Final Technical Report: Sparse Grid Scenario Generation and Interior Algorithms for Stochastic Optimization in a Parallel Computing Environment

Award DE-SC0005102
PI: Sanjay Mehrotra,
Northwestern University

The support from this grant resulted in seven published papers and a technical report. Two papers are published in *SIAM J. on Optimization* [87, 88]; two papers are published in *IEEE Transactions on Power Systems* [77, 78]; one paper is published in *Smart Grid* [79]; one paper is published in *Computational Optimization and Applications* [44] and one in *INFORMS J. on Computing* [67]). The works in [44, 67, 87, 88] were funded primarily by this DOE grant. The applied papers in [77, 78, 79] were also supported through a subcontract from the Argonne National Lab.

We start by presenting our main research results on the scenario generation problem in Sections 1–2. We present our algorithmic results on interior point methods for convex optimization problems in Section 3. We describe a new 'central' cutting surface algorithm developed for solving large scale convex programming problems (as is the case with our proposed research) with semi-infinite number of constraints in Section 4. In Sections 5–6 we present our work on two application problems of interest to DOE.

Note that all figures and tables providing information on our research results are given in the appendix.

1 Scenario generation for stochastic optimization problems via the sparse grid method

The probability measure P of a stochastic optimization problem is approximated by a discrete one, with finite number of scenarios ξ^k ($k = 1, \dots, K$) and weights w^k, which amounts to the approximation of the integral required in computing the expected value by a finite sum:

$$\min_{x \in X} \int_{\Xi} f(\xi, x) P(d\xi) \approx \min_{x \in X} \sum_{k=1}^{K} w^k f(\xi^k, x). \tag{1}$$

The generation of scenarios that efficiently achieve good approximation in (1) is, thus, an important problem in stochastic optimization.

Summary. In [44], we showed that, under a regularity assumption on the random function involved, the sequence of optimal solutions of the sparse grid approximations converges to the true optimal solutions as the number of scenarios increase. The rate of convergence is established and shown to be determined by the rate of convergence in sparse grid approximation of the integration of a function. We treat separately the special case when the underlying distribution is an affine transform of a product of univariate distributions, such as the multinormal distribution, and show how the sparse grid method can be adapted to the distribution by the use of quadrature formulas tailored to the distribution. Numerical comparison of the performance of the sparse grid method using different quadrature rules with classic quasi-Monte Carlo (QMC) methods, optimal rank-one lattice rules, and Monte Carlo (MC) scenario generation, using a series of utility maximization problems with up to 160 random variables, show that the sparse grid method is very efficient when the integrand is sufficiently smooth. In such problems the sparse grid scenario generation method is found to need several orders of magnitude fewer scenarios than MC and QMC scenario generation to achieve the same accuracy. It is indicated that the method appears scalable to problem with thousands of random variables. However, it has its limitations due to the generation of 'negative weights' in the quadrature formula.

Two variants of a sparse grid scenario generation method are proposed for the solution of stochastic optimization problems. Neither method requires knowledge of the exact distribution; it is sufficient that the moments up to a high enough order are known. However, to simplify the presentation we shall always assume that we have a known distribution. The following theorem on sparse grid approximation of an integration problems is extended to the case of solving the stochastic optimization problems.

Theorem 1.1. *([93], [60]) Consider the functional space*

$$\mathcal{W}_n^r := \left\{ f : \Omega^n \to \mathbb{R}, \max_{\|s\|_\infty \leq r} \|D^s f\|_\infty < \infty \right\}, \tag{2}$$

equipped with the norm $\|f\| = \max_{\|s\|_\infty \leq r} \|D^s f\|_\infty$. Assume that the chosen univariate quadrature rule satisfies $L(1) = 1$, $L(\nu) = O(2^\nu)$. For $n, r \in \mathbb{N}$, $f \in \mathcal{W}_n^r$, then K:=$|\mathcal{G}(q,n)| = O(2^q q^{n-1})$ is the cardinality of the sparse grid. Furthermore, for some $0 < c_{r,n} < \infty$ we have

$$\left| \int_{\Omega^n} f(\omega)\rho(\omega) d\omega - SG_q[f] \right| \leq c_{r,n} K^{-r} (\log K)^{(n-1)(r+1)} \|f\|. \tag{3}$$

The constant $c_{r,n}$ in Theorem 1.1 (though it may be exponential) depends only on dimension n, the order of differentiability r, and the underlying univariate quadrature rule used by the sparse grid method. For a given problem of dimension n, the integration error goes to zero fast for sufficiently differentiable functions since for $r \geq 2$ in (3) the term K^{-r} dominates $(\log K)^{(n-1)(r+1)}$. In comparison, the rate of convergence for classic QMC methods (including Halton, Sobol, or Niederreiter sequences) is of the order $O(K^{-1}(\log K)^n)$ for sufficiently "smooth" integrands, meaning integrands of bounded Hardy–Krause variation. Theorem 1.2 proved in our work presents a uniform convergence result for optimization using sparse grid approximations for functions with bounded weak derivatives; and a rate of convergence result for the sparse grid approximation is given in Theorem 1.3.

Theorem 1.2 (Convergence of the sparse grid method for stochastic optimization)**.** *Consider (**??**) and assume that $\mathcal{C}$ is closed and bounded; $f(\xi, x)\rho(\omega) \leq M < \infty$ for all $x \in \mathcal{C}$ and $\xi \in \Xi$; and that $\forall x \in \mathcal{C}$, $f(g(\cdot), x)$ is in the space $\mathcal{W}_n^r$, $1 \leq r < \infty$. Consider (1) where the scenarios are generated using the sparse grid method. Let x^K be a solution of (1), $K = 1, \ldots, \infty$, and z_K^* be the corresponding objective value. Then,*

(i) $z^* \geq \overline{\lim}_K z_K^*$.

*(ii) If $\{x^K\}_{K=1}^\infty$ has a cluster point $\hat{x}$, then $\hat{x}$ is an optimal solution of (**??**). Furthermore, for a subsequence $\{x^{K_t}\}_{t=1}^\infty$ converging to $\hat{x}$, $\lim_t z_{K_t}^* \to z^*$.*

Theorem 1.3 (Rate of convergence of sparse grid approximation for stochastic optimization)**.** *Consider (**??**) and its sparse grid approximation (1). Assume that $\mathcal{C}$ is closed and bounded, and that for every $x \in \mathcal{C}$ the function $f(\cdot, x)$ is bounded with $f(g(\cdot), x) \in \mathcal{W}_n^r$. Let x^* be an optimal solution of (**??**), and x^K be an optimal solution of (1). Then,*

$$\left| \sum_{k=1}^K w^k f(\xi^k, x^K) - \int_\Xi f(\xi, x^*) P(d\xi) \right| \leq \epsilon, \textit{ and} \tag{4}$$

$$\left| \int_\Xi f(\xi, x^K) P(d\xi) - \int_\Xi f(\xi, x^*) P(d\xi) \right| \leq 2\epsilon, \tag{5}$$

where

$$\epsilon = c_{r,n} K^{-r} (\log K)^{(n-1)(r+1)} \max_{x \in \mathcal{C}} \|f(g(\cdot), x)\|, \tag{6}$$

and $c_{r,n}$, K, $\mathcal{W}_n^r$, and $\|\cdot\|$ are defined as in Theorem 1.1.

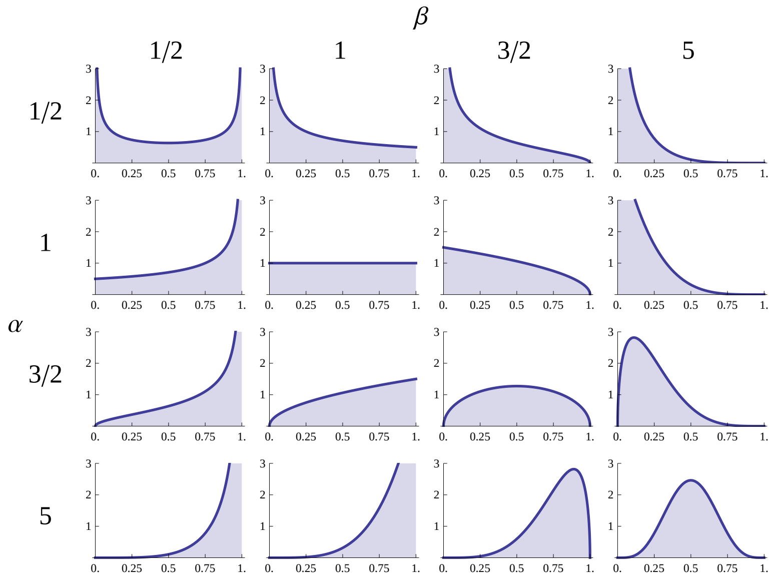


Figure 1: Shapes of Beta(α,β) distributions for $(\alpha, \beta) \in \{1/2, 1, 3/2, 5\}^2$.

We now illustrate that for sufficiently smooth (but not necessarily polynomial) integrands, sparse grid formulas with high degree of exactness provide a good approximation of the optimal objective function values of stochastic programs even for high-dimensional problems, regardless of the shape of the underlying distribution. For this purpose, we considered utility maximization examples of the form

$$\max_x \int_\Xi u(x^\mathbb{T}\xi)p(\xi)d\xi \quad \text{s.t.} \quad \|x\|_1 \leq 1, x \geq 0, \tag{7}$$

for different utility functions u and density functions p. The three utility functions considered were:

$$u_1(t) = -\exp(t) \qquad \text{(exponential utility),} \tag{8a}$$

$$u_2(t) = \log(1 + t) \qquad \text{(logarithmic utility), and} \tag{8b}$$

$$u_3(t) = (1 + t)^{1/2} \qquad \text{(power utility).} \tag{8c}$$

The probability densities considered were product Beta distributions, obtained by taking the product of univariate Beta(α,β) distributions with $\alpha, \beta \in \{1/2, 1, 3/2, 5\}$ (see Figure 1). The motivation behind this choice is that it allows us to experiment with distributions of various shapes, and also to transform the problem into product form, for which formulas with different degrees of polynomial exactness can be created and compared. We compared both variants of the sparse grid method: we used GKP formulas transformed with the appropriate diffeomorphism to scenarios for integration with respect to the product Beta distribution (or *transformed GKP* formulas for short) and sparse grid formulas using the Patterson-type quadrature rules derived for the Beta distribution (or *Patterson-type sparse grid* for short).

Table 1 shows the (estimated) optimal objective function values computed with different scenario generation techniques for the 100-dimensional exponential utility maximization (using u_1 from (8) in (7)), where p is the probability distribution function of the 100-fold product of the Beta(1/2,1/2) distribution (shown in the upper left corner of Figure 1). In this example the optimal objective function value can be computed relatively easily, because the objective function is the product of one-dimensional integrals; the optimal value is approximately 0.60690986. The table shows great difference in the rate of convergence of the different scenario generation methods. Monte Carlo (MC) integration achieves only 4 correct significant digits using 10^6 scenarios, quasi-Monte Carlo (QMC) integration with the Sobol sequence gets 6 digits with about

$2 \cdot 10^5$ scenarios, but only 3 digits with $2 \cdot 10^4$ scenarios. The optimal rank-one lattice rule achieves 4 correct digits with $2 \cdot 10^4$ scenarios, and one additional digit with $2 \cdot 10^5$ scnarios. In contrast, the transformed GKP rule gets 8 digits with $2 \cdot 10^4$ scenarios. (See column 4 in Table 1.) Finally, the Patterson-type sparse grid achieves 6 correct digits already with 201 scenarios. (Column 5.) The latter formula was created using a nested Patterson-type rule for the Beta($1/2$,$1/2$) distribution. We repeated the same experiment with all of the 16 distributions shown on Figure 1, with the same qualitative results, with the exception of the distribution Beta(1,1). The Beta(1,1) is the uniform distribution, hence the two sparse grid formulations are equivalent (and still outperform MC and QMC); the details are omitted for brevity.

We also considered examples with less regular distributions, using the same distributions from Figure 1 as components. The underlying 160-dimensional product distribution has ten components distributed as each of the distributions shown on Figure 1. The optimal objective function value is approximately 0.403148407; Table 2 shows the approximate objective function values computed with different techniques using up to half a million scenarios. The sparse grid formula using the Patterson-type rule achieves the same precision as QMC with an order of magnitude fewer points, reaching five correct digits with only 341 nodes. MC performs considerably worse than both of them, it needs about 500,000 scenarios to get the fourth significant digit correctly. Memory constraints prevented the solution of problems with more scenarios. We repeated the above experiments for the logarithmic and power utility maximization problem, that is, plugging u_2 and u_3 from (8) into (7), with the same experimental setup. The results were qualitatively very similar (see Tables 2–3 in the Appendix) to those in the previous section; we only present the detailed results of the 160-dimensional experiment involving the product Beta distribution with various parameters. We conclude that sparse grid scenario generation appears to be a promising alternative to classic quasi-Monte Carlo and Monte Carlo sampling methods, as well as to rank-one lattice rules for the solution of stochastic optimization problems whenever the integrands in the problem formulation are sufficiently smooth. The theoretical results on its efficiency, which state that the rate of convergence of the optimal objective value is the same as the rate of convergence of sparse grid formulas for integration, is complemented by excellent practical performance on a variety of utility maximization problems, which feature the expected values of smooth concave utility functions as objectives.

2 Generating moment matching scenarios using optimization techniques

Summary. Our efforts in overcoming the major shortcoming of sparse grid method that it generates negative weights corresponding to a scenario led to the development of a moment matching scenarios generation method for stochastic optimization [87]. Specifically, an optimization based method is proposed to generate moment matching scenarios for numerical integration and its use in stochastic programming. A major advantage of the moment matching scenario method is its flexibility: it can generate scenarios matching any prescribed set of moments of the underlying distribution rather than matching all moments up to a certain order, and the distribution can be defined over an arbitrary set. This allows for a reduction in the number of scenarios and allows the scenarios to be better tailored to the problem at hand. The method is based on a semi-infinite linear programming formulation of the problem that is shown to be solvable with polynomial iteration complexity. A practical column generation method is implemented. The column generation subproblems are polynomial optimization problems, however, they need not be solved to optimality. It is found that the columns in the column generation approach can be efficiently generated by random sampling. Extensive numerical experiments were used to compare the proposed method with Monte Carlo and quasi-Monte Carlo methods both on numerical integration problems and on stochastic optimization problems. The benefits of being able to match any prescribed set of moments, rather than all moments up to a certain order, is demonstrated using optimization problems with 100-dimensional random vectors. Empirical results show that the proposed approach outperforms Monte Carlo and quasi-Monte Carlo based approaches on the tested problems.

2.1 Moment matching scenario generation

Moment matching methods in stochastic programming were previously proposed in the context of sampling from partially specified distributions, where the goal is to efficiently sample some (unknown) distribution that has given moments up to a certain order (usually three or four). Such sampling methods in the past were only derived on a case-by-case basis. For instance in [82] the marginal distributions are given along with a covariance matrix, and the latter is used to determine a transformation which is then used to generate samples with the given marginals and the desired correlation. A similar approach is used in [66], where marginal moments up to the fourth order are matched along with the mixed second moments. The method in [66] has an additional drawback that it is not known to be convergent. The covariance matrix is matched and the first four marginal moments are approximated using semidefinite optimization in [50]. None of these methods can be generalized to match higher order moments, or any other set of moments.

The main motivation behind moment matching is the observation that cubature formulas with high degree of polynomial exactness should provide accurate estimates of the integrals of functions that can be approximated well by polynomials – this includes continuous functions on closed and bounded domains. In this section we summarize the rate of convergence of moment matching cubature formulas as a function of their modulus of smoothness proved in our paper [87]. We also showed that this rate of convergence cannot be improved by any other formula, aside from constants in this function, without additional assumptions on the integrand. We need the following definition.

Definition 2.1 (modulus of smoothness)**.** *The* modulus of smoothness *of a function $f\colon \Xi \mapsto \mathbb{R}$, denoted by ω_f is the function given by*

$$\omega_f(\delta) \overset{\text{def}}{=} \sup\left\{|f(x)-f(y)|\colon x\in\Xi, y\in\Xi, \|x-y\|\le\delta\right\}.$$

It is known that the error of the best polynomial approximation of every continuous f can be bounded by a function of ω_f as follows.

Theorem 2.1 (see, e.g. [22])**.** *Let $f\colon \Xi \mapsto \mathbb{R}$ be continuous on a compact set $\Xi \subseteq \mathbb{R}^n$. Then for every nonnegative integer d there is a polynomial P_d of degree d satisfying*

$$\sup_{\Xi} |f - P_d| \le C\omega_f(1/d),$$

where C is a positive constant depending only on Ξ, but not on d or f.

This rate of convergence (as the degree increases) is inherited by the error of cubature formulas.

Theorem 2.2. *Assume that the support $\Xi \subseteq \mathbb{R}^n$ of the probability measure μ is compact. Then there is a constant $\bar{C}$ (depending only on Ξ) such that for every continuous $f\colon \Xi\mapsto\mathbb{R}$ and for every cubature formula with nodes $\xi_1,\dots,\xi_K$, nonnegative weights $w_1,\dots,w_K$, and degree of exactness d,*

$$\left|\int_{\Xi} f(\xi)\mu(d\xi) - \sum_{k=1}^{K} w_k f(\xi_k)\right| \le \bar{C}\omega_f(1/d).$$

The rate of convergence of sequences of cubature formulas and scenario generation methods is usually expressed as a function of the number of nodes (or scenarios) K.

Theorem 2.3 (rate of convergence)**.** *Assume that the support $\Xi \subseteq \mathbb{R}^n$ of the probability measure μ is compact. Then for every K_0 there exist a cubature formula with $K \geq K_0$ nodes $\xi_1, \dots, \xi_K$ and nonnegative weights $w_1, \dots, w_K$ satisfying*

$$\left| \int_\Xi f(\xi)\mu(d\xi) - \sum_{k=1}^{K} w_k f(\xi_k) \right| \leq O(\omega_f(O(K^{-1/n})))$$

for every continuous function $f\colon \Xi \mapsto \mathbb{R}$. In particular, formulas with increasing polynomial exactness have this property.

The following theorem shows that this rate of convergence is essentially the best possible even when $\Xi = [0,1]^n$, and integration is with respect to the uniform distribution on Ξ.

Theorem 2.4. *Let $\Xi = [0,1]^n$ and $0 < C_1 < 1$ and $0 < C_2 < 2^{-n}$ be fixed constants. Then there exist no cubature formula on K nodes with positive weights that satisfies*

$$\left| \int_\Xi f(\xi) d\xi - \sum_{k=1}^{K} w_k f(\xi_k) \right| \leq C_1 \omega_f(C_2 K^{-1/n})$$

for every continuous f.

The estimates in Theorems 2.2 and 2.3 can be improved considerably for smooth functions.

Theorem 2.5. *Suppose that in Theorem 2.2 all rth order partial derivatives of f are continuous. Then the error bound of moment matching cubature formulas with degree of exactness d can be improved to*

$$\left| \int_\Xi f(\xi)\mu(d\xi) - \sum_{k=1}^{K} w_k f(\xi_k) \right| \leq \hat{C} d^{-r} \omega_f^{(r)}(1/d), \tag{9}$$

where

$$\omega_f^{(r)}(\delta) \overset{\text{def}}{=} \sup_{\substack{\rho \in \mathbb{N}^n \\ \sum_i \rho_i = r}} \omega_{D^\rho f}(\delta)$$

is the highest of the moduli of smoothness of the rth partial derivatives of f.

For functions of bounded rth partial derivatives, there exist moment matching formulas on K nodes, including those generated by the algorithms of Section 2.1.1, whose error (as a function of K) tends to zero at a rate $O(K^{-r/n})$.

2.1.1 A column generation approach for moment matching

We also proposed a new approach to construct cubature formulas that match any prescribed set of moments. This approach is based on a semi-infinite linear optimization formulation of the moment matching problem that allows the construction of formulas for a considerably larger variety of measures than the methods so far described. We shall also underline that the approach we are about to outline has more general applicability, as it can be used to find cubature formulas that give exact values in *any* given finite dimensional linear space of functions, not just in spaces of polynomials spanned by monomials. First, we need to introduce some notation.

For a point $x = (x_1, \dots, x_n) \in \mathbb{R}^n$, let $u_x \in \mathbb{R}^N$ denote the N-dimensional vector whose components are the monomials whose corresponding moments we are trying to match, in an arbitrary, but fixed, order (say, the graded lexicographic order). For example, if $n = 3$ and we want to match all moments up to order

$d = 2$, then the number of moments to match is $N = \binom{n+d}{d} = 10$, and the moments correspond to the monomials

$$u_x = (1, x_1, x_2, x_3, x_1^2, x_1x_2, x_1x_3, x_2^2, x_2x_3, x_3^2)^{\mathrm{T}},$$

(all the monomials in an arbitrary, but fixed, order) but if $n = 2$ and we want to match all mixed moments up to order 3 as well as marginal moments up to order 5, then we have $N = 14$, and

$$u_x = (1, x_1, x_2, x_1^2, x_1x_2, x_2^2, x_1^3, x_1^2x_2, x_1x_2^2, x_2^3, x_1^4, x_2^4, x_1^5, x_2^5)^{\mathrm{T}}.$$

Note that we may use bases of polynomials other than the monomial basis; in fact, in our computations we used Legendre polynomials to avoid numerical issues while trying to match high-order moments. In the sequel we will only assume (without loss of generality) that the components of u_x are linearly independent polynomials.

The cubature formula (ξ, w) matches all the required moments if and only if

$$\sum_{k=1}^{K} w_k u_{\xi_k} = m \overset{\text{def}}{=} \int_{\Xi} u_\xi \mu(d\xi), \tag{10}$$

with the integral on the right-hand side understood componentwise. The components of the vector $m \in \mathbb{R}^N$ are the required monomial moments of μ, ordered the same way as the components of u_x.

For fixed $\xi_1, \dots, \xi_K$, finding nonnegative weights to satisfy (10) is a linear programming (feasibility) problem with variables $w_1, \dots, w_K$. Hence, we can view the scenario generation problem as a semi-infinite linear programming (feasibility) problem, with a continuum of nonnegative variables indexed by the elements of $\mathbb{R}^n$ (or Ξ, if we are looking for an interior formula), and with N equality constraints. To motivate our approach, we rewrite this feasibility problem as an optimization problem using the notation $\alpha = \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{pmatrix}$:

$$\min_{w:\Xi\to\mathbb{R},\, \alpha\in\mathbb{R}^N} \left\{ \sum_{i=1}^{N} |\alpha_i| \;\middle|\; \int_{\Xi} w(\xi) u_\xi d\xi + \alpha = m;\ w(\xi) \geq 0\ \forall\, \xi \in \Xi \right\}, \tag{11}$$

whose optimal objective value is 0 if and only if all the required moments of the original distribution can be matched, that is, if m is indeed a vector of moments. The objective function $\sum_i |\alpha_i|$ may be replaced by any other norm of α. To keep the exposition simple we will continue using the L_1 norm.

To find the right (finite-dimensional) ξ in (10), we start with a candidate set of nodes $\{\xi_1, \dots, \xi_\ell\}$, possibly empty, and solve the auxiliary LP

$$\min_{w\in\mathbb{R}^\ell,\, \alpha\in\mathbb{R}^N} \left\{ \sum_{i=1}^{N} |\alpha_i| \;\middle|\; \sum_{k=1}^{\ell} w_k u_{\xi_k} + \alpha = m,\ w \geq 0 \right\}. \tag{12}$$

If the optimal objective function value is 0, with optimal solution $(\alpha^* = 0, w^*)$, then the cubature formula (ξ, w^*) is a positive formula matching all the desired moments. Otherwise, we can find a point $\xi_{\ell+1}$ with strictly negative reduced cost, and add it to the candidate node set.

Before discussing strategies to find the next node to add to the formula, we make two observations which are stated in the following theorems.

Theorem 2.6. *For every (not necessarily probability) measure μ there exists a positive interior cubature formula with degree of exactness d with respect to μ on $N(n, d) = \binom{n+d}{d}$ points. More generally, if an arbitrary collection of N moments are to be matched, there exists a positive interior cubature formula on N points that matches those moments.*

Also note that if $\{\xi_1, \ldots, \xi_K\}$ is the node set of some positive interior cubature formula, then the convex polytope

$$\left\{ w \geq 0 \,\middle|\, \sum_{k=1}^{K} w_k u_{\xi_k} = m, \right\}$$

is non-empty, and has a vertex (basic feasible solution), which has at most as many non-zero components as the number of equality constraints, which is N. Hence, an N-node formula can be obtained from *every* K-node formula with $K > N$ by solving a single linear programming problem.

Our next observation is that the ellipsoid method can be used to solve our moment matching semi-infinite linear programming formulation using a polynomial number of iterations.

Theorem 2.7. *Suppose we are given an oracle that finds a node $\xi_{\ell+1}$ with strictly negative reduced cost, given the nodes $\{\xi_1, \ldots, \xi_\ell\}$ and the optimal solution to the corresponding auxiliary linear program (12). Using this oracle, a positive formula matching all required moments with absolute precision ε can be found in oracle-polynomial time; here "polynomial" means polynomial jointly in $\log(1/\varepsilon)$ and N.*

In our implementation the auxiliary LP (12) is solved using the simplex method. If it is not ε-optimal, the oracle provides the new column $\xi_{\ell+1}$ to be added to (12), after which the primal simplex method can be used to resolve the auxiliary LP starting from the previous dual feasible solution. The column generation oracle is solved using an internal-MC or QMC procedure. We also found that approximation of smooth stochastic optimization problems using moment matching scenarios was significantly better than that from MC and QMC scenario based approximation. The results for numerical integration for standard test functions and stochastic optimization problems are shown in Figures 2–4 and Table 4.

3 Solution of Monotone Complementarity and General Convex Programming Problems Using a Modified Potential Reduction Interior Point Method

Summary. In [67] we developed a homogeneous algorithm equipped with a modified potential function for the monotone complementarity problem. We showed that this potential function is reduced by at least a constant amount if a scaled Lipschitz condition (SLC) is satisfied. A practical algorithm based on this potential function is implemented. This implementation maintains global linear and polynomial time convergence properties while achieving practical performance. It either successfully solves the problem, or concludes that the SLC is not satisfied for the user specified parameter. When compared with a mature software package `MOSEK` (barrier solver version `6.0.0.106`), our solver solves convex quadratic programming problems, convex quadratically constrained quadratic programming problems, and general convex programming problems in fewer iterations. Moreover, several problems for which `MOSEK` fails are solved to optimality. We also find that our approach detects infeasibility more reliably than general nonlinear solvers `Ipopt` (version `3.9.2`) and `Knitro` (version `8.0`).

Let us consider a monotone complementarity problem (MCP) of the following form

$$s = f(x) \tag{13}$$

$$0 \leq s \perp x \geq 0, \tag{14}$$

where $x, s \in \mathbb{R}^n$, $f(x)$ is a continuously differentiable monotone mapping from $\mathbb{R}^n_+ := \{x \in \mathbb{R}^n \mid x \geq 0\}$ to $\mathbb{R}^n$, and the notation $0 \leq s \perp x \geq 0$ means that $(x, s) \geq 0$ and that $x^T s = 0$. This requirement is called the complementarity condition. In monotone mapping, for every $x^1, x^2 \in \mathbb{R}^n_+$, we have

$$(x^1 - x^2)^T (f(x^1) - f(x^2)) \geq 0. \tag{15}$$

Let $\nabla f(x)$ denote the Jacobian matrix of $f(x)$. If $\nabla f(x)$ is positive semidefinite for all $x > 0$, that is,

$$h^T \nabla f(x) h \geq 0, \forall x > 0, h \in \mathbb{R}^n,$$

then $f(x)$ is a continuous monotone mapping.

Now, we consider an augmented homogeneous model related to MCP (HMCP):

$$s = \tau f(x/\tau), \tag{16}$$

$$\kappa = -x^T f(x/\tau), \tag{17}$$

$$0 \leq (s, \kappa) \perp (x, \tau) \geq 0. \tag{18}$$

For simplicity in the following we let $\bar{n} := n + 1, x := (x, \tau) \in \mathbb{R}^{\bar{n}}_+$ and $s := (s, \kappa) \in \mathbb{R}^{\bar{n}}_+$. Let

$$r^k := s^k - F(x^k),$$

and $(x^0, s^0) = e$ be the starting point. Let $\mathcal{C}(\hat{\mu})$ denote a continuous trajectory such that

$$\mathcal{C}(\hat{\mu}) := \left\{ (x^k, s^k) \mid s^k - F(x^k) = \hat{\mu} r^0, X^k s^k = \hat{\mu} e, 0 < \hat{\mu} \leq 1 \right\}.$$

Note that such a trajectory always exists (Yoshise [114]). Moreover, Andersen and Ye [19, Theorem 2] showed that this continuous trajectory is bounded and any limit point is a maximal complementary solution for HMCP. At iteration k with iterate $(x^k, s^k) > 0$, the algorithm computes the search direction (d_x, d_s) by solving the following system of linear equations:

$$d_s - \nabla F(x^k) d_x \;=\; -\eta r^k, \tag{19}$$

$$X^k d_s + S^k d_x \;=\; \gamma \mu^k e - X^k s^k. \tag{20}$$

Here η and γ are parameters between 0 and 1, and $\mu^k := \frac{(x^k)^T s^k}{\bar{n}}$. For a step size $\alpha > 0$, let the new iterate be

$$\begin{aligned} x^+ &:= x^k + \alpha d_x > 0 \text{ and} \\ s^+ &:= s^k + \alpha d_s + (F(x^+) - F(x^k) - \alpha \nabla F(x^k) d_x)) = F(x^+) + (1 - \alpha\eta) r^k > 0. \end{aligned} \tag{21}$$

We consider the following potential function for the HMCP:

$$\Phi_\rho(x, s) := (\rho/2) \log \left\{ (x^T s)^2 + \theta \|r\|^2 \right\} - \sum_{j=1}^{\bar{n}} \log x_j s_j, \tag{22}$$

where θ is a constant parameter with positive value and $\rho \geq \bar{n} + \sqrt{\bar{n}}$. A potential function of this form was introduced to the homogeneous LP model by Mehrotra and Huang [84]. Without loss of generality we choose $\theta = 1$ in (22).

In [67], we show that if f satisfies the SLC (and consequently so does F), then at iteration k the theoretical direction (d_x, d_s) with an appropriate step size α results in

$$\Phi_\rho(x^+, s^+) - \Phi_\rho(x^k, s^k) \leq \zeta,$$

where ζ is a negative constant. We show that if $\Phi_\rho(x^k, s^k)$ is reduced by at least a constant amount at each iteration and $\Phi_{\bar{n}}(x^k, s^k)$ (potential function (22) with parameter $\rho = \bar{n}$) is upper bounded by a constant amount with some mild assumptions, then the interior point algorithm generates a maximal complementary solution of desired precision in polynomial time.

All computations were performed on a 3.2 GHz Intel Dual-Core CPU machine with 4GB RAM. The program is run on one CPU only. An `AMPL` [2] interface is implemented in our implementation to read the `AMPL` nonlinear models and the first and second-derivative calculations. All problems are solved using the same default parameter settings.

3.1 Computational results on feasible problems

We solved the convex QP problems from Maros and Mézáros's QP test set [83], the convex QCQP problems from Mittelmann's QCQP test set [8] and Vanderbei's AMPL nonlinear Cute and Non-Cute set [3], the general convex CP problems from Vanderbei's AMPL nonlinear test set [3], and from Leyffer's mixed-integer nonlinear test set [9] while ignoring the integrality requirements.

Tables 5–8 provide a comparison between the MOSEK homogeneous interior point optimizer and our implementation under their default settings. We focus on the number of iterations needed to achieve the desired accuracy by both solvers.

Table 5 gives the computational results for solving 127 QP problems. The average number of iterations used by our implementation (15.92 iters) and MOSEK (16.22 iters) are comparable. All problems can be solved within 50 iterations by both solvers. Note that for three problems (liswet4, powell20, and qpilotno) MOSEK terminates with NEAR_OPTIMAL status. This implies MOSEK cannot compute a solution that has the prescribed accuracy.

Table 6 gives the computational results for solving Mittelmann's QCQP test set. Here two different average performance results are provided. "Avg1" provides the average performance on problems for which MOSEK terminates with OPTIMAL, NEAR_OPTIMAL or UNKNOWN status. For problems QQ-aug2dqp and QQ-powell20, MOSEK is not able to converge to the optimal solution within 400 iterations. We exclude both problems in "Avg1". For problems QQ-laser, QQ-qforplan, and QQ-qseba, MOSEK requires more than 50 interior point iterations to successfully solve. We recompute the average performance statistics again in "Avg2" by further excluding these three examples. Considering average performances in these experiments, our implementation (Avg1=16.43 iters, Avg2=16.26 iters) is significantly better than MOSEK (Avg1=20.91 iters, Avg2=18.34 iters). Note that for QQ-liswet1, QQ-liswet7–QQ-liswet12, MOSEK terminates with UNKNOWN status. This may happen when MOSEK converges slowly, and hence the primal, dual, or gap residuals may not attain the prescribed accuracy. On the other hand, our implementation is able to terminate with OPTIMAL status for all the test problems within 40 iterations.

Table 7 contains the computational results for solving 11 QCQP problems in Vanderbei's AMPL nonlinear test set. Comparing the size to those in Mittelmann's QCQP test set, the problems here are relatively small, but may include a quadratic objective term and more than one quadratic constraint. Considering the average iterations required for solving the problems, both solvers generate similar computational results (MOSEK Avg=11.36 iters versus our implementation Avg=11.64 iters).

Table 8 contains the computational results for solving 46 general CP problems. Thirty five of the CP problems are selected from Vanderbei's AMPL nonlinear Cute and Non-Cute set, and 11 of the CP problems are selected from Leyffer's mixed-integer nonlinear test set. Two different average performance statistics are provided. "Avg1" provides the average performance on all the instances, except problem dallass, for which MOSEK fails to converge to an optimal solution within 400 iterations. For problems dallasm, dallasl, and elena-s383, MOSEK respectively requires 128, 74, and 85 iterations before terminating with OPTIMAL status. Thus we recompute the average statistics again in "Avg2" by additionally excluding these three problems. Comparing "Avg1", the average number of iterations used by our implementation (14.00 iters) is better than those used by MOSEK (18.46 iters). By excluding the problems for which MOSEK requires more than 50 iterations to solve ("Avg2"), the performance of both solvers becomes comparable, though our implementation (12.69 iters) still performs slightly better than MOSEK (13.07 iters). Note that for problems antenna-antenna2, firfilter-fir_convex, and wbv-lowpass2, MOSEK terminates with NEAR_OPTIMAL status.

The average number of directions used by our implementation is 1.93, i.e., the corrector direction was not used in about 7% of the iterations. It implies that the average proportion of the computations for the theoretical direction is not significant. We observe that for about 11% problems, the value of the potential function cannot be decreased monotonically. In fact, it increased in 5.85% iterations. We also note that the average proportion of the effort used to evaluate the potential function is 3.1%, suggesting that the computational demands for evaluating the potential function is not significant.

Figure 5 shows the performance profiles for feasible problems comparing the iteration performances of our implementation and `MOSEK`.

3.2 Computational results on infeasible problems

We now discuss the performance of our implementation for solving QCQP and CP infeasible problems. With the lack of an infeasible test set in the public domain, we create infeasible test problems by adding invalid objective constraints to the feasible problems.

Let x^* be an optimal solution of the convex program. We created a corresponding infeasible problem that has the form:

$$\begin{aligned} \min \quad & c(x) \\ \text{s.t.} \quad & a_i(x) \geq 0, i = 1, \ldots, \hat{m}, \\ & a_i(x) = 0, i = \hat{m}+1, \ldots, m, \\ & c(x) \leq c(x^*) - c^*, \\ & \hat{x} \geq 0, \text{ and } \tilde{x} \text{ is free}, \end{aligned} \tag{23}$$

where c^* is a positive constant. We chose

$$c^* := \begin{cases} 2|c(x^*)|, & \text{if } |c(x^*)| \geq 100, \\ 100, & \text{if } |c(x^*)| < 100. \end{cases}$$

Hence, the problems are made significantly infeasible.

Using this construction we create 127 infeasible QCQP problems from Maros and Mézáros's QP test set [83], 11 QCQP infeasible problems from Vanderbei's `AMPL` nonlinear `Cute` and `Non-Cute` set [3], 35 CP infeasible problems from Vanderbei's `AMPL` nonlinear `Cute` and `Non-Cute` set [3], and 11 CP infeasible problems from Leyffer's mixed-integer nonlinear test set [9] without integrality requirements.

We focus on the number of iterations needed to detect infeasibility by both solvers. Table 9 contains the computational results for solving 127 QCQP infeasible problems created from Maros and Mézáros's QP test set. For 38 problems `MOSEK` fails to detect infeasibility within 400 iterations. For problems `iQQ-qscagr25` and `iQQ-qstandat`, `MOSEK` terminates with `UNKNOWN` status. We compute the average iterations in "Avg1" by excluding these 38 examples. Comparing "Avg1", our implementation (28.15 iters) requires approximately 30% fewer iterations than `MOSEK` (42.20 iters). This is because `MOSEK` requires more than 60 iterations to detect the infeasibility in 12 problems. We further exclude these 12 problems and recompute the average iterations in "Avg2". Comparing "Avg2", the performance of our implementation (28.15 iters) is comparable to that of `MOSEK` (27.39 iters). However, our implementation is able to correctly detect infeasibility for each problem within 50 interior point iterations.

Table 10 has the computational results for solving the 11 QCQP infeasible problems created from Vanderbei's `AMPL` nonlinear test set. We provide the average statistics by excluding problem `ipolak4`, which `MOSEK` fails to solve within 400 iterations. Though the comparison shows that our implementation (24.60 iters) requires approximately 20% more iterations than `MOSEK` (20.30 iters), our implementation correctly solves all these infeasible problems.

Table 11 has the computational results for solving 46 CP infeasible problems created from Vanderbei's `AMPL` nonlinear `Cute` and `Non-Cute` set, and Leyffer's mixed-integer nonlinear test set. For problems `idallasl`, `idallasm`, and `idallass`, `MOSEK` receives function evaluate error from `AMPL`. Our implementation receives the same error while solving problem `idallasl`. In seven problems `MOSEK` fails to declare the infeasible status within 400 iterations. We compute the average statistics "Avg2" by excluding these 10 problems. Comparing "Avg1", MOSEK (66.33 iters) requires significantly more iterations than our implementation (19.44 iters). This happens because MOSEK requires more than 60 iterations to detect infeasibility in 13 problems whereas our implementation needs this in only one problem. We further exclude these 10 problems and recompute the average statistics in "Avg2". Comparing "Avg2", our implementation (16.04 iters) requires 60% more iteration than MOSEK (11.74 iters). This is because MOSEK is able to detect infeasibility during the preprocessing phase in 11 cases; on the other hand, our implementation-preprocessor does not detect any infeasibility. This highlights the importance of advanced preprocessing implementations.

Note that for problem antenna-iantenna2, our implementation requires 186 iterations to solve, which is significantly larger than the average iterations (28.07 iters). With fixed step factor parameter settings $(\theta_l, \theta_u) = (0.9, 0.9)$, our implementation can solve this problem in 46 iterations.

Observe that the average number of directions used for this problem is around 1.39, which is significantly smaller than that used for solving the feasible problems. Similarly, the value of the potential function cannot be decreased monotonically for 33.33% of the problems, which is also significantly larger than that used for solving the feasible problems. This suggests the importance of using the theoretical direction and potential function to decide the usefulness of a direction.

Figure 6 shows the performance profiles for the infeasible problems comparing the iteration performances of our implementation and MOSEK. Note that problems which our implementation or MOSEK fail to solve or solve during the preprocessing phase are not included in this figure.

3.3 Computational comparison with general nonlinear solvers

In this section we compare our implementation with general nonlinear solver Knitro and Ipopt on CP feasible and infeasible problems. The computational results on QP and QCQP problems are not provided because these problems are stored in MOSEK QPS format, which Ipopt and Knitro do not support. Note that Knitro provides three different algorithms for solving the nonlinear problems. In our comparison, we choose the direct primal-dual IPM. For Ipopt, we use the MA57 library for the sparse symmetric indefinite matrix factorization, which is the same in our implementation. Default parameter settings are used for both solvers except that the iteration limit is set to 400.

We first focus on the computational results for solving the CP feasible problems. The last two columns of Table 8 contain the number of iterations used by Ipopt and Knitro. Considering the average number of iterations used for solving these problems, our implementation (12.77 iters) ranked first, Knitro (18.95 iters) ranked second, whereas Ipopt (27.84 iters) ranked last. For problems dallasm and polak3, both Knitro and Ipopt received a function evaluation error from AMPL. Ipopt failed to solve dallass to optimality in 400 iterations. Observe that Ipopt requires more than 50 iterations for three problems and Knitro requires more than 50 iterations for five problems, whereas our implementation solves all feasible CP problems within 50 iterations.

Now we focus on the CP infeasible problems. Computational results from Ipopt and Knitro are reported in the last two columns of Table 11. Knitro failed to solve 13 problems within 400 iterations, whereas Ipopt fails in three problems. Knitro received an AMPL function evaluation error in three problems and Ipopt received the same error in four problems. Note that both Knitro and Ipopt terminate with a near optimal status in one problem. In three problems Ipopt reported error in feasibility restoration phase. Overall, Ipopt successfully detects the infeasibility in 35 problems and Knitro in 29 problems.

4 A cutting surface algorithm for semi-infinite convex programming

Summary. In [88] we present and analyze a central cutting surface algorithm for general semi-infinite convex optimization problems. The proposed method is applicable to problems with non-differentiable semi-infinite constraints indexed by an infinite-dimensional index set. Examples comparing the cutting surface algorithm to the central cutting plane algorithm of Kortanek and No demonstrate the potential of our algorithm even in the solution of traditional semi-infinite convex programming problems, whose constraints are differentiable, and are indexed by an index set. The central cutting surface algorithm is adapted to solve a family of DRO problems that are considerably more general than the ones proposed to date. We describe the cutting surface algorithm in some details as it may form a basis for algorithmic development in our proposed research.

Let us consider a general semi-infinite convex optimization problem of the following form:

$$\begin{aligned} \text{minimize} \quad & x_0 \\ \text{subject to} \quad & g(x,t) \le 0 \quad \forall\, t \in T \\ & x \in X \end{aligned} \tag{SICP}$$

with respect to the decision variables x (whose first coordinate is denoted by x_0), where the sets X and T, and the function/set $g\colon X \times T \mapsto \mathbb{R}$ satisfy suitable boundedness and slater assumptions. Our algorithm is motivated by the "central cutting plane" algorithm of [74] for convex problems, which in turn is an extension of Gribik's algorithm [63]. Gribik's algorithm has been the prototype of several cutting plane algorithms in the field, and has been improved in various ways, such as in the "accelerated central cutting plane" method of [35]. Our algorithm can also be viewed as a modification of a traditional convex constraint generation method, in which the restricted master problem attempts to drive its optimal solutions towards the center of the current outer approximation of the feasible set. The traditional constraint generation method is a special case of our algorithm with all centering parameters set to zero.

4.1 A central cutting surface algorithm for semi-infinite convex programming

The pseudo-code of our cutting surface algorithm is given in Algorithm **Central cutting surface algorithm**, which computes an ϵ-optimal solution to (SICP). Throughout the algorithm, $y^{(k-1)}$ is the best ϵ-feasible solution found so far (or the initial vector $y^{(0)}$), and its first coordinate, $y_0^{(k-1)}$, is an upper bound on the objective function value of the best ϵ-feasible point. The initial value of $y_0^{(0)}$ is an arbitrary upper bound U on this optimum and the other components of $y^{(0)}$ may be initialized arbitrarily.

In Step 2 of the algorithm we attempt to improve on the current upper bound by as much as possible and identify a "central" point $x^{(k)}$ that satisfies all the added inequalities with a large slack. The algorithm stops in Step 3 when no such improvement is possible.

In each iteration k, either a new cut is added in Step 5 that cuts off the last infeasible $x^{(k)}$ (a *feasibility cut*), or it is found that $x^{(k)}$ is an ϵ-feasible solution, and the best found ϵ-feasible solution $y^{(k)}$ is updated in Step 6 (an *optimality cut*). In either case, some inactive cuts are dropped in the optional Step 7. The parameter β adjusts how aggressively cuts are dropped; setting $\beta = \infty$ is equivalent to skipping this step altogether.

In Step 5 of every iteration k a centering parameter $s^{(k)}$ needs to be chosen. To ensure convergence of the method, it is sufficient that this parameter is bounded away from zero, and that it is bounded from above: $s_{\min} \le s^{(k)} \le B$ for every k, with some $s_{\min} > 0$. It is without loss of generality that we use the same upper bound as we used for the subgradient norms. Another strategy that ensures convergence is to find a subgradient $d \in \partial_x g(x^{(k)}, t^{(k)})$ and set $s^{(k)} = \alpha\|d\|$ with an arbitrary $\alpha \in (0, 1]$, which will give positive values for the centering parameter, but is not necessarily bounded away from zero. Below we prove that the central cutting surface algorithm converges in all of these cases.

Central cutting surface algorithm

Parameters: a strict upper bound U on the optimal objective function value of (SICP); a bound $B > 0$ for which the boundedness assumption on the feasible set holds; a tolerance $\epsilon \geq 0$ for which slater assumption holds; and an arbitrary $\beta > 1$ specifying how aggressively cuts are dropped.

Step 1. **(Initialization.)** Set $k = 1$, $y^{(0)} = (U, 0, \ldots, 0) \in \mathbb{R}^n$, and $J^{(0)} = \emptyset$.

Step 2. **(Solve master problem.)** Determine the optimal solution $(x^{(k)}, \sigma^{(k)})$ of the optimization problem

$$\begin{aligned} \text{maximize} \quad & \sigma \\ \text{subject to} \quad & x_0 + \sigma \leq y_0^{(k-1)} \\ & g(x, t^{(j)}) + \sigma s^{(j)} \leq 0 \quad \forall j \in J^{(k-1)} \\ & x \in X. \end{aligned} \tag{24}$$

Step 3. **(Optimal solution?)** If $\sigma^{(k)} = 0$, stop and return $y^{(k-1)}$.

Step 4. **(Feasible solution?)** Find a $t^{(k)} \in T$ satisfying $g(x^{(k)}, t^{(k)}) > 0$ if possible.
If no such $t^{(k)}$ is found, go to Step 6.

Step 5. **(Feasibility cut.)** Set $J^{(k)} = J^{(k-1)} \cup \{k\}$ and $y^{(k)} = y^{(k-1)}$; choose a centering parameter $s_{\min} \leq s^{(k)} \leq B$. (See the text for different strategies.)
Go to Step 7.

Step 6. **(Optimality cut; update best known ϵ-feasible solution.)** Set $J^{(k)} = J^{(k-1)}$ and $y^{(k)} = x^{(k)}$.

Step 7. **(Drop cuts.)** Let $D = \{j \mid \sigma^{(j)} \geq \beta\sigma^{(k)} \text{ and } g(x^{(k)}) + \sigma^{(k)} s^{(j)} < 0\}$, and set $J^{(k)} = J^{(k)} \setminus D$.

Step 8. Increase k by one, and go to Step 2.

5 Modeling transmission line constraints in two-stage robust unit commitment problem

Summary. Integration of renewable energy sources and demand response poses new challenges to system operators as they increase the uncertainty of the power supply and demand. Unit commitment is a central scheduling decision for the system operators in both regulated and de-regulated markets. The unit commitment model determines the schedule and generation level of each generator on the grid to minimize the system-wide operating cost while meeting the demand and satisfying various physical and contingency constraints of the power system. The unit commitment problem has been studied extensively in the literature. (See reference [95] for a survey of unit commitment problems.) Power system operators face new challenges with regard to the unit commitment problem as the uncertainty of supply and demand has increased and will continue to do so in the future. With efforts to integrate more renewable resources, such as wind and solar energy, into the power grid, the variability of the electricity supply is expected to increase. Also, it may be more difficult to predict the electricity demand accurately in the future as more consumers participate in the demand response program and more plug-in electric vehicles are introduced in the market.

A viable approach to deal with the uncertainty of the supply and demand is *stochastic programming*. Within this framework, one describes the uncertainty using a (joint) probability distribution, and then minimizes the sum of unit commitment cost and expected dispatch cost or considers chance constraints for the reliability criteria. Stochastic unit commitment problems in various settings are extensively studied in the literature. (See [48, 94, 105, 110, 113, 117] and references therein.) One of the challenges in stochastic unit commitment is that it may not be easy to obtain an accurate probabilistic description of the uncertain supply and demand. Moreover, one needs to generate a large number of samples to estimate the expectation, which can be a daunting task for a large-size power system.

The two-stage robust unit commitment problems have a master-and-subproblem structure, and references [33, 70, 97], and [70] use the Benders decomposition approach to take advantage of their structure. Zhao and Zeng [116] develop a column-and-constraint generation algorithm for the problems and show its computational efficiency over the Benders decomposition framework. (See reference [115] for the details of the column-and-constraint generation algorithm.)

The subproblem of the robust models is to find the worst-case scenario and corresponding recourse dispatch decision, and it is a form of separable bilinear program, which is $\mathcal{NP}$-hard, in general. Bertsimas et al. [33] find the local optimal solution of the subproblem using an outer-approximation algorithm. This method can support general polyhedral uncertainty sets. Other authors [70, 116] transform the separable bilinear subproblem into an equivalent mixed-integer linear program (MILP) using the *big-M* constraints after they represent the extreme points of the uncertainty set using a set of binary variables. Several authors [70, 97] solve the MILP subproblem without dual variables of hard constraints such as ramping and transmission line constraints (TLCs) or solve the bilinear version of the subproblem approximately through the *mountain climbing* procedure of Konno [73]. This approach provides a lower bound of the master problem. Statistical upper bounds of the master problem are obtained as well, using a Monte Carlo simulation. Zhao and Zeng [116] find the optimal solution of the MILP subproblem for a medium-sized unit commitment problem without TLCs. TLCs in the unit commitment problem increase the size of the problem substantially and hence make the robust counterpart difficult to solve. Zhao and Zeng [116] do not consider the TLCs in their robust unit commitment models. Although Bertsimas et al. [33] conduct numerical studies on the real-world-size problem, they include four representative transmission lines in their model. Other authors [70, 97] consider full TLCs, but relax them to obtain only a lower bound for the original formulation and find a statistical upper bound through a Monte Carlo simulation which may be lower than the true upper bound.

5.1 Robust unit commitment models

Deterministic unit commitment model. We formulate deterministic and robust unit commitment models. First, we review a deterministic unit commitment model with forecast demand and wind power generation based on a linear optimal power flow model with load-shift-factor (LSF) representation. We consider a closed power system that consists of thermal generators and wind power stations. Under the assumptions that 1) the optimal power flow is approximated by the lossless linear optimal power flow model, 2) the convex production cost is approximated by a piecewise linear function, and 3) the load and wind power generation are known in advance, we formulate a deterministic unit commitment problem as a MILP:

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{p}} \quad & \boldsymbol{c}^\top \boldsymbol{x} + \boldsymbol{b}^\top \boldsymbol{y} && (25a)\\ \text{s.t.} \quad & \boldsymbol{x} \in \mathcal{X} && (25b)\\ & \boldsymbol{F}\boldsymbol{x} + \boldsymbol{G}\boldsymbol{y} + \boldsymbol{H}\boldsymbol{p} \geq \boldsymbol{g} && (25c)\\ & \boldsymbol{E}\left(\boldsymbol{p} - \boldsymbol{d}_0 + \boldsymbol{w}_0\right) = \boldsymbol{0} && (25d)\\ & \boldsymbol{f}_{\text{min}} \leq \boldsymbol{L}\left(\boldsymbol{p} - \boldsymbol{d}_0 + \boldsymbol{w}_0\right) \leq \boldsymbol{f}_{\text{max}}. && (25e) \end{aligned}$$

The objective $\boldsymbol{c}^\top \boldsymbol{x}$ includes the start-up, shutdown and maintenance costs of the generators and $\boldsymbol{b}^\top \boldsymbol{y}$ sums up the production cost. (25c) includes constraints related to ramping limits, reserve requirements, reserve capacity, power output capacity, and piecewise linear representation of production cost function as well as non-negativity of such variables. (25d) represents the system balance constraints. The load-shift-factor matrix $\boldsymbol{L}$ represents the sensitivity of power flow with respect to nodal power injection, and (25e) denotes the TLCs.

Uncertainty set of the net demand. We consider two uncertain factors in the unit commitment model: load and wind power generation. We assume that the uncertain load and wind power generation lie in a polyhedral uncertainty set. We further assume that the uncertainty set $\mathcal{D}$ is bounded and all the extreme points of the uncertainty set can be parameterized by a set of binary variables and linear constraints:

$$\widehat{\mathcal{D}} = \left\{ \boldsymbol{U}\boldsymbol{z} + \boldsymbol{u} \mid \boldsymbol{V}\boldsymbol{z} = \boldsymbol{v}, \boldsymbol{z} \text{ binary} \right\}, \tag{26}$$

where $\widehat{\mathcal{D}}$ is the set of all extreme points of $\mathcal{D}$, and $\boldsymbol{U}$, $\boldsymbol{V}$, $\boldsymbol{u}$, and $\boldsymbol{v}$ are some constant matrix and vector parameters. For example, the extreme points of a box uncertainty set $\mathcal{D} = \{(\boldsymbol{d}, \boldsymbol{w}) \mid \boldsymbol{d}_{\text{min}} \leq \boldsymbol{d} \leq \boldsymbol{d}_{\text{max}}, \boldsymbol{w}_{\text{min}} \leq \boldsymbol{w} \leq \boldsymbol{w}_{\text{max}} \}$ can be represented as

$$\widehat{\mathcal{D}} = \left\{ \begin{array}{l} \left(\boldsymbol{d}_{\text{min}} + \operatorname{diag}\left(\boldsymbol{d}_{\text{max}} - \boldsymbol{d}_{\text{min}}\right) \boldsymbol{z}_{\boldsymbol{d}}, \right. \\ \qquad \left. \boldsymbol{w}_{\text{min}} + \operatorname{diag}\left(\boldsymbol{w}_{\text{max}} - \boldsymbol{w}_{\text{min}}\right) \boldsymbol{z}_{\boldsymbol{w}} \right) \\ \mid \boldsymbol{z}_{\boldsymbol{d}}, \boldsymbol{z}_{\boldsymbol{w}} \text{ binary} \end{array} \right\}. \tag{27}$$

Here, $\boldsymbol{U} = \operatorname{diag}\left(\left[(\boldsymbol{d}_{\text{max}} - \boldsymbol{d}_{\text{min}})^\top, (\boldsymbol{w}_{\text{max}} - \boldsymbol{w}_{\text{min}})^\top\right]\right)$ and $\boldsymbol{u} = [\boldsymbol{d}_{\text{min}}^\top, \boldsymbol{w}_{\text{min}}^\top]^\top$. $\boldsymbol{V}$ and $\boldsymbol{v}$ take zero.

Minimax-cost-based robust model The unit commitment decision is made before the uncertain wind power generation and load scenario resolves, while the dispatch is a recourse decision. This leads to the following two-stage robust unit commitment model with minimax criteria:

$$\begin{aligned} & \min_{\boldsymbol{x}} \boldsymbol{c}^\top \boldsymbol{x} + \max_{\boldsymbol{d} \in \mathcal{D}} Q\left(\boldsymbol{x}, \boldsymbol{d}\right) && (28a)\\ & \text{s.t. } \boldsymbol{x} \in \mathcal{X} && (28b)\\ & \qquad \mathcal{Y}(\boldsymbol{x}, \boldsymbol{d}) \neq \emptyset, \qquad \forall \boldsymbol{d} \in \mathcal{D} && (28c) \end{aligned}$$

where

$$Q\left(\boldsymbol{x},\boldsymbol{d}\right) := \min_{(\boldsymbol{y},\boldsymbol{p})\in\mathcal{Y}(\boldsymbol{x},\boldsymbol{d})} \boldsymbol{b}^{\top}\boldsymbol{y}, \tag{29}$$

and

$$\mathcal{Y}(\boldsymbol{x},\boldsymbol{d}) := \left\{ (\boldsymbol{y},\boldsymbol{p}) \;\middle|\; \begin{array}{r} \boldsymbol{G}\boldsymbol{y}+\boldsymbol{H}\boldsymbol{p} \geq \boldsymbol{g}-\boldsymbol{F}\boldsymbol{x} \\ \boldsymbol{E}\boldsymbol{p} = \boldsymbol{E}\boldsymbol{d} \\ \boldsymbol{f}_{\min} \leq \boldsymbol{L}\left(\boldsymbol{p}-\boldsymbol{d}\right) \leq \boldsymbol{f}_{\max} \end{array} \right\}. \tag{30}$$

The minimax-cost-based robust model finds the unit commitment decision that minimizes the sum of generator operating cost and the worst-case dispatch cost. The unit commitment decision is also required to meet the reliability condition (28c) that the dispatch problem is feasible for any scenario in the uncertainty set. If a unit commitment solution satisfies (28c), we say that the unit commitment is *robustly feasible*.

We compared the deterministic unit commitment model with the forecast load and wind power generation and the robust unit commitment model with the above box uncertainty set for 24 hour planning horizon. The solution of the robust unit commitment model has 78 more unit-hours committed and results in about 11% higher fixed unit commitment cost compared to one of the deterministic model. In the solution of the robust unit commitment model, more units were committed every hour and two of the cheap units are replaced by more expensive units during the night where there are relatively higher penetration level of wind power generation. The nominal objective for the robust unit commitment model, total operation cost with the solution of the robust unit commitment model under forecast load and wind power generation scenario, is only about 0.8% higher than the optimal objective value of the deterministic model. However, if the solution from the deterministic model is employed, there exists a load and wind power generation scenario within the uncertainty set that the dispatch is not feasible with respect to security constraints. The worst-case wind power generation scenario found in the robust unit commitment model is illustrated in Figure 7.

6 Robust distribution network reconfiguration

Summary. We now describe our research on the distribution network reconfiguration problem with uncertain demands, where the uncertainty of the demand arises from the daily fluctuations of the loads [79]. The distribution network reconfiguration problem is to configure the distribution network topology in order to improve the efficiency and stability of the network by changing the status of lines. The distribution network is composed of buses, where the power is injected (substation buses) or consumed; and lines or switches, which connect the buses. The distribution network has a meshed structure but is normally operated as radial (i.e. with no loop) so as to make the protection coordination easier (upstream to downstream) and to make the distribution design easier. We studied a two-stage robust optimization model for the distribution network reconfiguration problem with uncertain loads. In our two-stage robust optimization model, the network reconfiguration is the first-stage decision; and the optimal power flow becomes the second-stage decision which is made after the realization of the uncertain demand. The uncertainty set of the loads can be constructed as a set of possible load scenarios for a given planning horizon.

6.1 Two-stage robust distribution network reconfiguration model

Let us consider a deterministic distribution network reconfiguration problem in the following form:

$$\begin{aligned}
\min_{\boldsymbol{x},\boldsymbol{y}} \quad & \boldsymbol{c}^\top \boldsymbol{x} + \boldsymbol{b}^\top \boldsymbol{y} - \boldsymbol{e}^\top \boldsymbol{d}_0 && \text{(31a)}\\
\text{s.t.} \quad & \boldsymbol{x} \in X && \text{(31b)}\\
& \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{y} \geq \boldsymbol{f} && \text{(31c)}\\
& \boldsymbol{F}\boldsymbol{y} = \boldsymbol{d}_0 && \text{(31d)}\\
& \|\boldsymbol{G}_l \boldsymbol{y}\| \leq \boldsymbol{g}_l^\top \boldsymbol{y}, \qquad \forall l = 1, \cdots, m && \text{(31e)}
\end{aligned}$$

The vector $\boldsymbol{x}$ denotes the variables $\boldsymbol{\alpha}, \boldsymbol{\beta}$ regarding the distribution network configuration. The vector $\boldsymbol{y}$ represents the rest of the continuous variables related to optimal power flow of the distribution network. The vector $\boldsymbol{d}_0$ is the forecast loads. The term $\boldsymbol{c}^\top \boldsymbol{x}$ represents the switching cost and $\boldsymbol{b}^\top \boldsymbol{y} - \boldsymbol{e}^\top \boldsymbol{d}_0$ represents the cost associated with the power losses. $\mathcal{X}$ is the feasible set of radial distribution network configurations. (31d) summarizes the power balance equations, (31e) represents second-order-cone constraints, and (31c) denotes all the other optimal power flow constraints that are linear.

We assume that the network configuration decision is made prior to the realization of loads. For each realization of uncertain loads, we can compute the power losses for the given radial network configuration by solving the optimal power flow problem. We make the first-stage network configuration which minimizes the worst-case second-stage power loss. This can be formulated as the following two-stage robust model:

$$\begin{aligned}
& \min_{\boldsymbol{x}} \boldsymbol{c}^\top \boldsymbol{x} + \max_{\boldsymbol{d} \in \mathcal{D}} L(\boldsymbol{x}, \boldsymbol{d}) && \text{(32a)}\\
& \text{s.t. } \boldsymbol{x} \in \mathcal{X} && \text{(32b)}\\
& \quad \mathcal{Y}(\boldsymbol{x}, \boldsymbol{d}) \neq \emptyset, \qquad \forall \boldsymbol{d} \in \mathcal{D} && \text{(32c)}
\end{aligned}$$

where

$$L(\boldsymbol{x}, \boldsymbol{d}) := \min_{\boldsymbol{y} \in \mathcal{Y}(\boldsymbol{x}, \boldsymbol{d})} \boldsymbol{b}^\top \boldsymbol{y} - \boldsymbol{e}^\top \boldsymbol{d} \tag{33}$$

and

$$\mathcal{Y}(\boldsymbol{x}, \boldsymbol{d}) := \left\{ \boldsymbol{y} \;\middle|\; \begin{aligned} & \boldsymbol{B}\boldsymbol{y} \geq \boldsymbol{f} - \boldsymbol{A}\boldsymbol{x} \\ & \boldsymbol{F}\boldsymbol{y} = \boldsymbol{d} \\ & \|\boldsymbol{G}_l \boldsymbol{y}\| \leq \boldsymbol{g}_l^\top \boldsymbol{y}, \qquad \forall l = 1, \cdots, m \end{aligned} \right\} \tag{34}$$

Here, $\mathcal{D}$ is the uncertainty set of the loads. We assume that the uncertainty set of the loads is a polyhedron. The polyhedral assumption guarantees the finite convergence of the column-and-constraint generation algorithm that solves the proposed model [115], since the polyhedron has a finite number of extreme points. We further assume that we can characterize all the extreme points of the polyhedral uncertainty set using binary variables and linear constraints as in [77]. This assumption allows us to convert the bilinear subproblem into a mixed-integer linear program (MILP) as shown in the next section.

The innermost problem solves the optimal power flow of a given radial network configuration $\boldsymbol{x}$ with a given load scenario $\boldsymbol{d}$, which replaces the forecast load $\boldsymbol{d}_0$ to obtain the cost of power losses $L(\boldsymbol{x}, \boldsymbol{d})$. In the midlevel, the load scenario that maximizes the minimum cost of power losses $L(\boldsymbol{x}, \boldsymbol{d})$ for the given network configuration is obtained. In the outermost level, the robust program finds a network configuration that minimizes the sum of switching cost and worst-case cost of power losses. If a distribution network configuration $\boldsymbol{x} \in X$ satisfies (32c) (that is, given such a configuration, if the distribution network is feasible for any possible demand scenario $\boldsymbol{d} \in \mathcal{D}$), then we say the network configuration $\boldsymbol{x}$ is *robustly feasible*. If no *robustly feasible* configuration $\boldsymbol{x}$ exists (in other words, there is no configuration under which the distribution network is feasible for all possible demand scenario $\boldsymbol{d} \in \mathcal{D}$), then the robust program is infeasible. It is possible for the robust program to be infeasible, if the uncertainty set is too large and so there cannot be a configuration that is feasible for all possible scenarios in the uncertainty set. For such case, one can make the uncertainty set smaller, for example, by considering a shorter planning horizon. When there is no solution for the robust distribution network model, the distribution system would need to change the configuration more frequently to adapt to the rapidly changing load.

6.2 Solution Characteristics

We considered four test cases from the literature: 16-bus [46], 33-bus [24], 70-bus [49], and 94-bus[104] distribution network for our computational experiments (see Figure 8 for 94-bus system). A basic information of each the distribution network is given in Table 12. In our case study, the switching costs are ignored. We consider the following budget uncertainty set for the loads:

$$\mathcal{D} = \left\{ \boldsymbol{P} \middle| P_i^{\min} \leq P_i \leq P_i^{\max}, \forall i \in N \sum_{i \in N} \left[\left(\frac{P_i^D - P_i}{P_i^D - P_i^{\min}} \right)^+ \right] + \sum_{i \in N} \left[\left(\frac{P_i - P_i^D}{P_i^{\max} - P_i^D} \right)^+ \right] \leq B \right\}. \tag{35}$$

We assume the real loads can vary between the lower and upper limit $P_i^{\min}$ and $P_i^{\max}$. We further assumed that the overall variation is controlled by a user-defined budget parameter B. For simplicity, we assumed that the uncertain reactive loads will be proportional to the uncertain real loads where the ratio is presumed to be fixed. However, one can assume an independent uncertainty set for the reactive loads as well. To make the problem more tractable, we aggregated the load buses to make a zone. We assumed that the load variation (in percentage) of each bus within the same zone is identical. This approximation reduces the number of binary variables in the subproblem and the solution time. We assumed the lower and upper limit parameters of the uncertainty set to be $\pm$ 5% of the nominal value. For all cases, we aggregate load buses into 9 zones and assume the budget parameter B to be 8.

The solution of the robust distribution network reconfiguration problem is now compared with the solution of the deterministic model in Table 13. For 16-bus test case, the topology of the network is rather simple, and the configuration decision from the robust model is identical to the one from the deterministic model. But, for the other test cases, we observed that the two configurations are similar in terms of the total power losses under the forecast scenario; namely, nominal power losses in Table 13. However, the solution from the deterministic model is not robust, as there exists at least one demand scenario within the uncertainty set under which the delivery of power is not possible without violating the physical and operational constraints of the network. On the other hand, the robust model finds the configuration that is *robustly feasible*. The users can adjust the uncertainty set (35) by changing the value of the demand variation limit $P^i_{\text{max}}, P^i_{\text{min}}$ and the budget of the uncertainty parameter B. Table 14 shows the sensitivity of the minimum worst-case power losses of the robust distribution network reconfiguration problem with respect to the different variation limits of the uncertainty set of the 94-bus test case with $B = 8$. We can see that the worst-case power losses increases as the size of the uncertainty set gets bigger. When the uncertainty set allows a variation of more than 7% from the forecast demand, the robust program is infeasible and there is no radial configuration that is feasible in all possible scenarios. This implies that with the current design of the distribution network and operational requirements, it is impossible to meet all possible demand scenarios within the prescribed uncertainty set. Table 15 shows the worst-case power losses of the robust distribution network reconfiguration with the uncertainty sets with different values of the budget parameter B while the maximum demand fluctuation levels are set at $\pm$ 5%, where we observe the similar pattern.

A Appendix

A.1 Scenario generation for stochastic optimization problems via the sparse grid method

# nodes	MC	QMC (Sobol)	QMC (lattice)	SG (GKP)	Patterson-type SG
201	0.5861458741	**0.605**7992465	**0.60**960463896	**0.6069**012301	**0.606909**7420
20401	**0.6059**951776	**0.6059**86365	**0.6069**3318101	**0.60690986**45	**0.6069098**767
200000	**0.606**784018	**0.606909**7597	**0.60691**252536		
1000000	**0.6069**217022	**0.606909**7569	**0.60691**066152		

Table 1: Results from a 100-dimensional exponential utility maximization example using Beta distributions.

# nodes	MC	QMC (Sobol)	QMC (lattice)	SG (GKP)	Patterson-type SG
341	**0.40**20710589	**0.40**27479863	**0.40**507651399	**0.40**07432411	**0.403148**3327
58331	**0.4031**793283	**0.40314**48802	**0.4031**6298189	**0.402**8398967	**0.403148407**1
250000	**0.4031**696956	**0.40314**78092	**0.40315**112035		
500000	**0.4031**910684	**0.403148**3028	**0.40315**062416		

Table 2: Results from the 160-dimensional exponential utility maximization example using Beta distributions.

# nodes	MC	QMC (Sobol)	QMC (lattice)	SG (GKP)	Patterson-type SG
341	**-1.38**63232853	**-1.38**21077423	**-1.38**05140863	**-1.38**35622039	**-1.3816379**583
58331	**-1.381**7632120	**-1.3816**406975	**-1.38163**00171	**-1.38**00861876	**-1.3816379**399
250000	**-1.3816**977241	**-1.38163**82958	**-1.38163**62062		
500000	**-1.38163**63645	**-1.3816379**652	**-1.3816368**776		

Table 3: Results from the 160-dimensional power utility maximization example using Beta distributions.

A.2 Generating moment matching scenarios using optimization techniques

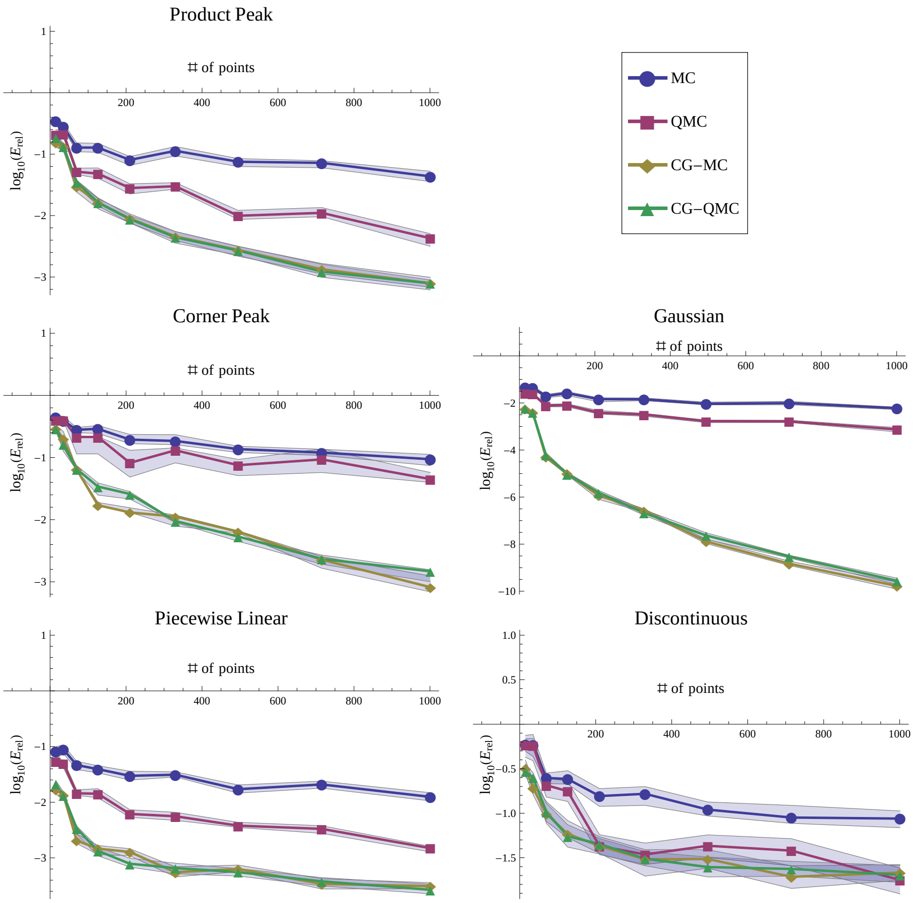


Figure 2: Performance of four cubature formulas using the four-dimensional parametric families of functions with uniform density. Horizontal axis: number of points used in the formulas; the points correspond to increasing degree of exactness of the CG-MC and CG-QMC formulas from 2 to 10. CG-MC implemented the column generation algorithm where the pricing problem in semi-infinite linear programming was solved using a Monte Carlo algorithm. CG-QMC implemented the column generation algorithm where the pricing problem in semi-infinite linear programming was solved using a Quasi-Monte Carlo (Sobol) algorithm. Vertical axis: base-10 logarithm of the median relative errors from 200 experiments. The gray shaded bands around the median relative errors are 0.95-level confidence intervals around the median. Note the differences on the vertical axes, and that on some of the figures the CG-MC and CG-QMC results are practically indistinguishable.

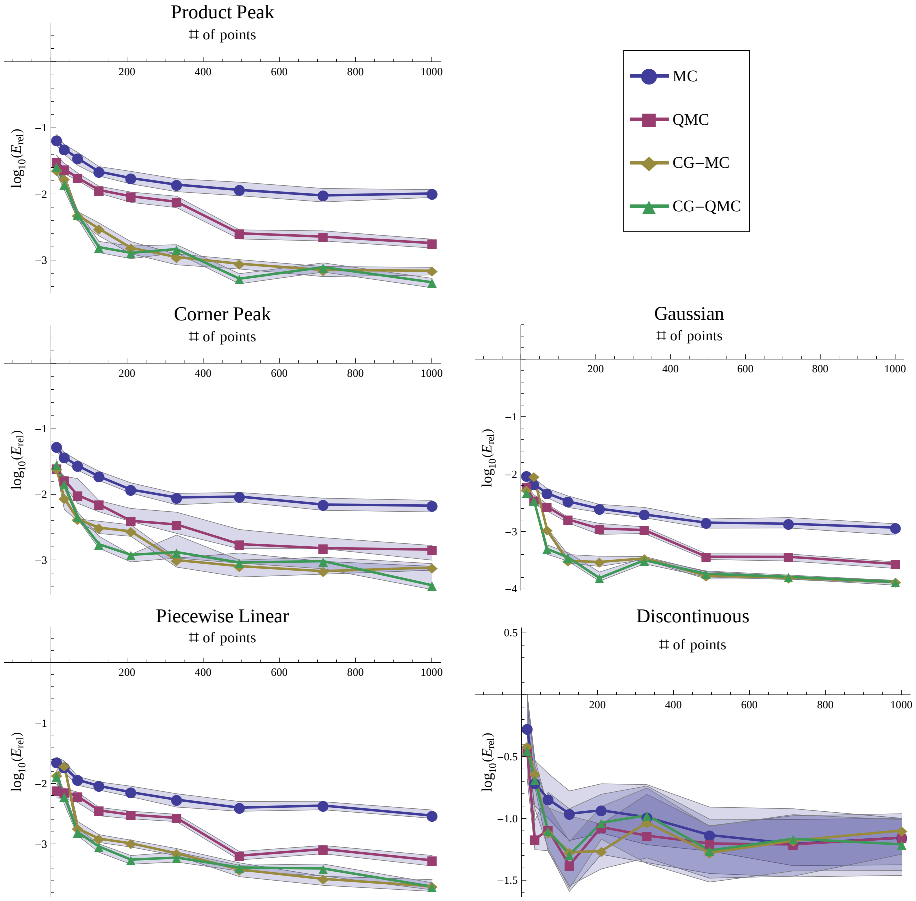


Figure 3: Performance of four cubature formulas using the four-dimensional parametric families of functions with normal density. Horizontal axis: number of points used in the formulas; the points correspond to increasing degree of exactness of the CG-MC and CG-QMC formulas from 2 to 10. CG-MC implemented the column generation algorithm where the pricing problem in semi-infinite linear programming was solved using a Monte Carlo algorithm. CG-QMC implemented the column generation algorithm where the pricing problem in semi-infinite linear programming was solved using a Quasi-Monte Carlo (Sobol) algorithm. Vertical axis: base-10 logarithm of the median relative errors from 200 experiments. The gray shaded bands around the median relative errors are 0.95-level confidence intervals on the median.

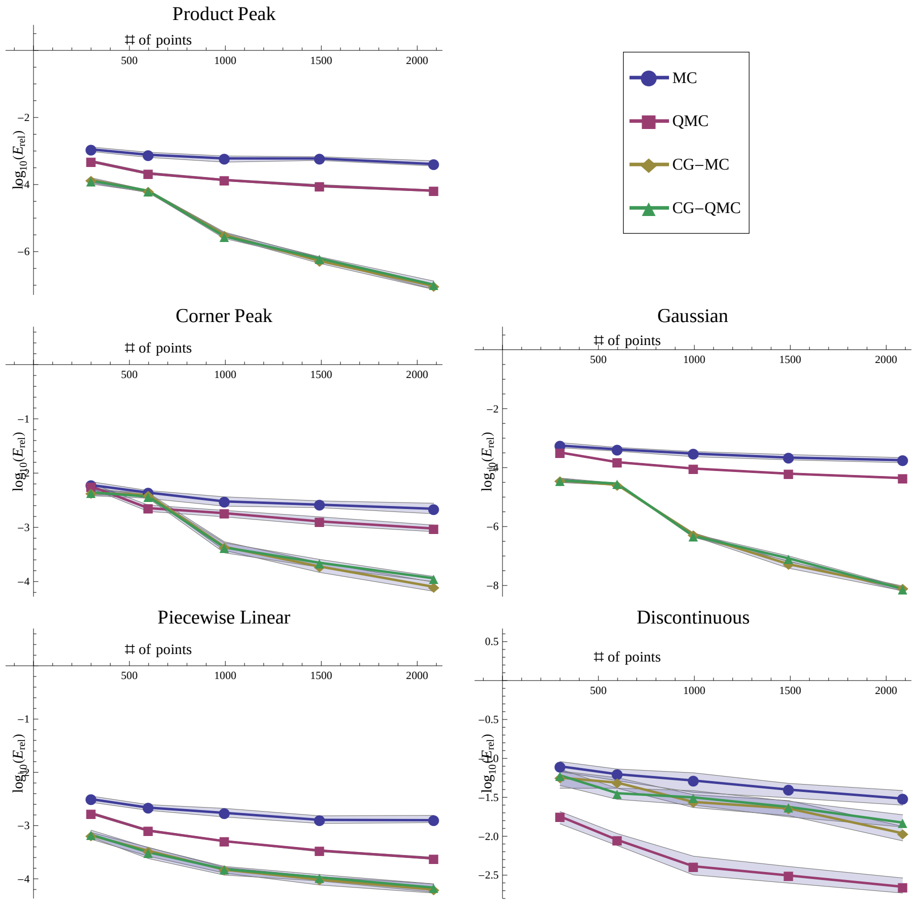
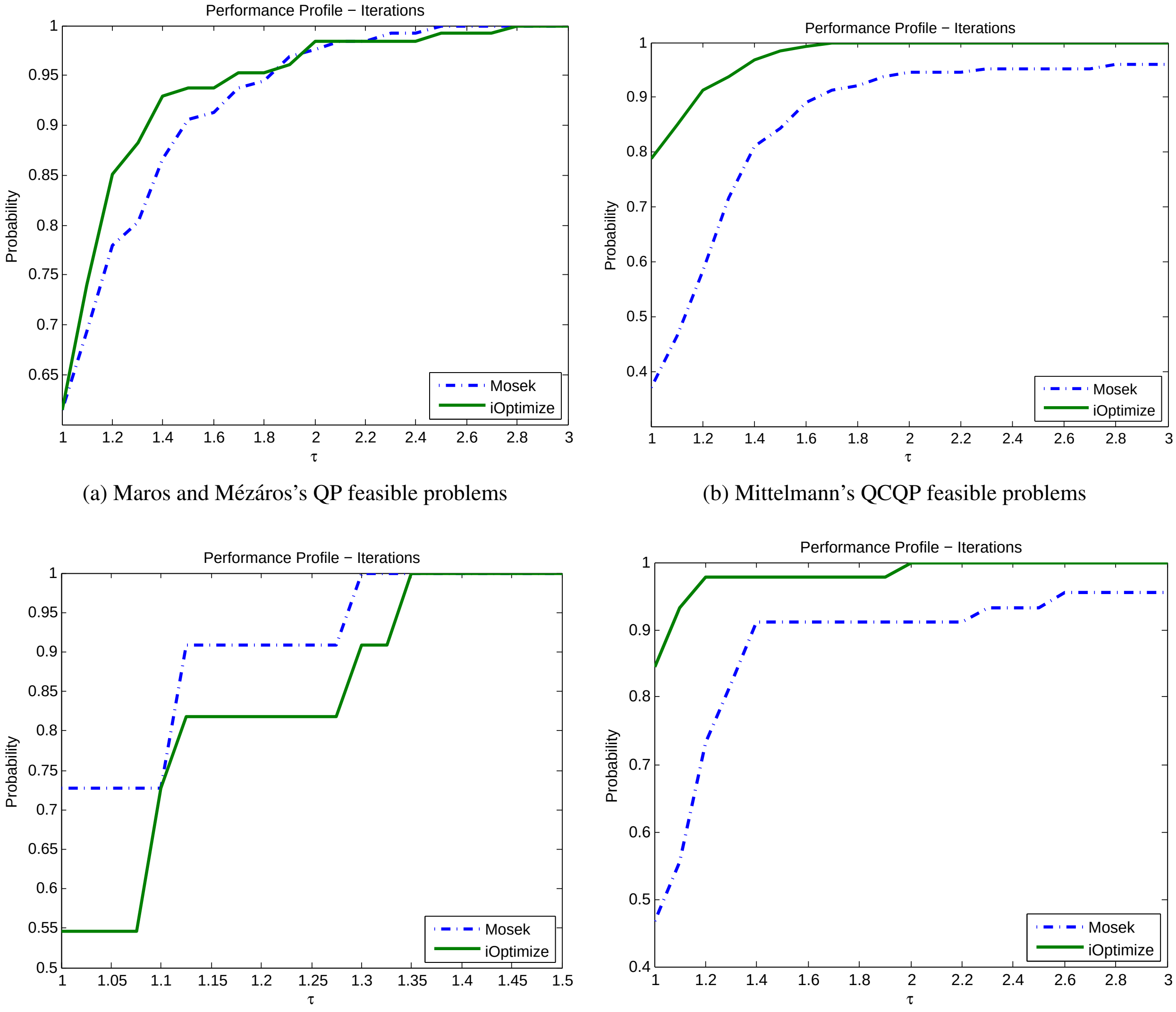


Figure 4: Performance of four cubature formulas using the 100-dimensional parametric families g_1–g_5. Horizontal axis: number of points used in the formulas; the points correspond to increasing degree of exactness of the CG-MC and CG-QMC formulas from 2 to 6. Vertical axis: base-10 logarithm of the median relative errors from 200 experiments. The gray shaded bands around the median relative errors are 0.95-level confidence intervals on the median.

d	K	MC	QMC	CG-MC	CG-QMC
2	28	0.1994	0.1817	0.0683	0.0102
3	84	0.1139	0.1130	0.0037	0.0726
4	210	0.0661	0.0626	0.0057	0.0015
5	462	0.0457	0.0319	0.0010	0.0019
6	924	0.0299	0.0189	0.0070	0.0028
7	1716	0.0245	0.0078	0.0044	0.0037

Table 4: Relative errors of the approximate solutions to the Utility maximization model, as a function of the degree of exactness d. The number of scenarios $K = \binom{d+6}{6}$ is also shown.

A.3 Solution of Monotone Complementarity and General Convex Programming Problems Using a Modified Potential Reduction Interior Point Method



(a) Maros and Mézáros's QP feasible problems

(b) Mittelmann's QCQP feasible problems

(c) Vanderbei's QCQP feasible problems

(d) General CP feasible problems

Figure 5: Performance profiles of our implementation (`iOptimize`) and `MOSEK` on feasible problems

Problem	MOSEK	our implementation	Problem	MOSEK	our implementation	Problem	MOSEK	our implementation
aug2d	6	12	laser	12	10	qpcboei1	20	22
aug2dc	6	12	liswet1	26	19	qpcboei2	21	26
aug2dcqp	13	13	liswet2	32	17	qpcstair	22	25
aug2dqp	15	14	liswet3	21	16	qpilotno	35§	42
aug3d	6	8	liswet4	26§	18	qrecipe	16	19
aug3dc	6	8	liswet5	25	16	qptest	6	6
aug3dcqp	11	11	liswet6	24	18	qsc205	16	17
aug3dqp	12	13	liswet7	31	19	qscagr25	20	22
cont-050	11	11	liswet8	38	37	qscagr7	19	22
cont-100	12	11	liswet9	37	31	qscfxm1	25	25
cont-101	13	9	liswet10	36	21	qscfxm2	30	33
cont-200	12	11	liswet11	31	22	qscfxm3	30	35
cont-201	15	10	liswet12	30	32	qscorpio	13	14
cont-300	16	13	lotschd	10	9	qscrs8	21	24
cvxqp1_m	10	10	mosarqp1	9	9	qscsd1	10	9
cvxqp1_s	8	8	mosarqp2	9	9	qscsd6	13	14
cvxqp2_m	10	10	powell20	10§	8	qscsd8	12	11
cvxqp2_s	9	10	primal1	10	9	qsctap1	18	17
cvxqp3_m	12	12	primal2	8	7	qsctap2	13	13
cvxqp3_s	10	10	primal3	9	9	qsctap3	14	15
dpklo1	5	3	primal4	10	9	qseba	16	26
dtoc3	5	4	primalc1	18	18	qshare1b	23	29
dual1	14	12	primalc2	15	25	qshare2b	17	17
dual2	12	9	primalc5	8	10	qshell	33	38
dual3	12	9	primalc8	10	12	qship04l	14	14
dual4	12	10	q25fv47	28	27	qship04s	13	14
dualc1	10	25	qadlittle	11	11	qship08l	13	14
dualc2	10	20	qafiro	12	14	qship08s	13	14
dualc5	7	13	qbandm	19	20	qship12l	17	16
dualc8	7	19	qbeaconf	14	14	qship12s	17	17
genhs28	4	3	qbore3d	13	19	qsierra	18	20
gouldqp2	14	13	qbrandy	14	17	qstair	22	23
gouldqp3	10	11	qcapri	26	36	qstandat	15	13
hs118	10	10	qe226	14	16	s268	22	12
hs21	9	9	qetamacr	27	32	stadat1	25	12
hs268	22	12	qfffff80	22	25	stadat2	41	18
hs35	5	5	qforplan	23	31	stadat3	46	19
hs35mod	11	11	qgfrdxpn	23	30	tame	3	3
hs51	5	3	qgrow15	22	22	ubh1	6	3
hs52	4	3	qgrow22	25	26	yao	23	23
hs53	6	4	qgrow7	23	21	zecevic2	5	5
hs76	5	5	qisrael	21	28			
ksip	21	16	qpcblend	19	17	Avg	16.22	15.92

[1] §: Near optimal solution found.

Table 5: Comparison with MOSEK on Maros and Meszaros's QP test problems

Problem	MOSEK	Our Implementation	Problem	MOSEK	Our Implementation	Problem	MOSEK	Our Implementation
QQ-aug2d	11	8	QQ-laser	**135**	19	QQ-qpcboei1	24	24
QQ-aug2dc	9	8	QQ-liswet1	41?	21	QQ-qpcboei2	29	29
QQ-aug2dcqp	18	19	QQ-liswet2	21	17	QQ-qpcstair	24	21
QQ-aug2dqp	†	21	QQ-liswet3	17	17	QQ-qpilotno	39	30
QQ-aug3d	6	5	QQ-liswet4	17	18	QQ-qrecipe	16	18
QQ-aug3dc	7	5	QQ-liswet5	15	20	QQ-qptest	7	7
QQ-aug3dcqp	17	11	QQ-liswet6	17	19	QQ-qsc205	17	17
QQ-aug3dqp	22	12	QQ-liswet7	34?	22	QQ-qscagr25	20	29
QQ-cont-050	14	11	QQ-liswet8	49?	30	QQ-qscagr7	18	19
QQ-cont-100	15	12	QQ-liswet9	26?	36	QQ-qscfxm1	43	30
QQ-cont-101	12	11	QQ-liswet10	18?	29	QQ-qscfxm2	39	36
QQ-cont-200	19	13	QQ-liswet11	27?	23	QQ-qscfxm3	26	37
QQ-cont-201	14	12	QQ-liswet12	24?	38	QQ-qscorpio	17	16
QQ-cont-300	15	12	QQ-lotschd	15	9	QQ-qscrs8	24	24
QQ-cvxqp1_m	15	10	QQ-mosarqp1	15	10	QQ-qscsd1	13	11
QQ-cvxqp1_s	15	8	QQ-mosarqp2	14	9	QQ-qscsd6	17	15
QQ-cvxqp2_m	18	10	QQ-powell20	†	19	QQ-qscsd8	16	12
QQ-cvxqp2_s	12	9	QQ-primal1	11	10	QQ-qsctap1	19	20
QQ-cvxqp3_m	18	13	QQ-primal2	9	9	QQ-qsctap2	18	13
QQ-cvxqp3_s	14	9	QQ-primal3	10	11	QQ-qsctap3	18	14
QQ-dpklo1	6	5	QQ-primal4	11	9	QQ-qseba	**62**	28
QQ-dtoc3	11	7	QQ-primalc1	20	19	QQ-qshare1b	27	26
QQ-dual1	14	13	QQ-primalc2	18	17	QQ-qshare2b	19	19
QQ-dual2	13	12	QQ-primalc5	11	10	QQ-qshell	26	30
QQ-dual3	14	12	QQ-primalc8	15	12	QQ-qship04l	21	15
QQ-dual4	13	10	QQ-q25fv47	39	28	QQ-qship04s	18	14
QQ-dualc1	17	21	QQ-qadlittle	19	12	QQ-qship08l	20	15
QQ-dualc2	14	17	QQ-qafiro	16	14	QQ-qship08s	21	15
QQ-dualc5	11	9	QQ-qbandm	21	21	QQ-qship12l	24	19
QQ-dualc8	18	13	QQ-qbeaconf	12	13	QQ-qship12s	24	19
QQ-genhs28	5	5	QQ-qbore3d	14	19	QQ-qsierra	24	20
QQ-gouldqp2	13	15	QQ-qbrandy	17	17	QQ-qstair	31	24
QQ-gouldqp3	13	10	QQ-qcapri	26	32	QQ-qstandat	16	17
QQ-hs118	11	11	QQ-qe226	17	17	QQ-s268	13	15
QQ-hs21	13	10	QQ-qetamacr	24	33	QQ-stadat1	53	19
QQ-hs268	13	15	QQ-qfffff80	30	26	QQ-stadat2	18	18
QQ-hs35	6	6	QQ-qforplan	**180**§	23	QQ-stadat3	18	18
QQ-hs35mod	13	12	QQ-qgfrdxpn	25	24	QQ-tame	5	6
QQ-hs51	5	5	QQ-qgrow15	21	21	QQ-ubh1	20	5
QQ-hs52	6	5	QQ-qgrow22	24	24	QQ-yao	27§	23
QQ-hs53	10	6	QQ-qgrow7	21	21	QQ-zecevic2	6	7
QQ-hs76	8	6	QQ-qisrael	28	29	Avg1	20.91	16.43
QQ-ksip	15	13	QQ-qpcblend	20	16	Avg2	18.34	16.26

[1] †: Max number of iterations reached.
[2] §: Near optimal solution found.
[3] ?: Terminated with unknown status.

Table 6: Comparison with MOSEK on Mittelmann's QCQP test problems

Problem	MOSEK	Our Implementation
airport	26	28
polak4	10	13
makela1	8	8
makela2	7	7
makela3	6	8
gigomez1	8	9
hanging	13	10
madsschj	10	11
rosenmmx	9	8
optreward	10	10
optprloc	18	16
Avg	11.36	11.64

Table 7: Comparison with MOSEK on Cute QCQP test problems

Problem	MOSEK	Our Implementation	Ipopt	Knitro
bigbank	20	21	24	19
gridnete	9	7	5	4
gridnetf	16	13	27	17
gridneth	8	6	6	3
gridneti	10	8	7	6
dallasl	**74**	33	**249**	**245**
dallasm	**128**	31	‡	‡
dallass	†	31	†	**135**
polak1	6	5	4	2
polak2	6	5	4	1
polak3	10	8	‡	‡
polak5	6	5	24	12
smbank	14	27	12	11
cantilvr	12	12	5	3
cb2	8	8	4	3
cb3	7	7	4	4
chaconn1	8	8	4	3
chaconn2	7	7	4	4
dipigri	14	10	6	2
gpp	12	10	22	11
hong	13	15	11	7
loadbal	22	21	13	7
svanberg	14	14	30	16
antenna\antenna2	21§	16	**178**	**58**
antenna\antenna_vareps	19	18	**106**	**61**
braess\trafequil	15	16	18	22
braess\trafequil2	11	13	26	13
braess\trafequilsf	14	15	18	24
braess\trafequil2sf	12	12	25	12
elena\chemeq	23	20	37	16
elena\s383	**85**	21	18	8
firfilter\fir_convex	33§	13	27	11
markowitz\growthopt	8	7	8	7
wbv\antenna2	15	13	31	16
wbv\lowpass2	30§	22	31	**66**
batch	13	12	13	16
stockcycle	11	10	23	12
synthes1	6	6	6	6
synthes2	8	8	10	7
synthes3	8	8	9	8
trimloss2	10	10	13	7
trimloss4	11	12	16	8
trimloss5	13	13	20	10
trimloss6	12	12	23	11
trimloss7	13	13	35	14
trimloss12	19	17	41	22
Avg1	18.46	14.00	–	–
Avg2	13.07	12.69	–	–

[1] †: Max number of iterations reached.

[2] §: Near optimal solution found.

[3] ‡: Function evaluate error from AMPL.

Table 8: Comparison with MOSEK, Ipopt, and Knitro on CP test problems

Problem	MOSEK	Our Implementation	Problem	MOSEK	Our Implementation	Problem	MOSEK	Our Implementation
iQQ-aug2d	†	16	iQQ-laser	29	13	iQQ-qpcboei1	†	12
iQQ-aug2dc	†	16	iQQ-liswet1	31	42	iQQ-qpcboei2	†	17
iQQ-aug2dcqp	†	29	iQQ-liswet2	30	43	iQQ-qpcstair	†	20
iQQ-aug2dqp	24	29	iQQ-liswet3	30	38	iQQ-qpilotno	†	16
iQQ-aug3d	21	35	iQQ-liswet4	30	40	iQQ-qrecipe	12	20
iQQ-aug3dc	16	16	iQQ-liswet5	21	47	iQQ-qptest	15	16
iQQ-aug3dcqp	**121**	33	iQQ-liswet6	30	43	iQQ-qsc205	16	41
iQQ-aug3dqp	**95**	37	iQQ-liswet7	24	40	iQQ-qscagr25	383?	14
iQQ-cont-050	†	9	iQQ-liswet8	26	33	iQQ-qscagr7	38	14
iQQ-cont-100	†	14	iQQ-liswet9	24	37	iQQ-qscfxm1	†	15
iQQ-cont-101	†	9	iQQ-liswet10	30	42	iQQ-qscfxm2	†	16
iQQ-cont-200	†	14	iQQ-liswet11	30	42	iQQ-qscfxm3	†	16
iQQ-cont-201	**70**	9	iQQ-liswet12	24	38	iQQ-qscorpio	31	25
iQQ-cont-300	†	9	iQQ-lotschd	26	12	iQQ-qscrs8	52	22
iQQ-cvxqp1_m	†	26	iQQ-mosarqp1	**100**	33	iQQ-qscsd1	28	20
iQQ-cvxqp1_s	21	35	iQQ-mosarqp2	**66**	38	iQQ-qscsd6	25	22
iQQ-cvxqp2_m	**97**	29	iQQ-powell20	†	16	iQQ-qscsd8	30	37
iQQ-cvxqp2_s	18	37	iQQ-primal1	31	43	iQQ-qsctap1	38	36
iQQ-cvxqp3_m	†	30	iQQ-primal2	32	37	iQQ-qsctap2	†	40
iQQ-cvxqp3_s	22	35	iQQ-primal3	33	41	iQQ-qsctap3	**87**	32
iQQ-dpklo1	†	13	iQQ-primal4	34	36	iQQ-qseba	28	20
iQQ-dtoc3	24	15	iQQ-primalc1	36	42	iQQ-qshare1b	51	17
iQQ-dual1	10	31	iQQ-primalc2	19	36	iQQ-qshare2b	**175**	13
iQQ-dual2	9	16	iQQ-primalc5	22	23	iQQ-qshell	20	17
iQQ-dual3	9	18	iQQ-primalc8	24	32	iQQ-qship04l	15	11
iQQ-dual4	9	33	iQQ-q25fv47	†	15	iQQ-qship04s	15	11
iQQ-dualc1	18	35	iQQ-qadlittle	†	14	iQQ-qship08l	†	16
iQQ-dualc2	15	35	iQQ-qafiro	**70**	39	iQQ-qship08s	†	17
iQQ-dualc5	10	44	iQQ-qbandm	37	14	iQQ-qship12l	†	17
iQQ-dualc8	30	34	iQQ-qbeaconf	26	14	iQQ-qship12s	†	18
iQQ-genhs28	†	13	iQQ-qbore3d	10	18	iQQ-qsierra	†	17
iQQ-gouldqp2	10	10	iQQ-qbrandy	†	16	iQQ-qstair	353?	17
iQQ-gouldqp3	14	16	iQQ-qcapri	†	22	iQQ-qstandat	**340**	39
iQQ-hs118	**232**	11	iQQ-qe226	**281**	36	iQQ-s268	26	45
iQQ-hs21	40	35	iQQ-qetamacr	†	34	iQQ-stadat1	16	26
iQQ-hs268	26	45	iQQ-qfffff80	†	20	iQQ-stadat2	†	35
iQQ-hs35	**72**	16	iQQ-qforplan	21	11	iQQ-stadat3	†	25
iQQ-hs35mod	25	16	iQQ-qgfrdxpn	21	19	iQQ-tame	6	10
iQQ-hs51	29	27	iQQ-qgrow15	22	16	iQQ-ubh1	†	26
iQQ-hs52	22	26	iQQ-qgrow22	25	18	iQQ-yao	27	38
iQQ-hs53	27	39	iQQ-qgrow7	22	18	iQQ-zecevic2	8	9
iQQ-hs76	**69**	13	iQQ-qisrael	†	19	Avg1	42.20	28.15
iQQ-ksip	30	42	iQQ-qpcblend	**105**	38	Acg2	27.39	28.15

[1] †: Max number of iterations reached.
[2] ?: Terminated with unknown status.

Table 9: Comparison with MOSEK on Maros and Meszaros's QCQP infeasible test problems

Problem	MOSEK	Our Implementation
iairport	10	19
ipolak4	†	25
imakela1	28	32
imakela2	29	32
imakela3	8	14
igigomez1	26	26
ihanging	32	30
imadsschj	24	17
irosenmmx	27	35
ioptreward	11	20
ioptprloc	8	21
Avg	20.30	24.60

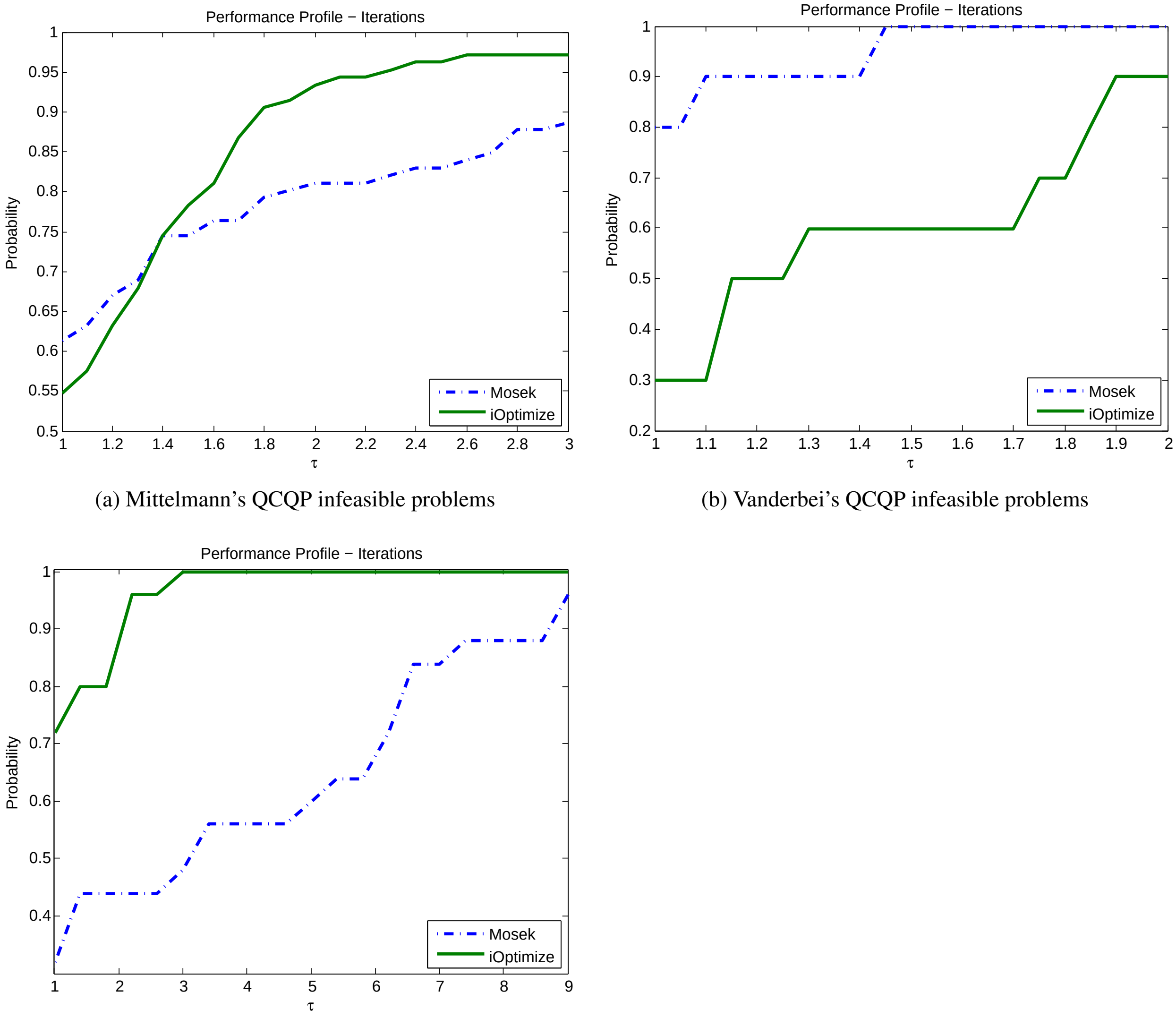
[1] †: Max number of iterations reached.

Table 10: Comparison with MOSEK on Cute QCQP infeasible test problems

Problem	MOSEK	Our Implementation	Ipopt	Knitro
ibigbank	**196**	27	48	42
igridnete	**134**	14	21	†
igridnetf	†	36	42	†
igridneth	**202**	23	25	18
igridneti	†	40	40	**133**§
idallasl	‡	‡	50§	‡
idallasm	‡	47	‡	0
idallass	‡	45	‡	0
ipolak1	50	16	‡	‡
ipolak2	7	13	‡	‡
ipolak3	**178**	27	◇	†
ipolak5	7	13	37	†
ismbank	**165**	31	34	37
icantilvr	0	12	25	0
icb2	**196**	32	◇	18
icb3	**196**	31	†	8
ichaconn1	**196**	32	◇	6
ichaconn2	**196**	31	10	17
idipigri	**157**	32	**188**	†
igpp	†	38	**65**	†
ihong	11	22	16	†
iloadbal	**123**	14	22	†
isvanberg	†	27	48	53
antenna\iantenna2	†	**186**	†	**156**
antenna\iantenna_vareps	†	47	†	†
braess\itrafequil	43	38	36	**126**
braess\itrafequil2	7	9	**82**	50
braess\itrafequilsf	47	46	34	**103**
braess\itrafequil2sf	†	9	54	**91**
elena\ichemeq	14	14	**102**	16
elena\is383	7	14	**67**	†
firfilter\ifir_convex	0	13	**99**	0
markowitz\igrowthopt	15	16	22	†
wbv\iantenna2	0	17	32	0
wbv\ilowpass2	0	13	32	0
ibatch	0	22	32	**158**
istockcycle	**117**	35	**72**	†
isynthes1	9	25	35	35
isynthes2	**62**	23	48	46
isynthes3	53	40	**94**	†
itrimloss2	0	15	52	0
itrimloss4	0	16	40	0
itrimloss5	0	15	**60**	0
itrimloss6	0	15	55	0
itrimloss7	0	16	**61**	0
itrimloss12	0	16	**73**	0
Avg1	66.33	19.44	–	–
Avg2	11.74	16.04	–	–

[1] †: Max number of iterations reached.
[2] ‡: Function evaluate error from AMPL.
[3] §: Solver terminates with a near optimal status.
[4] ◇: Ipopt fails in restoration phase.

Table 11: Comparison with MOSEK, Ipopt and Knitro on CP infeasible test problems



(a) Mittelmann's QCQP infeasible problems

(b) Vanderbei's QCQP infeasible problems

(c) General CP infeasible problems

Figure 6: Performance profiles of our implementation (`iOptimize`) and `MOSEK` on infeasible problems

A.4 Modeling Transmission Line Constraints in Two-stage Robust Unit Commitment Problem

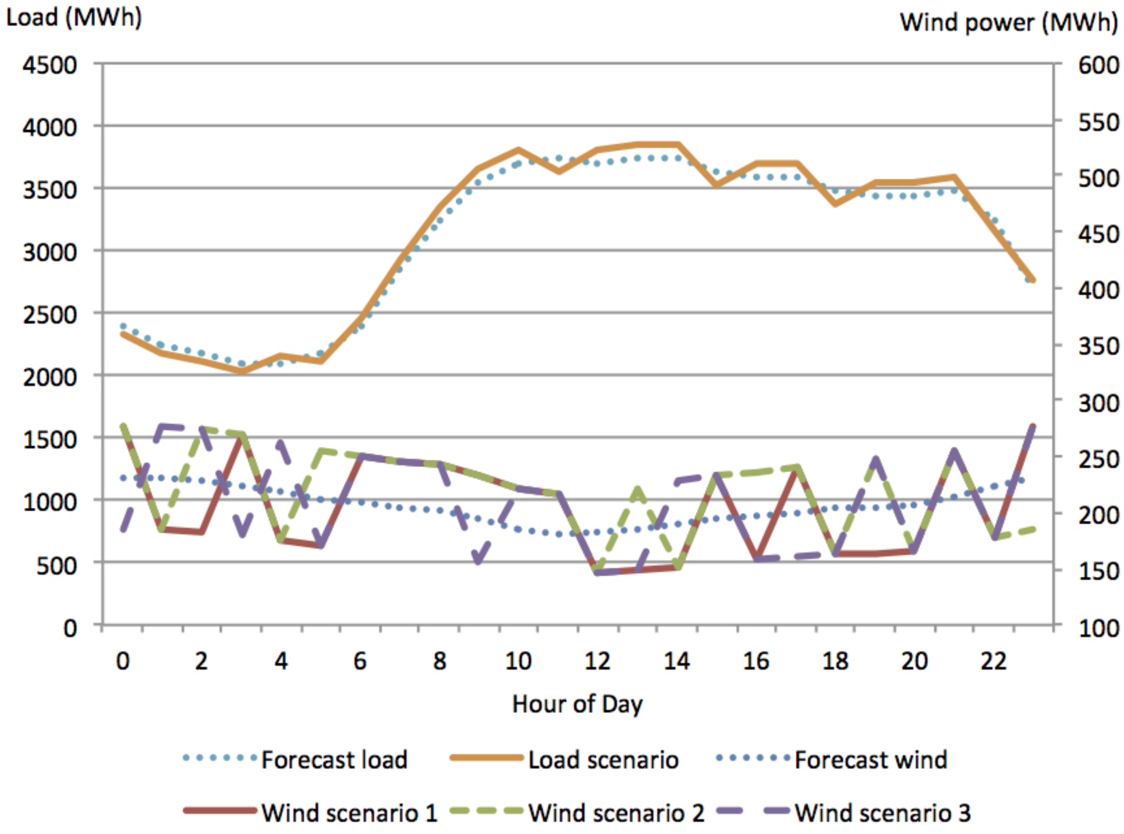


Figure 7: Example of the worst-case scenario.

A.5 Robust distribution network reconfiguration

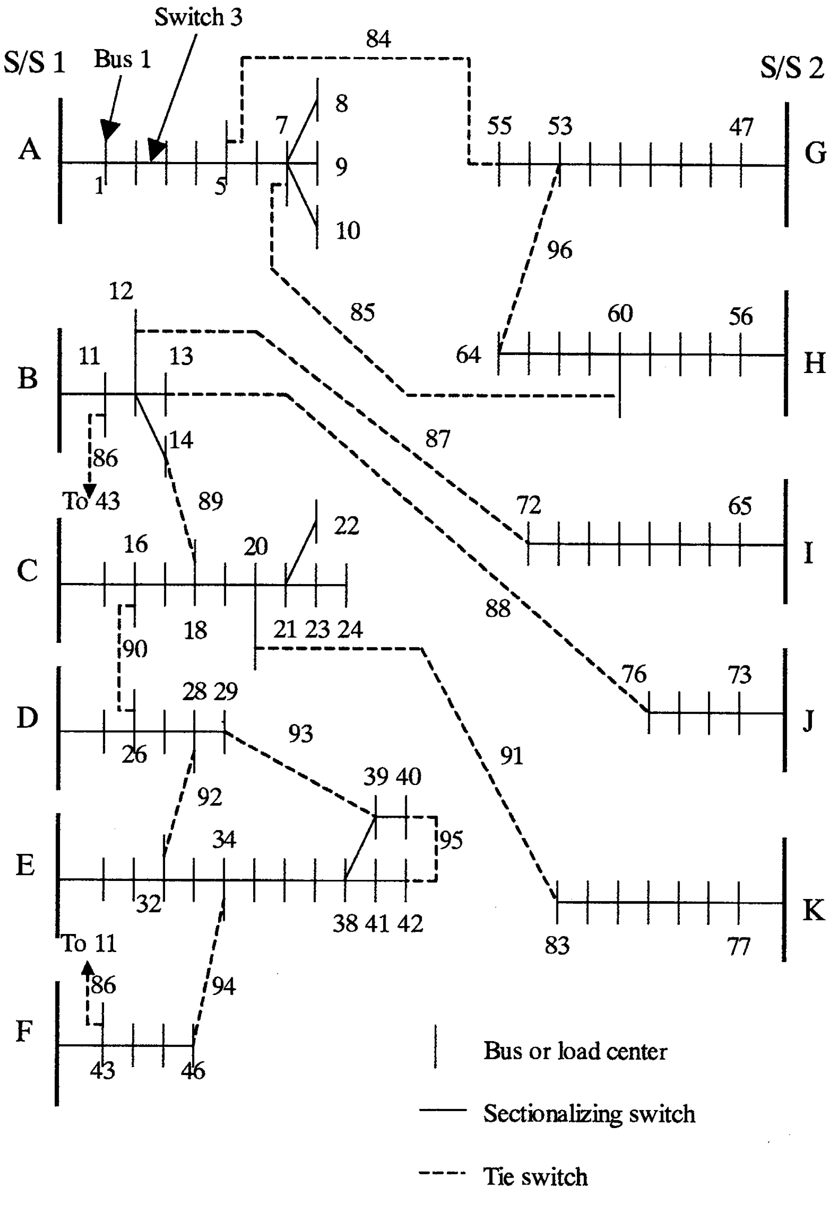


Figure 8: 94-bus distribution network [104]

Table 12: Summary of test cases

	16-bus	33-bus	70-bus	94-bus
No. of substations	3	1	2	11
No. of lines	16	37	79	96
No. of lines to disconnect	3	5	11	13

Table 13: Comparison of deterministic and robust models

		Deterministic	Robust model
16-bus	Configuration	7, 8, 16	7, 8, 16
	Nominal power losses (kW)	466	466
	Maximum power losses (kW)	517.8	517.8
	CPU Time (s)	0.8	17.8
	Number of iteration	-	1
33-bus	Configuration	7, 9, 14, 29, 32	7, 11, 14, 29, 32
	Nominal power losses (kW)	129.9	131.6
	Maximum power losses (kW)	infeasible	145.3
	CPU Time (s)	3.9	209.3
	Number of iteration	-	2
70-bus	Configuration	14, 30, 39, 46, 51, 66, 71, 75, 76, 77, 79	14, 30, 39, 46, 51, 66, 70, 71, 76, 77, 78
	Nominal power losses (kW)	204.1	207.7
	Maximum power losses (kW)	infeasible	224.5
	CPU Time (s)	13.3	123.1
	Number of iteration	-	2
94-bus	Configuration	7, 13, 33, 37, 40, 63, 72, 82, 84, 86, 89, 90 92	7, 13, 34, 39, 42, 61, 72, 82, 84, 86, 89, 90 92
	Nominal power losses (kW)	471.9	472.7
	Maximum power losses (kW)	infeasible	521.1
	CPU Time (s)	3	160.1
	Number of iteration	-	2

Table 14: Comparison of robust models with different variation limits of the uncertainty sets with $B = 8$

Variation limits	1%	2%	3%	4%	5%	6%
Maximum power loss (kW)	480	490	502	507	523	534
Number of iteration	1	1	1	1	2	2

Table 15: Comparison of robust models with different budget of uncertainty set parameter with ± 5% limit

Budget parameter B	2	4	6	8
Maximum power loss (kW)	481	493	510	523
Number of iteration	2	2	2	2

References

[1] A (PO)rtable (S)tochastic programming (T)est (S)et (POSTS) (`http://users.iems.northwestern.edu/ jrbirge/html/dholmes/post.html`).

[2] `AMPL`: A modeling language for mathematical programming. `www.ampl.com`.

[3] `AMPL` nonlinear test problems. `http://www.orfe.princeton.edu/˜rvdb/ampl/nlmodels/`.

[4] `Bonmin` (Basic Open-source Nonlinear Mixed INteger programming) `https://projects.coin-or.org/bonmin`.

[5] `IBM Cplex` optimizer (`http://www.ibm.com/`).

[6] `Knitro` (`http://www.ziena.com/knitro.htm`).

[7] `Lindo` (`http://www.lindo.com/`).

[8] Mittelmann's convex QCQP test problems `http://plato.asu.edu/ftp/ampl_files/qpdata_ampl/`.

[9] Mixed integer nonlinear programming (MINLP) test problems. `http://wiki.mcs.anl.gov/leyffer/index.php/MacMINLP`.

[10] `OptiRisk-FortSp` (`http://www.optirisk-systems.com/`).

[11] SIPLIB: A stochastic integer programming test library (`http://www2.isye.gatech.edu/ sahmed/siplib/`).

[12] Sampling test (`http://pages.cs.wisc.edu/ swright/stochastic/sampling/`).

[13] Kumar Abhishek, Sven Leyffer, and Jeff Linderoth. FilMINT: An Outer Approximation-Based Solver for Convex Mixed-Integer Nonlinear Programs. *INFORMS Journal on Computing*, 22(4):555–567, March 2010.

[14] S. Ahmed and N. V. Sahinidis. Robust process planning under uncertainty. *Industrial & Engineering Chemistry Research*, 37(5):1883–1892, 1998.

[15] F. Alizadeh. *Combinatorial optimization with interior point methods and semidefinite matrices*. PhD thesis, University of Minnesota, 1991.

[16] F. Alizadeh and D. Goldfarb. Second-order cone programming. *Mathematical Programming*, 95(1):3–51, 2003.

[17] F. Alizadeh and D. Goldfarb. Second-order cone programming. *Mathematical Programming*, 95(1):3–51, 2003.

[18] Baha Alzalg. Decomposition-based interior point methods for stochastic quadratic second-order cone programming. *Applied Mathematics and Computation*, 249:1–18, 2014.

[19] E.D. Andersen and Y. Ye. On a homogeneous algorithm for the monotone complementarity problem. *Mathematical Programming*, 84:375–399, 1999.

[20] K. A. Ariyawansa and Andrew J. Felt. On a new collection of stochastic linear programming test problems. *INFORMS Journal on computing*, 16(3):291–299, 2004.

[21] Alper Atamtrk and Vishnu Narayanan. Conic mixed-integer rounding cuts. *Mathematical Programming*, 122 (1):1–20, August 2008.

[22] T. Bagby, L. Bos, and N. Levenberg. Multivariate simulatenous approximation. *Constructive Approximation*, 18:569–577, 2002. doi: 10.1007/s00365-001-0024-6.

[23] E. Balas, S. Ceria, and G. Cornuéjols. A lift-and-project cutting plane algorithm for mixed 0-1 programs. *Mathematical Programming*, 58:295–324, 1993.

[24] M. Baran and F. Wu. Network reconfiguration in distribution systems for loss reduction and load balancing. *IEEE Transactions on Power Delivery*, 4(2):1401–1407, 1989.

[25] A. Ben-Tal and A. Nemirovski. Robust optimization - methodology and applications. *Mathematical Programming*, 92(3):453–480, 2002.

[26] A. Ben-Tal and A. Nemirovski. Selected topics in robust convex optimization. *Mathematical Programming*, 112(1):125–158, 2008.

[27] A. Ben-Tal, B. Golany, A. Nemirovski, and J.-Ph. Vial. Supplier-retailer flexible commitments contracts: A robust optimization approach. *Manufacturing & Service Operations Management*, 2003.

[28] A. Ben-Tal, S. Boyd, and A. Nemirovski. Extending scope of robust optimization: Comprehensive robust counterparts of uncertain problems. *Mathematical Programming*, 107(1-2):63–89, 2006.

[29] D. Bertsimas and D. Pachamanova. Robust multiperiod portfolio management in the presence of transaction costs. *Computers & Operations Research*, 35(1):3–17, 2008.

[30] D. Bertsimas and M. Sim. Tractable approximations to robust conic optimization problems. *Mathematical Programming*, 107(1-2):5–36, 2006.

[31] D. Bertsimas and A. Thiele. A robust optimization approach to inventory theory. *Operations Research*, 54(1): 150–168, 2006.

[32] D. Bertsimas, O. Nohadani, and K. M. Teo. Robust optimization in electromagnetic scattering problems. *Journal of Applied Physics*, 101(7):–, 2007.

[33] D. Bertsimas, E. Litvinov, X. A. Sun, J. Zhao, and T. Zheng. Adaptive robust optimization for the security constrained unit commitment problem. *IEEE Transactions on Power Systems*, 28(1):52–63, 2013.

[34] Dimitris Bertsimas and Melvyn Sim. Robust conic optimization. *Mathematical Programming*, 2004.

[35] Bruno Betrò. An accelerated central cutting plane algorithm for linear semi-infinite programming. *Mathematical Programming*, 101:479–495, 2004. doi: 10.1007/s10107-003-0492-5.

[36] John Birge and Francois Louveaux. *Introduction to Stochastic Programming*. Springer-Verlag, 1997.

[37] Merve Bodur and James Luedtke. Mixed-integer rounding enhanced benders decomposition for multiclass service system staffing and scheduling with arrival rate uncertainty. `http://www.optimization-online.org/DB_HTML/2013/10/4080.html`, 2013.

[38] Merve Bodur, Sanjeeb Dash, Oktay Günlük, and James Luedtke. Strengthened benders cuts for stochastic integer programs with continuous recourse. `www.optimization-online.org/DB_FILE/2014/03/4263.pdf`, 2014.

[39] Pierre Bonami. Lift-and-Project Cuts for Mixed Integer Convex Programs. In Oktay Gnlk and Gerhard J. Woeginger, editors, *Integer Programming and Combinatoral Optimization*, number 6655 in Lecture Notes in Computer Science, pages 52–64. Springer Berlin Heidelberg, 2011.

[40] Pierre Bonami, Lorenz T. Biegler, Andrew R. Conn, Grard Cornujols, Ignacio E. Grossmann, Carl D. Laird, Jon Lee, Andrea Lodi, Franois Margot, Nicolas Sawaya, and Andreas Wchter. An algorithmic framework for convex mixed integer nonlinear programs. *Discrete Optimization*, 5(2):186–204, May 2008.

[41] Pierre Bonami, Mustafa Kilin, and Jeff Linderoth. Algorithms and Software for Convex Mixed Integer Nonlinear Programs. In Jon Lee and Sven Leyffer, editors, *Mixed Integer Nonlinear Programming*, number 154 in The IMA Volumes in Mathematics and its Applications, pages 1–39. Springer New York, 2012.

[42] O. Boni, A. Ben-Tal, and A. Nemirovski. Robust solutions to conic quadratic problems and their applications. *Optimization and Engineering*, 9(1):1–18, 2008.

[43] M. Chen and S. Mehrotra. Self-concordance and Decomposition-based Interior Point Methods for the Two-stage Stochastic Convex Optimization Problem. *SIAM Journal on Optimization*, 21(4):1667–1687, October 2011.

[44] Michael Chen, Sanjay Mehrotra, and Dávid Papp. Scenario generation for stochastic optimization problems via the sparse grid method. *Computational Optimization and Applications*, 2014 (to appear).

[45] X. Chen, M. Sim, P. Sun, and J. W. Zhang. A linear decision-based approximation approach to stochastic programming. *Operations Research*, 56(2):344–357, 2008.

[46] S. Civanlar, J. J. Grainger, H. Yin, and S. S. H. Lee. Distribution feeder reconfiguration for loss reduction. *IEEE Transactions on Power Delivery*, 3(3):1217–1223, 1988.

[47] G. Consigli, J. Dupačová, and S. Wallace. Generating scenarios for multistage stochastic programs. *Annals of Operations Research*, 100:25–53, 2000.

[48] E. M. Constantinescu, V. M. Zavala, M. Rocklin, S. Lee, and M. Anitescu. A computational framework for uncertainty quantification and sochastic optimization in unit commitment with wind power generation. *IEEE Transactions on Power Systems*, 26(1):431–441, 2011.

[49] D. Das. A fuzzy multiobjective approach for network reconfiguration of distribution systems. *IEEE Transactions on Power Delivery*, 21(1):202–209, 2006.

[50] P. Date, R. Mamon, and L. Jalen. A new moment matching algorithm for sampling from partially specified symmetric distributions. *Operations Research Letters*, 36(6):669–672, November 2008. doi: 10.1016/j.orl.2008.07.004.

[51] E. de Klerk, C. Roos, and T. Terlaky. Semi-defimote problems in truss topology optimization. Technical report, T.U. Delft, 1995.

[52] Erick Delage. *DISTRIBUTIONALLY ROBUST OPTIMIZATION IN CONTEXT OF DATA-DRIVEN PROBLEMS*. PhD thesis, Stanford University, 2009.

[53] Erick Delage and Yinyu Ye. Distributionally robust optimization under moment uncertainty with application to data-driven problems. *Operations Research*, 58:595–612, 2010. doi: 10.1287/opre.1090.0741.

[54] M. A. H. Dempster and R. T. Thompson. EVPI-based importance sampling solution procedure for multistage stochastic linear programming on parallel MIMD architectures. *Annals of Operations Research*, 90:161–184, 1999.

[55] Dupačová, Gröwe-Kuska, and Römisch. Scenario reduction in stochastic programming: An approach using probability metrics. *Mathematical Programming Series A*, 95:493–511, 2003.

[56] Marco A. Duran and Ignacio E. Grossmann. An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Mathematical Programming*, 36(3):307–339, October 1986.

[57] M. T. ezik and G. Iyengar. Cuts for mixed 0-1 conic programming. *Mathematical Programming*, 104(1): 179–202, April 2005.

[58] A. Frangioni and C. Gentile. Perspective cuts for a class of convex 01 mixed integer programs. *Mathematical Programming*, 106(2):225–236, July 2005.

[59] Dinakar Gade, Simge Küçükyavuz, and Suvrajeet Sen. Decomposition algorithms with parametric gomory cuts for two-stage stochastic integer programs. *Mathematical Programming*, 144:39–64, 2014.

[60] Thomas Gerstner and Michael Griebel. Numerical integration using sparse grid. *Numerical Algorithm*, 18: 209–232, 1998.

[61] Laurent EL Ghaoui, Maksim Oks, and Francois Oustry. Worst-case value-at-risk and robust portofolio optimization: A conic programming approach. *Operations Research*, 51(4), 2003.

[62] D. Goldfarb and G. Iyengar. Robust portfolio selection problems. *Mathematics of Operations Research*, 28(1): 1–38, 2003.

[63] P. R. Gribik. A central cutting plane algorithm for semi-infinite programming problems. In R. Hettich, editor, *Semi-infinite programming*, number 15 in Lecture Notes in Control and Information Systems. Springer Verlag, New York, NY, 1979.

[64] Ignacio E. Grossmann and Zdravko Kravanja. Mixed-Integer Nonlinear Programming: A Survey of Algorithms and Applications. In Lorenz T. Biegler, Thomas F. Coleman, Andrew R. Conn, and Fadil N. Santosa, editors, *Large-Scale Optimization with Applications*, number 93 in The IMA Volumes in Mathematics and its Applications, pages 73–100. Springer New York, 1997.

[65] Holger Heitsch and Werner Römisch. Scenario tree modeling for multistage stochastic programs. *Mathematical Programming*, 118(2):371–406, 2009. ISSN 0025-5610. doi: 10.1007/s10107-007-0197-2.

[66] Kjetil Høyland, Michal Kaut, and Stein W. Wallace. A heuristic for moment-matching scenario generation. *Computational Optimization and Applications*, 24(2):169–185, February 2003. doi: 10.1023/A:1021853807313.

[67] K.-L. Huang and S. Mehrotra. Solution of monotone complementarity and general convex programming problems using a modified potential reduction interioir point method. http://www.optimization-online.org/DB_HTML/2012/04/3431.html, 2014.

[68] K.-L. Huang and S. Mehrotra. An empirical evaluation of a walk-relax-round heuristic for mixed integer convex programs. *Computational Optimization And Applications*, (To Appear), 2014. doi: 10.1007/s10589-014-9693-5.

[69] Kuo-Ling Huang, Kibaek Kim, and Sanjay Mehrotra. On implementing a hybrid-cut method for solving two-stage stochastic optimization problems. *(in preparation)*, 2015.

[70] R. Jiang, J. Wang, M. Zhang, and Y. Guan. Two-stage minimax regret unit commitment considering wind power uncertainty. *IEEE Transactions on Power Systems*, 2012.

[71] Kibaek Kim and Sanjay Mehrotra. A two stage stochastic integer programming approach for integrated staffing and scheduling under demand uncertainty. http://www.optimization-online.org/DB_FILE/2014/01/4200.pdf, 2014.

[72] Fatma Kln-Karzan. On Minimal Valid Inequalities for Mixed Integer Conic Programs. *arXiv:1408.6922 [math]*, August 2014. URL http://arxiv.org/abs/1408.6922.

[73] H. Konno. A cutting plane algorithm for solving bilinear programs. *Mathematical Programming*, 11:14–27, 1976.

[74] K. O. Kortanek and Hoon No. A central cutting plane algorithm for convex semi-infinite programming problems. *SIAM Journal on Optimization*, 3(4):901–918, November 1993.

[75] Kartik Krishnan and John E. Mitchell. A unifying framework for several cutting plane methods for semidefinite programming. *Optimization Methods and Software*, 21(1):57–74, February 2006.

[76] Gilbert Laporte and François Louveaux. The integer l-shaped method for stochastic integer programs with complete recourse. *Operations Research Letters*, 13:133–142, 1993.

[77] Changhyeok Lee, Cong Liu, and Sanjay Mehrotra. Modeling transmission line constraints in two-stage robust unit commitment problem. *IEEE Transactions on Power Systems*, 29(3):1221–1231, 2013.

[78] Changhyeok Lee, Cong Liu, and Sanjay Mehrotra. Two-stage distributionally-robust optimization for unit commitment problem. *IEEE*, 2015.

[79] Changhyeok Lee, Cong Liu, Sanjay Mehrotra, and Zhaohong Bie. Robust distribution network reconfiguration. *IEEE Transactions on Smart Grid*, 6(2):836–842, 2015.

[80] Z. K. Li and M. G. Ierapetritou. Robust optimization for process scheduling under uncertainty. *Industrial & Engineering Chemistry Research*, 47(12):4148–4157, 2008.

[81] M. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret. Applications of second-order cone programming. *Linear Algebra and its Applications*, 284:193–228, 1998.

[82] Philip M. Lurie and Matthew S. Goldberg. An approximate method for sampling correlated random variables from partially-specified distributions. *Management Science*, 44(2):203–218, February 1998. doi: 10.1287/mnsc.44.2.203.

[83] I. Maros and C. Mézáros. A repository of convex quadratic programming problems. *Optimization Methods and Software*, 11&12:671–681, 1999.

[84] S. Mehrotra and K.-L. Huang. Computational experience with a modified potential reduction algorithm for linear programming. *Optimization Methods And Software*, 27:865– 891, 2012. doi: 10.1080/10556788.2011.634911.

[85] Sanjay Mehrotra and Zhifeng Li. Branching on hyperplane methods for mixed integer linear and convex programming using adjoint lattices. *Journal of Global Optimization*, 49(4):623–649, May 2010.

[86] Sanjay Mehrotra and M. Gokhan Ozevin. On the implementation of interior point decomposition algorithms for two-stage stochastic conic programs. *SIAM (to appear)*, 19(4):1846–1880, 2009.

[87] Sanjay Mehrotra and Dávid Papp. Generating moment matching scenarios using optimization techniques. *SIAM Journal on Optimizaton*, 23(2):963–999, 2013.

[88] Sanjay Mehrotra and Dávid Papp. A cutting surface algorithm for semi-infinite convex programming with an application to moment robust optimization. *SIAM J. on Optimization*, 24(4):1670–1697, 2014.

[89] Sanjay Mehrotra and He Zhang. Models and algorithms for distributionally robust least squares problems. *Mathematical Programming*, 148(1–2):123–141, 2014.

[90] Sina Modaresi, Mustafa R. Kln, and Juan Pablo Vielma. Intersection Cuts for Nonlinear Integer Programming: Convexification Techniques for Structured Sets. *arXiv:1302.2556 [math]*, February 2013. URL http://arxiv.org/abs/1302.2556.

[91] Sina Modaresi, Mustafa R. Kln, and Juan Pablo Vielma. Split cuts and extended formulations for Mixed Integer Conic Quadratic Programming. *Operations Research Letters*, 43(1):10–15, January 2015.

[92] Yurii Nesterov and Arkadii Nemirovskii. *Interior Point Polynomial Algorithms In Convex Programming*. SIAM, Philadelphia, 1994.

[93] Erich Novak and Klaus Ritter. High dimensional integration of smooth functions over cubes. *Numer. Math.*, 75:79–97, 1996.

[94] U. A. Ozturk, M. Mazumdar, and B. A. Norman. A solution to the stochastic unit commitment problem using chance constrained programming. *IEEE Transactions on Power Systems*, 19(3):1589–1598, 2004.

[95] N. P. Padhy. Unit commitment - a bibliographical survey. *IEEE Transactions on Power Systems*, 19(2):1196–1205, 2004.

[96] G.Ch. Pflug. Scenario tree generation for multiperiod financial optimization by optimal discretization. *Mathematical Programming*, 89:251–271, 2001.

[97] G. Li R. Jiang, M. Zhang and Y. Guan. Two-stage robust power grid optimization problem. *available on optimization-online*, 2010.

[98] S.T. Rachev. *Probability Metrics and the Stability of Stochastic Models*. Wiley, 1991.

[99] Jean-Philippe P. Richard and Mohit Tawarmalani. Lifting inequalities: a framework for generating strong cuts for nonlinear programs. *Mathematical Programming*, 121(1):61–104, May 2008.

[100] Hanif D. Sherali and Warren P. Adams. A Reformulation-Linearization Technique (RLT) for semi-infinite and convex programs under mixed 0-1 and general discrete restrictions. *Discrete Applied Mathematics*, 157: 1319–1333, March 2009.

[101] Joelle Skaf and Stephen Boyd. Design of affine controllers via convex optimization. Technical report, Stanford University, 2008.

[102] R. M. Van Slyke and Roger J. B. Wets. L-shaped linear programs with applications to optimal control and stochastic programming. *Siam Journal On Applied Mathematics*, 17(4):638–663, July 1969. ISSN 0036-1399.

[103] Robert A. Stubbs and Sanjay Mehrotra. A branch-and-cut method for 0-1 mixed convex programming. *Mathematical Programming*, 86(3):515–532, December 1999.

[104] C.-T. Su and C.-S. Lee. Network reconfiguration of distribution systems using improved mixed-integer hybrid differential evolution. *IEEE Transactions on Power Delivery*, 18(3):1022–1027, 2003.

[105] S. Takriti, J. R. Birge, and E. Long. A stochastic model for the unit commitment problem. *IEEE Transactions on Power Systems*, 11(3):1497–1508, 1996.

[106] Svyatoslav Trukhanov, Lewis Ntaimo, and Andrew Schaefer. Adaptive multicut aggregation for two-stage stochastic linear programs with recourse. *European Journal of Operational Research*, 206:395–406, 2010.

[107] S.S. Vallender. Calculation of the wasserstein distance between probability distributions on the line. *Theory of probability and its applications*, 18:784–786, 1973.

[108] L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review*, 38:49–95, 1996.

[109] Juan Pablo Vielma, Shabbir Ahmed, and George L. Nemhauser. A Lifted Linear Programming Branch-and-Bound Algorithm for Mixed-Integer Conic Quadratic Programs. *INFORMS Journal on Computing*, 20(3): 438–450, August 2008.

[110] Q. Wang, J. Wang, and Y. Guan. Stochastic unit commitment with uncertain demand response. *IEEE Transactions on Power Systems*, 28(1):562–563, 2013.

[111] Tapio Westerlund and Frank Pettersson. An extended cutting plane method for solving convex MINLP problems. *Computers & Chemical Engineering*, 19, Supplement 1:131–136, June 1995.

[112] Christian Wolf. *Advanced acceleration techniques for Nested Benders decomposition in Stochastic Programming*. PhD thesis, Universität Paderborn, 2013.

[113] L. Wu, M. Shahidehpour, and T. Li. Stochastic security-constrained unit commitment. *IEEE Transactions on Power Systems*, 22(2):800–811, 2007.

[114] A. Yoshise. Interior point trajectories and a homogeneous model for nonlinear complementarity problems over symmetric cones. *SIAM Journal on Optimization*, 17(4):1129–1153, 2006.

[115] B. Zeng and L. Zhao. Solving two-stage robust optimization problems using a column-and-constraint generation method. *Operations Research Letters*, 41(5):457–461, 2013.

[116] L. Zhao and B. Zeng. Robust unit commitment problem with demand response and wind energy. *IEEE Power and Energy Society General Meeting*, pages 1–8, 2012.

[117] L. Zhao, B. Zeng, and B. Buckley. A stochastic unit commitment model with cooling systems. *IEEE Transactions on Power Systems*, 28(1):211–218, 2013.