AP bound.

As in Table 3(b), we reach the time limit in Table 4(b) for all methods for the computationally more challenging Model II. However, in contrast to Table 4(a) we observe that the QP bound lags behind not only the IR bound but also the AP bound. For larger instances of Model II, the upper limit of the CI for the QP bound is lower than the lower limit of the CI for both the IR and AP bounds. The IR bound performs the best here, consistently delivering the highest improvement CIs for all instances except the smallest instance with $|\Omega| = 100$. For the smallest instance, the QP performs better for both Model I and Model II, further validating our empirical claim that the IR and AP bounds are most suitable for the larger instances. Finally, we also observe that the improvement by the IR bound is significantly higher for Model II than for Model I. This is especially evident from the last rows of Table 4(a) and Table 4(b) where the improvements in Model II increase by an order of magnitude, e.g., from 6.7% to 69.5%.

Summarizing, the computational difficulties associated with the challenging structure of Model II are reflected in the three heuristics we study as well. We now revisit Q1 and Q2 from Section 1. All instances of Model II consume the entire time limit for each of the three heuristics; yet, to answer Q1, they show significant merit compared to a naive solution method. To answer Q2 for the lower bounding procedures, we observe that the IR bound is the most effective in terms of obtaining large lower bounds for both Model I and Model II. The AP bound is not effective for Model I, however it has value for Model II. For Model I, the QP bound is also efficient in the sense of achieving good quality bounds quickly in only a fraction of the time required by the IR bound.

We further conclude that regularization with aggregated scenarios, at least using the hierarchical aggregation procedures described in [13], has no significant advantage compared to regularization using i.i.d. scenarios. Future work could examine specialized aggregation techniques that assist in regularization methods for chance-constrained programs. Quantile-based bounds are particularly effective for models that are easy to solve when the binary z variables are fixed. On the other hand, regularization-based heuristics offer advantage when the second-stage model is easy to solve given the first-stage variables. Future work could also study lower bounding procedures that begin with a quantile-based bound and then proceed to regularization-based heuristics.

5.4 Analysis: comparison of the six step-size rules

In this section, we statistically validate the performance of the six step-size rules of Section 2.2. For each of the four scenario-size regimes, and for the two tolerance threshold regimes of $\varepsilon = 0.01, 0.05$, for both Model I and Model II, we compare the gaps of the Lagrangian relaxation procedure given by Algorithm 1. In Algorithm 1, we use $\underline{\mathcal{Z}}$ as the maximum of the three lower bounds computed in Section 5.3 for the corresponding instance. We are interested in determining step-size rules that reject the null hypothesis

of the Welch-t test of equation (7). Table 5 summarizes the results of this test for Model I and Model II. Here, we use the same 20 batches of scenarios that we report in Section 5.2 for each of the four scenario-size regimes. A ✓ denotes that we have statistically significant evidence to reject the null hypothesis of equation (7), while a ✗ denotes that we are unable to do so; i.e., a ✓ suggests Rule r performs better than Rule s. Thus, every entry in Table 5 results from the solution of 20 sets of optimization models.

(r,s)	$\lvert\Omega\rvert$			
	100	600	900	1500
(I,II)	✓ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(I,III)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(I,IV)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(I,V)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(I,VI)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(II,I)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(II,III)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(II,IV)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(II,V)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(II,VI)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗

(r,s)	$\lvert\Omega\rvert$			
	100	600	900	1500
(III,I)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(III,II)	✓ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(III,IV)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(III,V)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(III,VI)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(IV,I)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(IV,II)	✓ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(IV,III)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(IV,V)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(IV,VI)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗

(r,s)	$\lvert\Omega\rvert$			
	100	600	900	1500
(V,I)	✗ | ✓	✗ | ✗	✗ | ✗	✗ | ✗
(V,II)	✗ | ✓	✗ | ✗	✗ | ✗	✗ | ✗
(V,III)	✗ | ✓	✗ | ✗	✗ | ✗	✗ | ✗
(V,IV)	✗ | ✓	✗ | ✗	✗ | ✗	✗ | ✗
(V,VI)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(VI,I)	✗ | ✓	✗ | ✗	✗ | ✗	✗ | ✗
(VI,II)	✓ | ✓	✗ | ✗	✗ | ✗	✗ | ✗
(VI,III)	✗ | ✓	✗ | ✗	✗ | ✗	✗ | ✗
(VI,IV)	✗ | ✓	✗ | ✗	✗ | ✗	✗ | ✗
(VI,V)	✓ | ✓	✗ | ✗	✗ | ✗	✗ | ✗

(a) Model I

(r,s)	$\lvert\Omega\rvert$			
	100	600	900	1500
(I,II)	✗ | ✓	✗ | ✗	✗ | ✗	✗ | ✗
(I,III)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(I,IV)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(I,V)	✗ | ✓	✗ | ✗	✗ | ✗	✗ | ✗
(I,VI)	✗ | ✓	✗ | ✗	✗ | ✗	✗ | ✗
(II,I)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(II,III)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(II,IV)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(II,V)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(II,VI)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗

(r,s)	$\lvert\Omega\rvert$			
	100	600	900	1500
(III,I)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(III,II)	✗ | ✓	✗ | ✗	✗ | ✗	✗ | ✗
(III,IV)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(III,V)	✗ | ✓	✗ | ✗	✗ | ✗	✗ | ✗
(III,VI)	✗ | ✓	✗ | ✗	✗ | ✗	✗ | ✗
(IV,I)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(IV,II)	✗ | ✓	✗ | ✗	✗ | ✗	✗ | ✗
(IV,III)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(IV,V)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(IV,VI)	✗ | ✓	✗ | ✗	✗ | ✗	✗ | ✗

(r,s)	$\lvert\Omega\rvert$			
	100	600	900	1500
(V,I)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(V,II)	✗ | ✓	✗ | ✗	✗ | ✗	✗ | ✗
(V,III)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(V,IV)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(V,VI)	✗ | ✓	✗ | ✗	✗ | ✗	✗ | ✗
(VI,I)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(VI,II)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(VI,III)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(VI,IV)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗
(VI,V)	✗ | ✗	✗ | ✗	✗ | ✗	✗ | ✗

(b) Model II

Table 5: Statistical performance of step-size rules for Algorithm 1 for $\varepsilon = 0.01 \mid \varepsilon = 0.05$. An entry of (r, s) in the first column denotes that the null hypothesis is $H_0 : \mu_r \geq \mu_s$. Rejection of the null hypothesis is denoted by a ✓, while a failure to reject is denoted by a ✗. For details, see Section 2 and Section 5.4.

First, we analyze the results for $\varepsilon = 0.05$ that we report on the right side of the entries of Table 5. For Model I, Rule VI has statistically significant evidence to perform better than all other rules for the smallest regime of $|\Omega| = 100$. Rule VI—given by Polyak [22] and further modified by Held et al. [11]—is one of the most employed subgradient step-size rules with a rich history of extensions [9,16,19]. However, for Model II, the computational evidence in favor of any particular rule is not convincing; e.g., Rule I, Rule II, Rule III, Rule IV, Rule V, and Rule VI have only three, zero, three, two, two, and zero instances, respectively, that perform better than any of the other five rules (over the 32 instances). Thus, overall, the only statistically validated conclusion we can draw is that Rule V and Rule VI perform well on instances that are the cheapest computationally; i.e., those for Model I with $|\Omega| = 100$. For all of the other instances, however, we have very few rejections and thus little evidence for which rule performs the best.

The above results bring us to the two questions we raise in Section 1. For computationally challenging instances of Model I and Model II, iterations of model (3) are often not solved to optimality in the given time limit. Depending on the step size, this suboptimality happens at different iterations of Algorithm 1. This observation is especially visible for Rule IV where for all the scenario regimes, except $|\Omega| = 100$, even the first iteration of Algorithm 1 is not solved to optimality. Although our computational setup ensures that at least two iterations of Algorithm 1 are completed in the given time limit (albeit, suboptimally), we observe that the smallest upper bound is almost always obtained at the first iteration itself. Thus, the upper bound is often the same for all six rules. The first iteration is likely assisted by the dual values

of λ from the LP relaxation in Step 1 of Algorithm 1. Hence, there is inconclusive evidence to determine a single rule that performs better than others.

To further validate this observation, we conduct additional computational experiments. First, we repeat this experiment for Model I and Model II at a 90% significance level (not shown), however the results are practically unchanged. Second, we perform the same experiments and tests for a tolerance of $\varepsilon = 0.01$. These results are listed as the left entries in Table 5. However, this new experimental design does not significantly change the results we reported previously. For Model I, there are only five rejections over all the sets of instances—and all of these are for the $|\Omega| = 100$ regime. For Model II, there is not even a single rejection. This again provides a validation to our observation that no single rule performs better than the others. Third, we repeat these experiments using a time limit that is an order of magnitude larger than our previous limit; i.e., in Algorithm 1 we set $time = 41000$ seconds. Further, in this setup, we solve the first two iterations of Algorithm 1 with a time limit of 5000 seconds each, and allow all other iterations a time limit of 15000 seconds. Thus, we are ensured that at least five iterations of Algorithm 1 are completed for each of the six rules. We summarize these results in Table 6.

(r,s)	$\lvert\Omega\rvert$				(r,s)	$\lvert\Omega\rvert$				(r,s)	$\lvert\Omega\rvert$			
	100	600	900	1500		100	600	900	1500		100	600	900	1500
(I,II)	✗	✗	✗	✗	(III,I)	✗	✗	✗	✗	(V,I)	✓	✗	✗	✗
(I,III)	✗	✗	✗	✗	(III,II)	✗	✗	✗	✗	(V,II)	✓	✗	✗	✗
(I,IV)	✗	✗	✗	✗	(III,IV)	✗	✗	✗	✗	(V,III)	✓	✗	✗	✗
(I,V)	✗	✗	✗	✗	(III,V)	✗	✗	✗	✗	(V,IV)	✓	✗	✗	✗
(I,VI)	✗	✗	✗	✗	(III,VI)	✗	✗	✗	✗	(V,VI)	✗	✗	✗	✗
(II,I)	✗	✗	✗	✗	(IV,I)	✗	✗	✗	✗	(VI,I)	✓	✗	✗	✓
(II,III)	✗	✗	✗	✗	(IV,II)	✗	✗	✗	✗	(VI,II)	✓	✓	✗	✓
(II,IV)	✗	✗	✗	✗	(IV,III)	✗	✗	✗	✗	(VI,III)	✓	✗	✗	✓
(II,V)	✗	✗	✗	✗	(IV,V)	✗	✗	✗	✗	(VI,IV)	✓	✗	✗	✓
(II,VI)	✗	✗	✗	✗	(IV,VI)	✗	✗	✗	✗	(VI,V)	✓	✓	✗	✓

(a) Model I

(r,s)	$\lvert\Omega\rvert$				(r,s)	$\lvert\Omega\rvert$				(r,s)	$\lvert\Omega\rvert$			
	100	600	900	1500		100	600	900	1500		100	600	900	1500
(I,II)	✓	✗	✗	✗	(III,I)	✓	✗	✗	✗	(V,I)	✗	✗	✗	✗
(I,III)	✗	✗	✗	✗	(III,II)	✓	✗	✗	✗	(V,II)	✓	✗	✗	✗
(I,IV)	✗	✗	✗	✗	(III,IV)	✗	✗	✗	✗	(V,III)	✗	✗	✗	✗
(I,V)	✗	✗	✗	✗	(III,V)	✓	✗	✗	✗	(V,IV)	✗	✗	✗	✗
(I,VI)	✗	✗	✗	✗	(III,VI)	✗	✗	✗	✗	(V,VI)	✗	✗	✗	✗
(II,I)	✗	✓	✓	✓	(IV,I)	✓	✗	✗	✗	(VI,I)	✓	✗	✗	✗
(II,III)	✗	✓	✓	✓	(IV,II)	✓	✗	✗	✗	(VI,II)	✓	✗	✗	✗
(II,IV)	✗	✓	✓	✓	(IV,III)	✗	✗	✗	✗	(VI,III)	✗	✗	✗	✗
(II,V)	✗	✓	✓	✓	(IV,V)	✓	✗	✗	✗	(VI,IV)	✗	✗	✗	✗
(II,VI)	✗	✓	✓	✓	(IV,VI)	✗	✗	✗	✗	(VI,V)	✓	✗	✗	✗

(b) Model II

Table 6: Analogous results to Table 5 for $\varepsilon = 0.05$ but with a time limit of $time = 41000$ in Algorithm 1.

Compared to Table 5, we now have seven additional instances in Table 6(a) for Model I where we can reject the null hypothesis. For Model II, there are 16 such additional instances in Table 6(b). This supports our premise that the previous time limit was not sufficiently high enough to allow for rejections. For Model I in Table 5(a), Rule VI is statistically the best performer, however only for the smallest scenario regime of $|\Omega| = 100$. Now—given additional time as in Table 6(a)—Rule VI has rejections in its favor compared to all other rules for two scenario regimes $|\Omega| = \{100, 1500\}$. For Model II, given this markedly larger amount of additional time, Rule II has statistically significant evidence to perform better than all other rules for all regimes except the smallest one of $|\Omega| = 100$. We examine this particular scenario regime in greater detail. Here, Algorithm 1 manages to complete between five and eight iterations for the different instances. However, as opposed to the larger instances where the first iteration typically provides the best bound, here the best iteration varies. This behavior is expected from smaller-sized

Lagrangian models as the bounds can improve in later iterations, especially when iterations are solved optimally. Observing the sets of instances for this regime, we find different rules performing better; thus, there is no statistical best performer for this regime.

Summarizing, the above results suggest that statistical evidence in favor of a step-size rule only appears when given a sufficiently high time limit. Even then, given differences in the structure of the underlying optimization models, different rules can perform better. Revisiting Q1, we can say that for applications where this large time limit is not a concern, our results suggest investing in the determination of a good step-size rule. Then, the answer to Q2 is that Rule II and Rule VI—although neither can be declared as the best—are strong candidates. Indeed, the constant step-length rule given by Rule II has been used in several applications; see, e.g., [26,35]. Polyak's step-size Rule VI, as we mention before, has also been widely used, as well as studied; see, e.g., [5, 22]. However, for several applications, the nearly half-a-day time limit we consider could be impractical. Then, our answer to Q2 changes. Here, our results suggest that a practitioner could well choose nearly any of the step-size rules and achieve results that have no statistical differences from other rules.

5.5 Analysis: comparison of a naive solution method and Algorithm 1

As we mention in Section 1, one of the goals of this work is to investigate whether a Lagrangian relaxation procedure using an appropriate step-size rule improves the computational tractability of model (1). In this section, we bring together the results and analysis of the previous sections and analyze in detail the output of Algorithm 1. We do so using the same batch of scenarios as those in Table 1. We choose r = Rule VI for the computational experiments in this section. Similar to the results in Section 5.4, we run Algorithm 1 with $\underline{\mathcal{Z}}$ set to the largest of three lower bounds of Table 3. Table 7 summarizes our results. Here, the columns "Objective" and "Gap" denote the two outputs of Algorithm 1, $\mathcal{Z}_D$ and δ, respectively. The "Improvement" column denotes the relative reduction in the optimality gap compared to the optimality gap obtained naively; i.e., $100\frac{\text{gap}_{\text{naive}}-\text{gap}_{\text{LR}}}{\text{gap}_{\text{naive}}}\%$, where $\text{gap}_{\text{naive}}$ and gap_{LR} are the optimality gaps in Table 1 and Table 7, respectively. Thus, a positive value of "Improvement" means Algorithm 1 succeeds in reducing the optimality gap compared to that obtained naively.

For Model I, except for the four instances in the first four rows of Table 7(a) and one additional instance of $|\Omega| = 1500$, $\varepsilon = 0.01$, Algorithm 1 always presents an improvement. For Model II there are a total of three such instances; all are in the $|\Omega| = 100$ regime. The instances of $|\Omega| = 100$ are the computationally least demanding instances; for Model I, all four of these instances are solved to optimality naively, see Table 1. Excluding these instances, the value of the algorithm is immediately visible in the larger and computationally more challenging instances. For Model I—see Table 7(a)—the average improvement over the 16 instances is 29.4%. For Model II—see Table 7(b)—the average improvement over the 16 instances is 12.3%.

Next, we revisit the sufficient conditions that we discussed in Section 2 for the BBP to hold with equality. These conditions ensure that the Lagrangian procedure identifies an optimal solution for model (1). For one of the 32 instances—which is also computationally the least expensive— the conditions do hold. This is the first row in Table 7(a) and Table 1(a) with an objective of 278.54. In this particular instance, the lower bound and the upper bounds from Algorithm 1 are identical (subject to a tolerance). Thus, there is no additional contribution from the BBP. Instead, if the lower bound was not tight, this contribution would have been immediately apparent. Then, we could conclude that the optimal objective function value of this instance is indeed that reported by Algorithm 1 even without knowledge of the lower bound. That being said, as we mention in Section 2 and as our computational experiments show, in practice, the BBP rarely holds with an equality.

Next, we address question Q1 from Section 1 that we reformulate as: does Algorithm 1, coupled with a suitable step-size rule and a suitable lower bounding heuristic, improve the optimality gaps as opposed to a naive solution method? In Table 7 we already provided evidence and conditions to answer this question in the affirmative with a "yes" for r = Rule VI. These results include a single batch of scenarios for the 16 instances each for Model I and Model II; thus, they are insufficient to make a statistical conclusion. We now repeat this experiment, as we did in Section 5.3 and Section 5.4, using the same 20 batches of scenarios for each of the four scenario-size regimes. We use $\varepsilon = 0.05$, the computational setup for Algorithm 1 as described in Section 5.1, and again set $\underline{\mathcal{Z}}$ as the maximum value obtained from the three lower bounding procedures of Section 3. We then apply a Welch t-test similar to that in Section 2 but

$\lvert\Omega\rvert$	ε	Objective	Time	Gap	Improvement
100	0.01	278.54	101	-	-
	0.03	284.30	69	1.81%	-
	0.05	290.47	135	3.45%	-
	0.07	296.99	63	4.47%	-
600	0.01	252.55	**T**	1.60%	76.64%
	0.03	273.42	**T**	7.86%	52.28%
	0.05	284.88	**T**	10.63%	44.64%
	0.07	291.25	**T**	11.27%	28.35%
900	0.01	264.84	**T**	7.53%	52.47%
	0.03	277.81	**T**	11.56%	41.70%
	0.05	286.26	**T**	11.37%	46.24%
	0.07	292.49	**T**	12.80%	45.04%
1500	0.01	266.12	**T**	11.56%	-11.48%
	0.03	279.52	**T**	14.33%	17.50%
	0.05	287.34	**T**	11.78%	37.01%
	0.07	293.32	**T**	13.09%	39.21%

(a) Model I

$\lvert\Omega\rvert$	ε	Objective	Time	Gap	Improvement
100	0.01	8,898.50	**T**	32.00%	-8.44%
	0.03	10,617.61	**T**	37.38%	-0.83%
	0.05	12,207.23	**T**	41.65%	4.16%
	0.07	14,632.89	**T**	47.92%	-0.19%
600	0.01	9,646.32	**T**	52.29%	1.91%
	0.03	11,334.21	**T**	56.05%	11.01%
	0.05	12,939.53	**T**	56.46%	15.67%
	0.07	14,556.63	**T**	57.79%	17.52%
900	0.01	9,819.28	**T**	53.69%	11.67%
	0.03	11,506.56	**T**	55.82%	23.33%
	0.05	13,147.37	**T**	57.39%	26.27%
	0.07	14,764.62	**T**	58.59%	24.93%
1500	0.01	9,904.72	**T**	53.52%	13.45%
	0.03	11,594.00	**T**	56.17%	15.36%
	0.05	13,228.31	**T**	58.09%	20.34%
	0.07	14,845.89	**T**	59.20%	21.02%

(b) Model II

Table 7: Computational results of Algorithm 1 on instances of model (1) using r = Rule VI and $\underline{\mathcal{Z}}$ = the maximum of the three lower bounds in Table 3. The "Gap" column denotes the relative optimality gap between $\underline{\mathcal{Z}}$ and the upper bound obtained using the Lagrangian procedure of Algorithm 1. The "Improvement" column denotes the relative reduction in the optimality gap as compared to Table 1. A "-" indicates that an improvement cannot be computed as the instance solved to optimality naively. For details, see Section 5.5.

with the null and alternate hypotheses modified for step-size rule r as follows:

$$H_0 : \mu_r \geq \mu_{\text{naive}} \qquad H_1 : \mu_r < \mu_{\text{naive}}, \tag{10}$$

here μ_r is the optimality gap reported by Algorithm 1 for step-size rule r and μ_{naive} denotes the optimality gap from the the naive solution method. Rejecting the null hypothesis suggests that there is sufficient evidence to conclude that Algorithm 1, with step size r, performs better than the naive solution method. As in equation (8), the test statistic T has a Student-t distribution with ν degrees of freedom, where

$$T = \frac{\overline{r} - \overline{\text{naive}}}{\sqrt{\frac{s_r^2}{20} + \frac{s_{\text{naive}}^2}{20}}}, \quad \text{where } T \sim t_\nu, \quad \text{with } \nu = \left\lfloor \frac{\left(\frac{s_r^2}{20} + \frac{s_{\text{naive}}^2}{20}\right)^2}{\frac{1}{19}\left(\frac{s_r^2}{20}\right)^2 + \frac{1}{19}\left(\frac{s_{\text{naive}}^2}{20}\right)^2} \right\rfloor. \tag{11}$$

In equation (11), $\overline{r}$ and $\overline{\text{naive}}$ denote the average gap over the 20 batches for step-size rule r and the naive solution method, respectively. Analogously, s_r and s_{naive} are the corresponding sample standard deviations of the gaps. For a 5% level of significance, we reject the null hypothesis if $T < t_{0.05,\nu}$, where $t_{0.05,\nu}$ denotes the 5-percentile quantile of the t-distribution with ν degrees of freedom. In Table 8,

we present the results of this statistical test; i.e., every entry results from the solution of 20 sets of optimization models.

r	$\lvert\Omega\rvert$			
	100	600	900	1500
I	✗	✓	✓	✓
II	✗	✓	✓	✓
III	✗	✓	✓	✓
IV	✗	✓	✓	✓
V	✗	✓	✓	✓
VI	✗	✓	✓	✓

(a) Model I

r	$\lvert\Omega\rvert$			
	100	600	900	1500
I	✓	✓	✓	✓
II	✗	✓	✓	✓
III	✗	✓	✓	✓
IV	✗	✓	✓	✓
V	✗	✓	✓	✓
VI	✗	✓	✓	✓

(b) Model II

Table 8: Statistical performance of step-size rules for Algorithm 1 for $\varepsilon = 0.05$. An entry of Rule r in the first column denotes that the null hypothesis is $H_0 : \mu_r \geq \mu_{\text{naive}}$. Rejection of the null hypothesis is denoted by a ✓, while a failure to reject is denoted by a ✗. For details, see Section 5.5.

Table 8(a) and Table 8(b) present consistent results for both Model I and Model II, respectively. For all scenario regimes, except $|\Omega| = 100$, for both models we have statistically sufficient evidence to reject the null hypothesis. These results provide further validation for our conclusions from Section 5.4, that the improvement of Algorithm 1 over the naive solution method is particularly noticeable in the computationally more intensive regimes; i.e., for these upper bounds, we answer Q1 with a convincing "yes". For computationally less intensive regimes, our results suggest there is no significant benefit to use a Lagrangian procedure; this is again intuitive. Further, in line with our conclusions in Section 5.4, our results suggest that there is an improvement regardless of the chosen step-size rule. This answers Q2. For computationally intensive regimes, a Lagrangian relaxation procedure is always effective over a naive solution, and that too for any step-size rule. However, we note with caution, as our results in Section 5.4 suggest, that the extent of this improvement is dependent not only on the chosen step-size rule but also on the underlying structure of the model.

6 Summary

We study a Lagrangian relaxation of a general two-stage chance-constrained optimization model. We show that there exist considerable differences between theory and practice, especially for computationally challenging instances of such models. Specifically, well-studied schemes to update step-sizes in the classical subgradient method do not always perform well in practice. To further validate this observation, we employ a series of statistical tests by solving batches of scenarios. In marked contrast to rules that are known to perform better than others in theory—for example in rates of convergence—our results show that no single rule can be uniformly declared as the best performer in practice. Nonetheless, our results demonstrate that such schemes show significant improvements over naive methods even when iterations in the Lagrangian procedure are solved suboptimally; i.e., our results are highly conservative. Future work could examine necessary or sufficient conditions to use the optimistic bound, rather than the conservative bound, resulting from the Lagrangian procedure.

We also study three lower bounds for such chance-constrained models. We find a quantile-based bound to be highly effective in achieving fast solutions, however regularization-based bounds are better when more compute time is available. We also find no benefit in constructing representative scenarios using hierarchical clustering methods over i.i.d. scenarios. This suggests another line of future work where customized aggregation schemes for chance-constrained programs could be developed. A third direction of future work is the development of iterative algorithms that compute lower bounds at each iteration of the Lagrangian procedure.

All our codes are publicly available at: https://github.com/charlotteritter/ArticleSubgradient.

Acknowledgements We gratefully acknowledge the compute resources and support provided by the Erlangen Regional Computing Center (RRZE). The authors acknowledge the financial support by the Federal Ministry for Economic Affairs and Energy of Germany in the project METIS (project number 03ET4064).

References

1. Ahmed, S.: Two-stage stochastic integer programming: a brief introduction. Wiley Encyclopedia of Operations Research and Management Science (2010). doi:10.1002/9780470400531.eorms0092
2. Ahmed, S., Luedtke, J., Song, Y., Xie, W.: Nonanticipative duality, relaxations, and formulations for chance-constrained stochastic programs. Mathematical Programming **162**(1-2), 51–81 (2017). doi:10.1007/s10107-016-1029-z
3. Ahmed, S., Shapiro, A.: Solving chance-constrained stochastic programs via sampling and integer programming. In: State-of-the-art decision-making tools in the information-intensive age, pp. 261–269. INFORMS (2008). doi:10.1287/educ.1080.0048
4. Ahmed, S., Xie, W.: Relaxations and approximations of chance constraints under finite distributions. Mathematical Programming **170**(1), 43–65 (2018). doi:10.1007/s10107-018-1295-z
5. Allen, E., Helgason, R., Kennington, J., Shetty, B.: A generalization of Polyak's convergence result for subgradient optimization. Mathematical Programming **37**(3), 309–317 (1987). doi:10.1007/BF02591740
6. Bertsekas, D.P.: Constrained optimization and Lagrange multiplier methods. Computer science and applied mathematics. Athena Scientific (1996)
7. Boyd, S., Boyd, S.P., Vandenberghe, L.: Convex optimization. Cambridge University Press (2004). doi:10.1017/CBO9780511804441
8. Charnes, A., Cooper, W.W.: Chance-constrained programming. Management Science **6**(1), 73–79 (1959). doi:10.1287/mnsc.6.1.73
9. d'Antonio, G., Frangioni, A.: Convergence analysis of deflected conditional approximate subgradient methods. SIAM Journal on Optimization **20**(1), 357–386 (2009). doi:10.1137/080718814
10. Deng, Y., Shen, S.: Decomposition algorithms for optimizing multi-server appointment scheduling with chance constraints. Mathematical Programming **157**(1), 245–276 (2016). doi:10.1007/s10107-016-0990-x
11. Held, M., Wolfe, P., Crowder, H.P.: Validation of subgradient optimization. Mathematical Programming **6**(1), 62–88 (1974). doi:10.1007/BF01580223
12. Hoffmann, M., Kotzur, L., Stolten, D., Robinius, M.: A review on time series aggregation methods for energy system models. Energies **13**(3), 641 (2020). doi:10.3390/en13030641
13. Hoffmann, M., Priesmann, J., Nolting, L., Praktiknjo, A., Kotzur, L., Stolten, D.: Typical periods or typical time steps? A multi-model analysis to determine the optimal temporal aggregation for energy system models. Applied Energy **304**, 117825 (2021). doi:10.1016/j.apenergy.2021.117825
14. Kotzur, L., Markewitz, P., Robinius, M., Stolten, D.: Impact of different time series aggregation methods on optimal energy system design. Renewable Energy **117**, 474–487 (2018). doi:10.1016/j.renene.2017.10.017
15. Küçükyavuz, S.: On mixing sets arising in chance-constrained programming. Mathematical Programming **132**(1-2), 31–56 (2010). doi:10.1007/s10107-010-0385-3
16. Larsson, T., Patriksson, M., Strömberg, A.B.: Conditional subgradient optimization — theory and applications. European Journal of Operational Research **88**(2), 382–403 (1996). doi:10.1016/0377-2217(94)00200-2
17. Luedtke, J.: A branch-and-cut decomposition algorithm for solving chance-constrained mathematical programs with finite support. Mathematical Programming **146**(1-2), 219–244 (2014). doi:10.1007/s10107-013-0684-6
18. Mak, W.K., Morton, D.P., Wood, R.K.: Monte Carlo bounding techniques for determining solution quality in stochastic programs. Operations research letters **24**(1-2), 47–56 (1999). doi:10.1016/S0167-6377(98)00054-6
19. Nedić, A., Ozdaglar, A.: Approximate primal solutions and rate analysis for dual subgradient methods. SIAM Journal on Optimization **19**(4), 1757–1780 (2009). doi:10.1137/070708111
20. Ozturk, U.A., Mazumdar, M., Norman, B.A.: A solution to the stochastic unit commitment problem using chance constrained programming. IEEE Transactions on Power Systems **19**(3), 1589–1598 (2004). doi:10.1109/TPWRS.2004.831651
21. Polyak, B.T.: A general method for solving extremal problems. Doklady Akademii Nauk **174**(1), 33–36 (1967)
22. Polyak, B.T.: Subgradient methods: a survey of Soviet research. Nonsmooth Optimization **3**, 5–29 (1978)
23. Polyak, B.T.: Introduction to optimization. Optimization Software (1987)
24. Rockafellar, R.T., Wets, R.J.B.: Scenarios and policy aggregation in optimization under uncertainty. Mathematics of Operations Research **16**(1), 119–147 (1991). doi:10.1287/moor.16.1.119
25. Rokach, L., Maimon, O.: Clustering methods. In: Data mining and knowledge discovery handbook, pp. 321–352. Springer (2005). doi:10.1007/0-387-25465-X_15

26. Sashirekha, A., Pasupuleti, J., Moin, N.H., Tan, C.S.: Combined heat and power (CHP) economic dispatch solved using Lagrangian relaxation with surrogate subgradient multiplier updates. International Journal of Electrical Power & Energy Systems **44**(1), 421–430 (2013). doi:10.1016/j.ijepes.2012.07.038
27. Shor, N.Z.: The rate of convergence of the generalized gradient descent method. Cybernetics and Systems Analysis **4**(3), 79–80 (1968). doi:10.1007/BF01073933
28. Singh, B., Knueven, B.: Lagrangian relaxation based heuristics for a chance-constrained optimization model of a hybrid solar-battery storage system. Journal of Global Optimization **80**(4), 965–989 (2021). doi:10.1007/s10898-021-01041-y
29. Singh, B., Morton, D.P., Santoso, S.: An adaptive model with joint chance constraints for a hybrid wind-conventional generator system. Computational Management Science **15**(3-4), 563—-582 (2018). doi:10.1007/s10287-018-0309-x
30. Singh, B., Watson, J.P.: Approximating two-stage chance-constrained programs with classical probability bounds. Optimization Letters **13**(6), 1403–1416 (2019). doi:10.1007/s11590-019-01387-z
31. Takriti, S., Birge, J.R.: Lagrangian solution techniques and bounds for loosely coupled mixed-integer stochastic programs. Operations Research **48**(1), 91–98 (2000). doi:10.1287/opre.48.1.91.12450
32. Tikhonov, A.N., Arsenin, V.Y.: Solutions of ill-posed problems. John Wiley & Sons, Washington, D.C. (1977)
33. Verweij, B., Ahmed, S., Kleywegt, A.J., Nemhauser, G., Shapiro, A.: The sample average approximation method applied to stochastic routing problems: a computational study. Computational Optimization and Applications **24**(2-3), 289–333 (2003). doi:10.1023/a:1021814225969
34. Watson, J.P., Wets, R.J.B., Woodruff, D.L.: Scalable heuristics for a class of chance-constrained stochastic programs. INFORMS Journal on Computing **22**(4), 543—554 (2010). doi:10.1287/ijoc.1090.0372
35. Yuan, D., Xu, S., Zhao, H.: Distributed primal–dual subgradient method for multiagent optimization via consensus algorithms. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) **41**(6), 1715–1724 (2011). doi:10.1109/tsmcb.2011.2160394