A Heuristic for Complementarity Problems Using Difference of Convex Functions

Steven A. Gabriel, Dominic Flocco, Trine K. Boomsma,
Martin Schmidt, Miguel A. Lejeune

Abstract. We present a new difference of convex functions algorithm (DCA) for solving linear and nonlinear mixed complementarity problems (MCPs). The approach is based on the reformulation of bilinear complementarity constraints as difference of convex (DC) functions, more specifically, the difference of scalar, convex quadratic terms. This reformulation gives rise to a DC program, which is solved via sequential linear approximations of the concave term using standard DCA techniques. The reformulation is based on a generalization of earlier results on recasting bilinearities and leads to a novel algorithmic framework for MCPs. Through extensive numerical experimentation, the proposed approach, referred to as DCA-BL, proves to be an efficient heuristic for complementarity problems. For linear complementarity problems (LCPs), we test the approach on a number of randomly generated instances by varying the size, the density, and the eigenvalue distribution of the LCP matrix, providing insights into the numerical properties of DCA-BL. In addition, we apply the framework to a market equilibrium problem and find that DCA-BL scales well on realistic LCP instances. Also, through experimentation, we find that that DCA-BL performs particularly well compared to other DC-based complementarity approaches in the literature if the LCP is highly dense, asymmetric, or indefinite. Lastly, the method is successfully applied to a set of equilibrium problems with second-order cone constraints, which give rise to nonlinear complementarity problems, with applications to stochastic equilibrium problems in water infrastructure and finance.

1. Introduction

Many nonlinear optimization problems contain so-called difference of convex (DC) functions. A DC function $f : \mathbb{R}^n \to \mathbb{R}$ can be expressed as a difference of convex (DC) functions $g, h : \mathbb{R}^n \to \mathbb{R}$, i.e., $f = g - h$. For a DC optimization problem, the objective function and all constraints are DC functions. Overviews of the theoretical background, DC-based algorithms (DCA), as well as numerical experiments in various application areas are provided by An and Tao (2005), Dinh Tao and Le An (1997), and Le Thi and Pham Dinh (2018, 2023) as well as by the more recent contributions of Le Thi (2020), Le Thi et al. (2019), and Oliveira (2020).

Of particular interest for this work are iterative DC algorithms that rely on successive DC decompositions. Given a particular DC decomposition $f = g - h$, such an algorithm solves a convex problem at iteration k using the approximation $f^k(x) := g(x) - h_k(x)$, where h_k is a linear approximation of h. The methodology provides a powerful extension of convex programming that is capable of covering a wide array of nonlinear and nonconvex programs. For instance, the approach has proven successful in solving nonconvex quadratic programs (Pham Dinh et al. 2008), binary linear and quadratic programs (Le Thi and Dinh 2001), nonlinear bilevel programs (Hoai An et al. 2009), and complementarity problems (Le Thi and Pham Dinh 2011). In this paper, we present a variant of these methods that is specifically tailored for complementarity problems. While not enjoying the theoretical convergence properties of DCA, our proposed heuristic approach exhibits very good numerical performance.

Date: October 7, 2025.
Key words and phrases. DC programming, Complementarity problems, Equilibrium problems, Infrastructure modeling, Portfolio selection.
Supplementary Material. Replication files, including data and code, are provided in the GitHub repository (Gabriel et al. 2025a). Appendices are provided in the online supplement to this manuscript.

Although DCA variants have been deployed to solve linear complementarity problems (Jara-Moroni et al. 2018; Le Thi et al. 2023), the application of DCA variants for solving linear and nonlinear equilibrium problems is relatively new to the literature. Such problems generalize the Karush–Kuhn–Tucker (KKT) conditions of one or more constrained optimization problems and cover many problems in non-cooperative game theory, economics, and engineering, in addition to a host of important areas in mathematical optimization itself (Cottle et al. 2009; Gabriel et al. 2012). Our focus is on equilibrium problems that can be formulated as mixed complementarity problems (MCPs). For mixed linear complementarity problems (MLCPs) there has been a long history of algorithms, some based on pivotal methods (e.g., Lemke's method), others on iterative methods (e.g., matrix splitting methods with projection). Some MLCP methods, including zero-finding for residue or complementarity functions, are likewise applicable to solving MCPs; see Gabriel et al. (2012).

Recent contributions to the literature apply DCA to LCPs. An important example includes the work by Le Thi and Pham Dinh (2011), in which the authors propose four optimization problems based on linearly constrained quadratic optimization, concave separable minimization, bound-constrained quadratic programming, and a bilinear program, along with four DC-based reformulations. Similarly, Le Thi and Pham Dinh (2011) develop a local DC-based optimization approach for solving LCPs, which minimizes a nonconvex complementarity function subject to linear constraints. The authors find that the linearly constrained quadratic programming approach, referred to as DCA1, and the concave separable minimization approach, referred to as DCA2, show the best performance. Convergence theory is presented along with numerical comparisons with each other and with Lemke's method. For this reason, the algorithms DCA1 and DCA2 also serve as benchmarks in the numerical testing of our new approach.

In more recent work, Le Thi et al. (2023) consider complementarity functions that admit DC decompositions, including the min- and the Fischer–Burmeister functions (Harker and Pang 1990; Pang and Gabriel 1993), to be minimized in the objective. The resulting optimization problem has a DC objective function with linear constraints and standard DCA is applied. The authors prove theoretical convergence and also carry out numerical experiments. The contribution of our work is an extensive study of an alternative strategy for the application of DCA to complementarity problems through bilinear constraints, referred to as DCA-BL:

(i) Similar to the work of Le Thi et al. (2023) and Le Thi and Pham Dinh (2011), we express the complementarity conditions as constraints with bilinear terms, the violations of which are penalized in the objective. This is already an established approach for solving mathematical programs with complementarity constraints as nonlinear optimization problems; see, e.g., Fletcher and Leyffer (2004) and Scheel and Scholtes (2000). We show that the bilinear terms admit DC decompositions such that the resulting optimization problem has a linear objective with DC constraints. Like Le Thi et al. (2023), our DCA approach is a sequential quadratic programming approach.

However, in contrast to standard DCA, our subproblems have a linear objective function and convex quadratic constraints. Furthermore, we propose specific DC decompositions of the bilinear terms by differences of smooth convex quadratic functions. This allows us to easily evaluate the derivatives and obtain linear approximations by uniquely determined supporting hyperplanes to the concave part of a DC function. Our DC decompositions generalize previous suggestions from the literature; see, e.g., Constante-Flores et al. (2022), Gabriel et al. (2006), and Siddiqui and Gabriel (2013).

(ii) We carry out extensive numerical experiments of our DCA approach for a wide variety of LCP test problems, including LCP instances found in the literature, randomly generated instances with varying density, spectrum, and symmetry properties of the LCP matrix, and finally, for

a real-world natural gas market equilibrium problem with both price-making or price-taking behavior of the producers and with varying number of producers and number of time periods. Furthermore, we extend our study to nonlinear MCPs and demonstrate our DCA approach for chance-constrained optimization via second-order cone programming. We present two real-world applications to market equilibrium problems, including a stochastic multi-portfolio equilibrium selection problem and a stochastic water equilibrium resource problem.

It should be remarked that, although our focus is on complementarity problems, our DCA approach may be extended to a wider class of bilinear problems. Bilinear terms appear in a number of classic operations research problems, an example being mathematical programs with equilibrium constraints (MPECs). As an example of an application, an upper level determines production decisions, while a lower level represents a market equilibrium with endogenous prices. The upper level objective function includes a revenue term that is bilinear in price and production (Gabriel et al. 2012; Labbé and Violin 2013).

Our numerical experiments reveal that DCA-BL compares favorably with existing methods. First, we compare DCA-BL to the DC-based LCP methods by Le Thi and Pham Dinh (2011) and Pham Dinh et al. (2008). Our approach is shown to outperform other DCA methods for most of the problem instances found in the literature. Moreover, we employ DCA-BL on fully dense LCPs up to size $n = 5000$, extending the largest instance size of $n = 3000$ previously explored in the DC-based complementarity literature (Le Thi and Pham Dinh 2011). Next, we demonstrate the effectiveness of the DCA-BL heuristic for the randomly generated LCP instances and find that it performs particularly well for dense, asymmetric, or indefinite LCP matrices. To show that the results are not solver-specific, we test these findings using several convex optimization solvers. The practical usefulness of the DCA-BL approach for LCPs is illustrated for a class of market equilibrium problems. Lastly, we further exhibit the versatility of DCA-BL on a selection of nonlinear MCPs that arise from chance-constrained equilibrium problems.

The rest of the paper is organized as follows. In Section 2, we motivate the study of MCPs and provide necessary theoretical background on complementarity problems. Section 3 reformulates MCPs as bilinearly constrained optimization problems and recasts these nonconvex constraints as DC functions, giving rise to an equivalent DC program. This DC program can be solved using the general DC algorithm by Le Thi et al. (2014), as outlined in Section 4, along with the convex subproblem used in the proposed DCA-BL approach. In Section 5, we carry out extensive tests of DCA-BL on a wide variety of LCPs, and in Section 6, we present results on nonlinear complementarity problems. The paper closes in Section 7 with some concluding remarks and further directions for future research.

2. Complementarity Problems

Given the vector-valued functions $F^1 = (F_1, \ldots, F_{n_1}) : \mathbb{R}^n \to \mathbb{R}^{n_1}$ and $F^2 = (F_{n_1+1}, \ldots, F_n) : \mathbb{R}^n \to \mathbb{R}^{n-n_1}$, the goal of the mixed complementarity problem (MCP) is to find vectors $x^1 = (x_1, \ldots, x_{n_1})^\top \in \mathbb{R}^{n_1}$ and $x^2 = (x_{n_1+1}, \ldots, x_n)^\top \in \mathbb{R}^{n-n_1}$ that satisfy

$$0 \leq F^1(x^1, x^2) \perp x^1 \geq 0, \tag{1a}$$

$$0 = F^2(x^1, x^2), \ x^2 \text{ free}, \tag{1b}$$

where the complementarity condition $F^1(x^1, x^2) \perp x^1$ is a shorthand notation for $F^1(x^1, x^2)^\top x^1 = 0$, which again is equivalent to the component-wise conditions $F_i(x^1, x^2)x_i^1 = 0$ for $i = 1, \ldots, n_1$. If $n_1 = n$, the problem is simply referred to as a complementarity problem.

If the functions F_i^1 for $i = 1, \ldots, n_1$ and F_i^2 for $i = n_1 + 1, \ldots, n$ are affine-linear, the problem is referred to as a mixed linear complementarity problem (MLCP), which can be written as

$$0 \leq M_{11}x^1 + M_{12}x^2 + q_1 \perp x^1 \geq 0, \tag{2a}$$

$$0 = M_{21}x^1 + M_{22}x^2 + q_2, \ x^2 \text{ free}, \tag{2b}$$

where $M_{11} \in \mathbb{R}^{n_1 \times n_1}, M_{12} \in \mathbb{R}^{n_1 \times (n-n_1)}, M_{21} \in \mathbb{R}^{(n-n_1)\times n_1}, M_{22} \in \mathbb{R}^{(n-n_1)\times(n-n_1)}$ and $q_1 \in \mathbb{R}^{n_1}$, $q_2 \in \mathbb{R}^{n-n_1}$ are given matrices and vectors, respectively.

It is well-known that complementarity problems arise in many fields of applications such as energy markets, contact mechanics, or in game theory and as optimality conditions of certain optimization problems. We refer the interested reader to the textbooks (Cottle et al. 2009; Gabriel et al. 2012) for more detailed discussions of such applications.

3. Difference of Convex Functions

In this section, we reformulate MCPs as bilinearly constrained problems and recast the bilinear terms as DC functions.

3.1. Reformulation of MCPs as Bilinearly Constrained Problems.

The MCP (1) can be expressed as a bilinearly constrained problem by introducing a vector of auxiliary variables $y \in \mathbb{R}^{n_1}$:

$$0 = F^1(x^1, x^2) - y, \tag{3a}$$

$$0 = y^\top x^1, \ x^1, y \geq 0, \tag{3b}$$

$$0 = F^2(x^1, x^2), \ x^2 \text{ free}. \tag{3c}$$

Here, the bilinear constraint (3b) captures the complementarity. The problem is equivalent to

$$(3a), (3c), \quad 0 = y_i x_i^1, \ x_i^1, y_i \geq 0, , \quad i = 1, \ldots, n_1. \tag{4}$$

Regardless of whether the MCP is linear or not, the bilinear constraint renders the problem nonconvex. Next, we present an exact reformulation of the bilinear constraint using DC functions.

3.2. Bilinear Terms as Differences of Convex Functions.

The following formally defines a difference of convex (DC) functions.

Definition 1. *A function $f : \mathbb{R}^n \to \mathbb{R}$ is a DC function if there exist convex functions $g, h : \mathbb{R}^n \to \mathbb{R}$ such that*

$$f(z) = g(z) - h(z)$$

for all $z \in \mathbb{R}^n$.

The above is also called a *DC decomposition* of f. Note that the DC decomposition is not unique.

Now, consider the complementarity constraint (3b). For $y, x \in \mathbb{R}^n$, we can recast each of the bilinear terms $y_i x_i$, $i = 1, \ldots, n$, via a difference of convex quadratic functions by introducing auxiliary variables. The DC decomposition can be obtained from the following result.

Theorem 1. *Let $y, x \in \mathbb{R}$ be given and let $u, v \in \mathbb{R}$ be defined by*

$$\begin{pmatrix} y \\ x \end{pmatrix} = N \begin{pmatrix} u \\ v \end{pmatrix} \quad \text{with} \quad N = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

satisfying (i) $ad - bc \neq 0$, (ii) $ad + bc = 0$, and (iii) $ac \geq 0$. Then, N is invertible and there exist constants $\alpha, \beta > 0$ such that $yx = \alpha u^2 - \beta v^2$.

Proof. First of all, the invertibility of N is implied by the determinant condition (i). From (ii), we obtain

$$yx = (au + bv)(cu + dv) = acu^2 + (ad + bc)uv + bdv^2 = acu^2 + bdv^2.$$

By (ii), if $d = 0$ then $b = 0$ or $c = 0$, which would render the matrix N singular. Hence, $d \neq 0$. A similar line of reasoning shows that $a \neq 0$ holds and, consequently, all elements of N are non-zero. Let $\alpha = ac$. By (iii), $\alpha \geq 0$. Moreover, $\alpha > 0$ since $a, c \neq 0$. Let $\beta = -bd$. By (ii), $ad = -bc$, which after multiplying by $d \neq 0$ and dividing by $a \neq 0$ on both sides of the equality gives $0 < d^2 = -bdc/a$. Since $ac > 0$, also $c/a > 0$, and we have that $bd < 0$. □

By Theorem 1, we obtain the following reformulation of the MCP (4):

$$F^1(x^1, x^2) - y = 0, \tag{5a}$$

$$\begin{pmatrix} y_i \\ x_i^1 \end{pmatrix} - N^i \begin{pmatrix} u_i \\ v_i \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad i = 1, \ldots, n_1, \tag{5b}$$

$$\alpha_i u_i^2 - \beta_i v_i^2 \leq 0, \quad i = 1, \ldots, n_1, \tag{5c}$$

$$-\alpha_i u_i^2 + \beta_i v_i^2 \leq 0, \quad i = 1, \ldots, n_1, \tag{5d}$$

$$F^2(x^1, x^2) = 0, \tag{5e}$$

$$x^1, y \geq 0, \tag{5f}$$

$$x^2, u, v \text{ free}, \tag{5g}$$

where $u, v \in \mathbb{R}^{n_1}$ of variables are vectors and $N^i \in \mathbb{R}^{2\times 2}$ are invertible matrices with entries a_i, b_i, c_i, and d_i satisfying (i)–(iii) of Theorem 1. Moreover, we have $\alpha_i = a_i c_i$ and $\beta_i = -b_i d_i$ for $i = 1, \ldots, n_1$. It should be noted that (5d) is redundant because of (5f), since $y_i x_i^1 = \alpha_i u_i^2 - \beta_i v_i^2 \geq 0$ by the nonnegativity of x^1 and y. However, we include it in the formulation, since this will allow for approximation of $\alpha_i u_i^2 - \beta_i v_i^2$ from both above and below. Note also that an alternative reformulation is obtained from MCP (3):

$$(5a), (5b), (5e), (5f), (5g), \quad \sum_{i=1}^{n_1} (\alpha_i u_i^2 - \beta_i v_i^2) \leq 0, \quad -\sum_{i=1}^{n_1} (\alpha_i u_i^2 - \beta_i v_i^2) \leq 0. \tag{6}$$

For the remainder of the paper, we assume that F_i^1 for $i = 1, \ldots, n_1$ and F_i^2 for $i = n_1 + 1, \ldots, n$ are likewise DC functions. Under this assumption, the problems (5) and (6) may be nonconvex, but only involves DC functions, and we refer to them as a DC feasibility problem.

For the MLCP, the specific DC decomposition of the bilinear terms produces a quadratically constrained DC feasibility problem of the form (5) with

$$F^1(x^1, x^2) = M_{11}x^1 + M_{12}x^2 + q_1 \quad \text{and} \quad F^2(x^1, x^2) = M_{21}x^1 + M_{22}x^2 + q_2 \tag{7}$$

The reformulations (5) involve $2n_1$ new variables and n_1 new constraints compared to Problem (4). By substitution, however, the original variables can be deleted. Using that $y = \operatorname{diag}(a)u + \operatorname{diag}(b)v$ and $x^1 = \operatorname{diag}(c)u + \operatorname{diag}(d)v$, where a, b, c, d are defined in Theorem 1 and $\operatorname{diag}(a) \in \mathbb{R}^{n_1 \times n_1}$ is the matrix with entries $a_1, \ldots, a_{n_1}$ on the diagonal, Problem (5) may be written equivalently with just free variables variables u, v and x^2. The resulting problem involves as many variables and constraints as the original Problem (4). After substitution, the problem the may not be a DC problem, in general, but for the MLCP, the substitution produces another quadratically constrained DC problem.

4. DC Algorithms for Complementarity Problems

We now make use of DC algorithms to solve Problem (5). In particular, we focus on the approach to general DC programs with DC constraints, referred to as *general DCA2* (GDCA2), by Le Thi et al. (2014) and adapt it to our specific DC reformulation.[1]

To outline the algorithm, consider a general DC feasibility problem of the form

$$f_i(z) \leq 0, \quad i = 1, \ldots, n, \tag{8}$$

where $f_i = g_i - h_i$ and g_i, h_i are convex functions for $i = 1, \ldots, n$. The basic idea of GDCA2 is to apply a linear approximation to the concave part of each DC function f_i, i.e., to $-h_i$ and employ a sequential convex programming method to solve the approximated problem. In particular, the DC constraints are

[1]More generally, a DC program can have an objective function and inequality constraints that are DC. As will be shown below, we specialize such a DC program to the MCP case in which the constraints related to complementarity are re-expressed as pairs of DC inequalities while the other constraints are linear or convex. Moreover, the objective function we use is a scalar penalty function, hence linear to test for feasibility of the resulting DC subproblem.

replaced by

$$g_i(z) - h_i(z^k) - (\eta_i^k)^\top (z - z^k) \leq 0, \quad i = 1, \ldots, n, \tag{9}$$

where z^k is the iterate and η_i^k is a subgradient of h_i at z^k, i.e., $\eta^k \in \partial h_i(z^k) = \{\eta \in \mathbb{R}^n \colon h_i(z) - h_i(z^k) \geq \eta^\top (z - z^k)\ \forall z \in \mathbb{R}^n\}$, in iteration k. For the specific DC decomposition of each bilinear term by a difference of differentiable convex functions, we can simply replace the subgradient by the gradient $\nabla h_i(z^k)$ of h_i at z^k. Our linear approximations uniquely determine supporting hyperplanes to the concave part of a DC function. By approximating $-h_i(z)$ from above, (9) becomes a restriction of (8).

As noted by Le Thi et al. (2014), the problem (9) may be infeasible. Hence, we relax the constraints and penalize violations in the objective function. In particular, the objective minimizes a slack variable to obtain an ℓ_∞-norm penalty term, i.e., we obtain

$$\min_{z,t} \quad \rho_k t \tag{10a}$$

$$\text{s.t.} \quad g_i(z) - h_i(z^k) - (\eta_i^k)^\top (z - z^k) \leq t, \quad i = 1, \ldots, n, \tag{10b}$$

$$t \geq 0, \tag{10c}$$

where $\rho_k > 0$ is the penalty parameter in iteration k. GDCA2 is presented in Algorithm 1.

Algorithm 1 GDCA2 using slack variables and adaptive penalty parameters (Le Thi et al. 2014)

1: *Initialization.* Find an initial point z^1. Choose parameters $\delta_1, \delta_2 > 0$, an initial penalty parameter value $\rho_1 > 0$, and set $k = 1$.
2: *Subproblem.* Solve the convex subproblem (10) to obtain an optimal solution (z^{k+1}, t^{k+1}) and corresponding Lagrange multipliers $(\lambda^{k+1}, \mu^{k+1}) \in \mathbb{R}^{n+1}$.
3: *Convergence check.* If $z^{k+1} = z^k$ and $t^{k+1} = 0$, then STOP and return z^{k+1} as a solution to (8). Otherwise, go to Step 4.
4: *Penalty parameter update.* Let

$$r_k := \min\left\{||z^{k+1} - z^k||_2^{-1}, ||\lambda^{k+1}||_1 + \delta_1\right\},$$

where $\|\cdot\|_1$ is the ℓ_1-norm and $\|\cdot\|_2$ is the ℓ_2-norm. Set

$$\rho_{k+1} = \begin{cases} \rho_k, & \text{if } \rho_k \geq r_k, \\ \rho_k + \delta_2, & \text{if } \rho_k < r_k. \end{cases}$$

5: *Iterate.* Set $k \leftarrow k + 1$ and go to Step 2.

We solve the DC program (5) via the general DCA2, and refer to the approach as bilinear DCA (DCA-BL). Let F^1 and F^2 be DC functions with DC decompositions

$$F_i^1(x^1, x^2) =: G_i^1(x^1, x^2) - H_i^1(x^1, x^2), \quad i = 1, \ldots, n_1,$$
$$F_i^2(x^1, x^2) =: G_i^2(x^1, x^2) - H_i^2(x^1, x^2), \quad i = n_1 + 1, \ldots, n,$$

where $G^1, H^1 : \mathbb{R}^n \to \mathbb{R}^{n_1}$ and $G^2, H^2 : \mathbb{R}^n \to \mathbb{R}^{n-n_1}$ are vector-valued functions with each component G_i^1, H_i^1 for $i = 1, \ldots, n_1$ and G_i^2, H_i^2 for $i = n_1 + 1, \ldots, n$ being a convex function. The convex subproblem

solved at each DCA-BL iteration (see Algorithm 1) is given by

$$\begin{aligned}
\min\quad & \rho_k t && && \text{(11a)}\\
\text{s.t.}\quad & G_i^1(x^1,x^2) - H_i^1(x^{1,k},x^{2,k}) - \partial_{x^1} H_i^1(x^{1,k},x^{2,k})^\top (x^1 - x^{1,k}) \\
& - \partial_{x^2} H_i^1(x^{1,k},x^{2,k})^\top (x^2 - x^{2,k}) - y_i \le t, && i = 1,\dots,n_1, && \text{(11b)}\\
& y_i + H_i^1(x^1,x^2) - G_i^1(x^{1,k},x^{2,k}) - \partial_{x^1} G_i^1(x^{1,k},x^{2,k})^\top (x^1 - x^{1,k}) \\
& - \partial_{x^2} G_i^1(x^{1,k},x^{2,k})^\top (x^2 - x^{2,k}) \le t, && i = 1,\dots,n_1, && \text{(11c)}\\
& \begin{pmatrix} y_i \\ x_i^1 \end{pmatrix} = N^i \begin{pmatrix} u_i \\ v_i \end{pmatrix}, && i = 1,\dots,n_1, && \text{(11d)}\\
& \alpha_i u_i^2 - \beta_i (v_i^k)^2 - 2\beta_i (v_i^k)\left(v_i - v_i^k\right) \le t, && i = 1,\dots,n_1, && \text{(11e)}\\
& \beta_i v_i^2 - \alpha_i (u_i^k)^2 - 2\alpha_i (u_i^k)\left(u_i - u_i^k\right) \le t, && i = 1,\dots,n_1, && \text{(11f)}\\
& G_i^2(x^1,x^2) - H_i^2(x^{1,k},x^{2,k}) - \partial_{x^1} H_i^2(x^{1,k},x^{2,k})^\top (x^1 - x^{1,k}) \\
& - \partial_{x^2} H_i^2(x^{1,k},x^{2,k})^\top (x^2 - x^{2,k}) \le t, && i = n_1+1,\dots,n, && \text{(11g)}\\
& H_i^2(x^1,x^2) - G_i^2(x^{1,k},x^{2,k}) - \partial_{x^1} G_i^2(x^{1,k},x^{2,k})^\top (x^1 - x^{1,k}) \\
& - \partial_{x^2} G_i^2(x^{1,k},x^{2,k})^\top (x^2 - x^{2,k}) \le t, && i = n_1+1,\dots,n, && \text{(11h)}\\
& x^1, y, t \ge 0 && && \text{(11i)}\\
& x^2, u, v \text{ free.} && && \text{(11j)}
\end{aligned}$$

Theorem 3.1 of Le Thi et al. (2014) establishes convergence of DCA2 under certain differentiability and convexity properties as well as the Mangasarian–Fromowitz constraint qualification (MFCQ). Unfortunately, the MFCQ does not hold in the presence of complementarity conditions, as demonstrated in Appendix A.1 of the online supplement. Nevertheless, the following shows that if DCA-BL stops, it is at a solution. Furthermore, we have convergence towards a solution, if the penalty parameter remains constant and the distance between iterates becomes sufficiently small from some iteration on of the algorithm. The proof of the next theorem follows directly from Le Thi et al. (2014). For completeness, we adapt it to our framework in Appendix A.2 of the online supplement.

Theorem 2. *Suppose that $F_i^1 = G_i^1 - H_i^1, F_i^2 = G_i^2 - H_i^2$ and G_i^1, H_i^1 and G_i^2, H_i^2 are continuously differentiable and convex functions for all $i \in \{1,\dots,n_1\}$ and $i \in \{n_1+1,\dots,n\}$, respectively. Let $\{z^k\}$ be a sequence produced by DCA-BL, where $z^k = (x^{1,k}, x^{2,k}, y^k, u^k, v^k)$. Then, either the algorithm stops after finitely many iterations at a solution z^k to (5) or it generates an infinite sequence of points. If there exists an iteration k_0 such that $\rho_k = \rho_{k_0}$ and $\|z^{k+1} - z^k\|_2 \le (\|\lambda^{k+1}\|_1 + \delta_1)^{-1}$ for all $k \ge k_0$, then every limit point z^∞ is a solution to (5).*

In the following two sections we carry out numerical experiments for DCA-BL on a number of test problems. More specifically, Section 5 presents results for a collection of linear complementarity problems (LCP) and compares it with various alternative approaches. In Section 6, we present a numerical analysis for two types of nonlinear complementarity problems that involve second-order cone constraints.

5. Numerical Results for Linear Complementarity Problems

In this section, we describe the numerical results of using DCA-BL on a wide variety of LCP test problems.

Specifically, we are concerned with classic LCPs of the form in (1) with $n_1 = n$, $F^1(x) = Mx + q$ for $M \in \mathbb{R}^{n\times n}$ and $q \in \mathbb{R}^n$. We focus on LCPs with nonnegative variables since adding free variables and associated equations would be rather straightforward but provide relatively less additional numerical insights. In Section 5.1, we describe the various LCP test problems, which are summarized in Table 1

TABLE 1. Overview of LCP test instances.

Problem Description	Additional Information	% Successful
Known Test LCPs	18 total test problems, 4 LCPs $n \in \{1000, 2000, 5000, 10000, 20000, 50000\}$	100 %
Random LCPs of moderate size	Varying M's density 750 total test problems 25 symmetric, PSD M, random eigenvalues $\in [0, 1]$ uniformly distributed random $q \in [0, 1]$ 10 densities for M in $\{0.1, 0.2, \ldots, 1.0\}$ 3 sizes, $n \in \{100, 200, 500\}$	93.5 %
Random LCPs of large size	Varying M's density 200 total test problems 10 symmetric PSD M, eigenvalues $\in [0, 1]$ uniformly distributed random $q \in [0, 1]$ 10 densities for M in $\{0.1, 0.2, \ldots, 1.0\}$ 2 sizes, $n \in \{1000, 2000\}$	94.0 %
Random LCPs of moderate size	Varying M's eigenvalues & symmetry 2250 total test problems 25 symmetric PSD M, eigenvalues $\in [0, 1]$ 25 SID M, eigenvalues $\in [-1, 1]$ 25 ASID M, eigenvalues $\in [-1, 1]$ 10 densities for M in $\{0.1, 0.2, \ldots, 1.0\}$ 3 sizes, $n \in \{100, 200, 500\}$	PSD: 93.2% SID: 50.2 % ASID: 76.8%
Market Equilibrium LCPs	Varying # of producers n_p Varying # of time periods n_t 720 total test problems $n_p \in \{2, 3, 5, 10, 12, 15\} \times$ $n_t \in \{2, 3, 5, 10, 12, 15\}$ 10 random instances for each of the 36 combinations producers as price-takers or price-makers largest LCP $n_t = n_p = 15$ resulting in $n = 480$ (price-takers), $n = 465$ (price-makers)	85.3 %

and comprise 3932 instances. Then, in Section 5.2, we employ DCA-BL on a set of LCP instances from the literature. In Section 5.2.1, we test the approach on random linear complementarity problems for which we vary the density, spectrum, and symmetry properties of the associated LCP matrix M. We also consider random right-hand side vectors q in some of the tests. In Section 5.3, we focus on a market equilibrium problem varying the price-making or price-taking behavior of the producers, the number of them, and the number of time periods. The associated model is described in (Gabriel et al. 2025b), where it is applied to the Brazilian gas market.

Unless stated otherwise, all experiments were run on an Apple Silicon M2 Max processor with 64 GB RAM, a 12-core CPU, and a 38-core GPU. The models and algorithms applied to LCPs are implemented using the Pyomo (Bynum et al. 2021; Hart et al. 2011) modeling package in Python and the convex subproblems are solved with MOSEK 10.1.27 (ApS 2019) for each of the DCA methods. Experiments on large-scale LCPs with dimension $n \geq 1000$ were run using a high-performance computing cluster running on 16 CPU cores with 5 GB RAM per core. Replication files, including data and code, are provided in the GitHub repository (Gabriel et al. 2025a).

We use the PATH solver (Ferris and Munson 1999) and the classic Lemke method as benchmarks. In addition, Le Thi and Pham Dinh (2011) offer four DC programming algorithms for solving LCPs and compare the algorithms' performance to Lemke's method. These methods differ from our bilinear approach in the formulation of the optimization model for the LCP. Out of these four approaches we

compare our bilinear DC programming method to DCA1 and DCA2 from Le Thi and Pham Dinh (2011), as these are the best performing ones in their tests. In addition, we compare our approach to Algorithm 4.2 from Pham Dinh et al. (2008), which we refer to as DCA-QP. For completeness, DCA1, DCA2, and DCA-QP are presented in Appendix C of the online supplement. For further details on Lemke's method, the reader is referred to Cottle et al. (2009).

Throughout experimentation, we refer to a successful solve if the solution method converges within the iteration limit and to a solution that satisfies $|(Mx+q)^\top x| < \varepsilon$ for a given tolerance $\varepsilon > 0$. In addition to recording total computational time and the number of iterations, we compute the time-per-solve for DC-based methods, which represents the average CPU time required to solve each convex subproblem, i.e., total computational time divided by number of iterations. Unless stated otherwise, the maximum iteration limit for DC-based methods (i.e., DCA-BL, DCA1, DCA2 and DCA-QP) is set to 500 and the iteration limit for Lemke's method is set to 3000.

We use the default settings for the PATH solver. In all cases, the convergence tolerance is set to $\varepsilon = 10^{-6}$ and the starting point $(x_1, \ldots, x_n, y)^\top = (0, \ldots, 0, 0)^\top$ was used. Furthermore, we use

$$N^i = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \quad i = 1, \ldots, n,$$

with $\alpha_i = \beta_i = 1$ in the definition of the auxiliary variables u_i, v_i for $i = 1, \ldots, n$; see Theorem 1. See Appendix F of the online supplement for a discussion on alternative choices of N^i.

5.1. **Description of Test Problems.** The LCP test problems considered can be grouped into three main categories: instances found in the literature, randomly generated instances with varying linear algebra properties, and application-specific instances. Table 1 shows the complete list of LCP test problems.

5.1.1. *LCP Test Problems from the Literature.* Le Thi and Pham Dinh (2011) compare the performance of DCA1 and DCA2 with Lemke's method on a set of positive semi-definite LCPs from the literature. The test problems considered in our numerical experiments are given in Appendix B.1 of the online supplement.

5.1.2. *Random LCP Test Problems.* We consider three sets of randomly generated LCPs: (i) moderate size with varying density, (ii) large size with varying density, and (iii) moderate size with varying spectra and symmetry properties. As usual, the matrix density is defined as the ratio of nonzero entries of the matrix and its size, i.e., ζ/n^2, where ζ is the number of nonzero entries. In addition, a "moderate" size problem is one with dimension $n < 1000$, while a "large" instance is one with dimension $n \geq 1000$.

For moderate-sized LCPs, we generate 25 random, symmetric, and positive semi-definite matrices M with eigenvalues uniformly distributed in $[0, 1]$, for each density $0.1, 0.2, \ldots, 0.9, 1.0$ and problem size $n \in \{100, 200, 500\}$. The random matrices are generated using MATLAB's sprandsym function. In addition, the vector q is randomly sampled from a uniform distribution on $[0, 1]$. This results in 750 problem instances. Similarly, for large-sized LCPs, we vary the density of the random LCP matrix M and increase the size of the LCP to $n \in \{1000, 2000\}$. For $n = 1000$ and $n = 2000$, we generate 100 random LCP instances; 10 instances for each density $0.1, 0.2, \ldots, 1.0$, resulting in 200 test problem instances.

For moderate-sized LCPs with varying spectra and symmetry properties, we randomly generate 25 problem instances with 10 different densities from 0.1 to 1.0 as before. However, we also vary the eigenvalue distribution of the underlying LCP matrix so that the resulting matrix is either symmetric and positive semi-definite (PSD), symmetric and indefinite (SID), or asymmetric and indefinite (ASID). Thus, we build 250 problem instances for each matrix class for sizes $n \in \{100, 200, 500\}$. The desired matrix properties are obtained by generating matrices with MATLAB's sprandsym function, where PSD matrices have eigenvalues uniformly distributed on $[0, 1]$, and SID matrices have eigenvalues uniformly distributed on $[-1, 1]$. In addition, in the SID and ASID cases, the vector q is generated so that the existence of a

solution to the LCP is guaranteed. Specifically, we randomly sample a solution vector x from a uniform distribution on $[0, 1]$ with 50 % nonzero entries and compute $q = -Mx$ as in Gabriel (2017).

5.1.3. *Application: Market Equilibrium Problems.* To demonstrate the usefulness of the DCA-BL approach in solving real-world problems, we proceed to apply the framework to a class of market equilibrium problems. We explore the game-theoretic framework of Nash equilibrium problems, a widely used format for problems involving multiple, non-cooperative agents in a market (Gabriel et al. 2012). Each player solves an optimization problem with the other players' decisions fixed at their optimal values. As such, multiple convex, quadratic optimization problems are solved simultaneously, and the complementarity problem consists of the KKT conditions of each player's optimization problem, giving rise to an LCP. The problem is applied to the Brazilian gas market in Gabriel et al. (2025b), and the data and problem size used in the testing of DCA-BL is representative. For simplicity, the market equilibrium framework does not consider any specific engineering constraints specific to gas or power production, but is a general model of market competition between producers.

Consider a set of players $P = \{1, \ldots, n_p\}$, each making their decision to maximize profit over a set of time periods $T = \{1, \ldots, n_t\}$. Accordingly, player $p \in P$ makes decisions $x^p = (x^p_t)_{t \in T}$ by solving the quadratic program with linear constraints

$$\max_{x^p} \quad f_p(x^p) = \pi^\top x^p - \frac{1}{2}(x^p)^\top Q^p x^p + (c^p)^\top x^p \tag{12a}$$

$$\text{s.t.} \quad A^p x^p \leq d^p, \quad (\lambda^p) \tag{12b}$$

$$x^p \geq 0, \tag{12c}$$

where the vector $\pi \in \mathbb{R}^{n_t}$ captures market prices, the matrix $Q^p \in \mathbb{R}^{n_t \times n_t}$ and the vector $c^p \in \mathbb{R}^{n_t}$ describe the convex quadratic costs, with Q^p being positive semi-definite, and where $A^p \in \mathbb{R}^{m \times n_t}$ and $d^p \in \mathbb{R}^m$ contain the data in the constraints of the problem, e.g., relating to maximum production rates or other operational considerations (Gabriel et al. 2025b). Finally, the vector $\lambda^p \in \mathbb{R}^m$ represents the Lagrange multipliers associated to the linear constraints. The KKT conditions are necessary and sufficient for optimality, which gives rise to an LCP. The mathematical details of the market equilibrium LCP considered for numerical testing is described in Appendix B.2 of the online supplement.

To analyze the performance of DCA-BL for this class of market equilibrium problems, we build random instances of the price-taker and price-maker models with varying problem sizes. Specifically, we build 10 random problem instances for the 36 combination of number of players $n_p \in \{2, 3, 5, 10, 12, 15\}$ and number of time steps $n_t \in \{2, 3, 5, 10, 12, 15\}$ for the price-maker and price-taker models, leading to a total of 720 problem instances up to size $n = 480$.

5.2. **Results: Linear Complementarity Problems.**

5.2.1. *Benchmarking with Other Methods.* We first apply DCA-BL and each of the five benchmark methods to the set of selected positive semi-definite LCPs of Section 5.1.1. This permits the comparison of DCA-BL to conventional MCP methods and other DC-based techniques for LCPs.

The numerical results given in Table 2 show that Lemke's algorithm is the best performing method for LCP6, while PATH exhibits the lowest computation time for LCP7. Moreover, DCA2 is fastest for LCP8 and LCP9. Compared to the other DC-based methods, DCA-BL has the lowest runtime in 5 out of 6 instances of LCP6 and LCP7, which correlates to the results on LCP matrix density in Section 5.2.2.

For each problem instance with size $n \in \{1000, 2000, 5000, 10\,000, 20\,000\}$, DCA-BL exhibits convergence in less than 5 iterations for all problem instances. This is comparable to the iteration counts of at most 2 for DCA2 and DCA-QP, while DCA1 converged in at most 32 iterations for larger instances. Note that the iteration counts for PATH and Lemke are not reported since the nature of the iterations in this iterative and pivoting method, respectively, are qualitatively different than the DC-based approaches. When

TABLE 2. Numerical comparison between the PATH solver, Lemke's method, DCA1, DCA2, DCA-QP, and DCA-BL on 18 large test instances with dimension $n \in \{1000, 2000, 5000, 10000, 20000, 50000\}$. A "–" indicates that the method does not converge within the prescribed maximum iteration limit. The best computational time among DC-based methods in each row is indicated by an asterisk, while the fastest time among all 6 methods is printed in **bold**.

ID	n	CPU Time (sec.)						Nr. of DC-based Iterations			
		PATH	Lemke	DCA1	DCA2	DCA-QP	DCA-BL	DCA1	DCA2	DCA-QP	DCA-BL
LCP6	1000	3.897	**0.012**	4.488	3.174	144.486	2.466⋆	1	2	2	1
	2000	25.033	**0.045**	18.226	16.331	1077.980	8.527⋆	1	2	2	1
	5000	96.524	**0.244**	167.393	184.963	18114.110	56.999⋆	1	2	2	1
LCP7	1000	**0.125**	1.968	0.704	0.509⋆	1.108	0.691	1	2	2	4
	2000	**0.208**	12.122	3.641	3.295	2.886	1.596⋆	2	2	2	4
	5000	**0.634**	–	22.708	62.687	19.224	4.790⋆	2	2	2	4
	10000	**2.804**	–	38.100	827.536	45.645	21.822⋆	3	2	2	4
	20000	**5.379**	–	384.820	6538.167	396.625	45.582⋆	32	2	2	4
	50000	**18.043**	–	1022.205	76034.878	1730.697	142.271⋆	32	2	2	4
LCP8	1000	0.109	1.960	0.753	**0.022**⋆	1.106	0.505	1	1	2	3
	2000	0.198	11.834	2.491	**0.039**⋆	2.883	1.282	1	1	2	4
	5000	0.595	–	20.889	**0.118**⋆	23.913	3.493	2	1	2	4
	10000	2.606	–	24.029	**0.862**⋆	47.845	16.474	16	1	2	3
	20000	5.573	–	–	**2.044**⋆	689.564	33.598	–	1	2	3
	50000	**11.603**	–	–	760.810	4669.391	101.196⋆	–	2	2	3
LCP9	1000	0.928	3.971	3.584	**0.882**⋆	137.647	1.871	1	1	2	1
	2000	3.800	–	16.640	**3.782**⋆	1066.657	7.351	1	1	2	1
	5000	21.238	–	106.383	**21.044**⋆	16220.452	56.806	1	1	2	1

analyzing the computational time for each solution method, observe that while DCA-BL computes solutions in comparable time to other methods for all instances, its performance compared to the benchmarks improves as the problem size increases. Figure 1 shows the computational time of each benchmark for problem instances of size $n = 5000$. Note that DCA-BL outperforms DCA1 in all problem instances, Lemke's method in three problem instances (LCP7, LCP8 and LCP9), and DCA2 and PATH in two problem instances (LCP6 and LCP7). As compared to the benchmarks, DCA-BL performs best on the LCP6 problem instance, with a CPU time less than PATH, DCA1, and DCA2. Note that LCP6 (see Section 5.1.1) is the only test problem which is fully dense. This performance against the benchmarks indicates that DCA-BL may perform particularly well on LCPs with dense coefficient matrices, which will be explored further in Section 5.2.2.

An interesting observation of the results in Table 2 is that DCA-BL often requires (slightly) more iterations to converge compared to DCA2 and DCA-QP, but reports a lower total solution time. This trend is particularly relevant on test problems LCP7 and LCP8. The convex subproblems for each of these methods differ: we have a linear program for DCA2, a convex-quadratic program with linear constraints for DCA-QP, and a problem with linear objective and quadratic constraints for DCA-BL. Hence, some variation in solution time for the convex subproblem can be expected, even when the same convex solver (MOSEK 10.1.27) was used for each algorithm. However, close examination of the results indicate that the high variance in time per iteration was not caused by the solution time for the convex subproblem, but the update of approximation parameters. DCA2 and DCA-QP are described in Appendix C of the online supplement. For DCA2, the approximation parameter $\eta^k \in \partial h_2(x^k)$ requires evaluating the sub-differential ∂h_2 at the current iterate x^k given by (C.2). This computation requires computing $Mx^k + q$. Similarly, for DCA-QP, the update for η^k requires computing $\rho I_n - (M + M^\top)x^k$. In both cases, the complexity of the update is $\mathcal{O}(n^2)$, which introduces additional computational overhead, even when using sparse linear

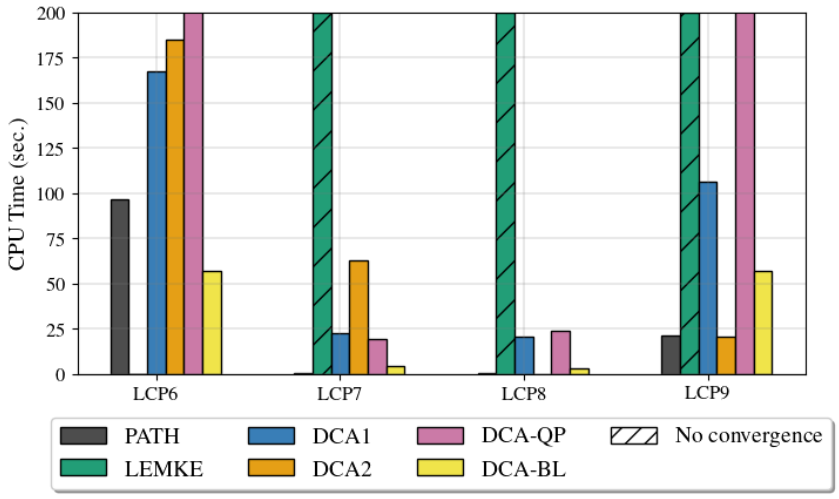

FIGURE 1. Computational time (in seconds) on LCP6, LCP7, LCP8, and LCP9 with $n = 5000$ for PATH, Lemke's method, DCA1, DCA2, DCA-QP, and DCA-BL.

algebra. By contrast, the update for DCA-BL only requires the update of vectors u^k and v^k in the convex subproblem of Line 2 of Algorithm 1, which is an update of order $\mathcal{O}(n)$.

A breakdown of the iteration time by solve and update is provided in Table E1 for LCP6, LCP7, LCP8, and LCP9 with $n = 2000$ to illustrate this point.[2] In almost all instances, the time required to update the approximation parameter η^k exceeds the convex subproblem solution time for DCA-QP. Conversely, the update is computationally inexpensive for DCA-BL and DCA1, as shown by the average update time of 0.02 seconds, obtained by the taking the average of the update columns of DCA-BL and DCA1. For LCP7 and LCP8, whose LCP coefficient matrices have a tridiagonal structure, the update time is significantly less than the update time required for fully-dense LCP6 and upper-triangular LCP9, showing the effect of sparse computations in reducing the update time. The time split for DCA1 is similar to that of DCA-BL: The solution time of the convex subproblem dominates the total time of each iteration. DCA1 has efficient update times similar to DCA-BL, i.e., $\mathcal{O}(n)$. However, we observe that the time for solving the convex quadratic subproblem of DCA1, which is quadratic with linear constraints, is slower than DCA-BL for dense problems LCP6 and LCP9. Overall, this indicates a potential benefit of DCA-BL over other DC-based methods and sheds light on the importance of the complexity of the approximation update on the algorithms' runtimes.

5.2.2. *LCP Matrix Density.* Next, we analyze the performance of DCA-BL on randomly generated LCPs by varying the density of the LCP matrix M. The study of density is motivated by numerical properties of modern MCP solvers such as PATH (Ferris and Munson 1999), which is an iterative method based on damped Newton steps. As such, iterative MCP methods solve a system of linear equations in each step, so faster solution times are expected if the coefficient matrix is more sparse. In the case of PATH, the coefficient matrix is stored as a sparse object to leverage sparse linear algebra in each Newton step (Dirkse and Ferris 1995). For further details on MCP algorithms, the reader is referred to (Cottle et al.

[2]Note that the experiments summarized Table 2 (and Figure 1) and Table E1 were run in different computational environments. Both were run on a high-performance computing cluster, as described in the beginning of Section 5, but with a different number of available cores and memory, and shared (rather than exclusive) access to compute nodes. As a result, there is a slight discrepancy between the total solve times reported in Tables 2 and E1. However, DCA-BL performance is still roughly the same in terms of its performance relative to other DC-based methods. In particular, the trends from Table 2 in the $n = 2000$ case remain largely the same: DCA-BL is the best performing DC-based method for LCP7, DCA2 is the fastest for LCP8 and LCP9. For LCP6, DCA2 now has a slightly faster runtime than DCA-BL. Nonetheless, the purpose of these experiments was to shed light on the complexity of parameter updates on solver performance.

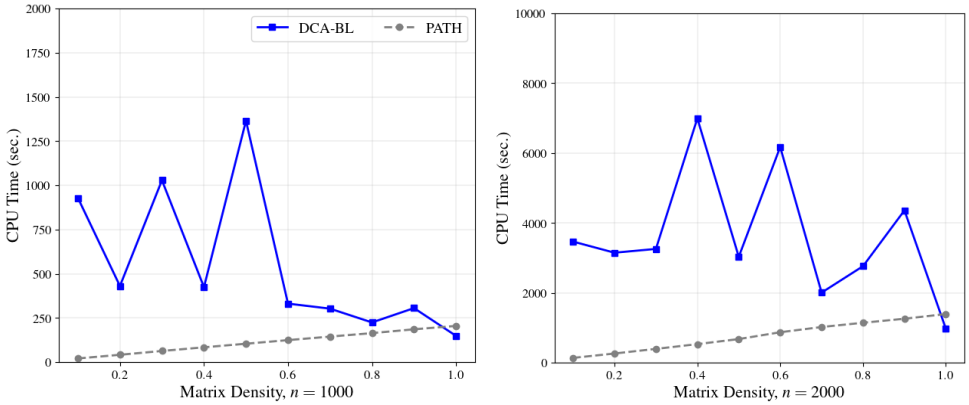

FIGURE 2. Average computational time (in seconds) of DCA-BL algorithm and PATH solver on random symmetric positive semi-definite LCPs with varying densities and size $n = 1000$ (left) and $n = 2000$ (right).

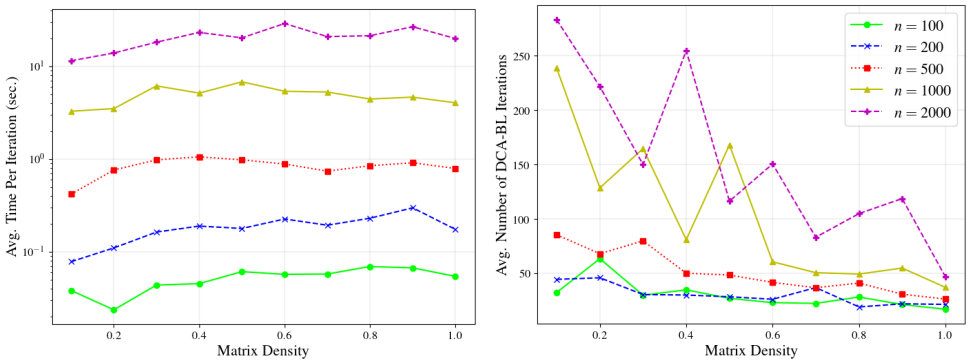

FIGURE 3. Left: Average time per DCA-BL iteration (in seconds). Right: Average number of DCA-BL iterations (log scale) until convergence. Both for $n \in \{100, 200, 500, 1000, 2000\}$ and varying LCP matrix densities.

2009; Gabriel et al. 2012). In contrast, DCA-BL is a sequential QP approach, in which an interior-point method is used to solve each subproblem, as is the case with MOSEK.

First, we compare the performance of DCA-BL and PATH related to the density of the underlying LCP. Figure 2 shows the average CPU time of DCA-BL and PATH on $n \in \{1000, 2000\}$ LCP problem instances by the density of the LCP matrix, which typify the results for smaller instances reported in Table E2. As expected, the total CPU time for PATH increases consistently with the density of the LCP matrix for each problem size, indicating that the problem instance becomes harder to solve as the coefficient matrix M becomes more dense. Interestingly, the runtime of the DCA-BL routine is non-monotonic as a function of density and the CPU time does not show a significant upward trend as seen for PATH. In the cases of $n \in \{1000, 2000\}$, the average CPU time is slightly lower than that of PATH for a fully dense coefficient matrix. This fact also holds for $n = 500$ as shown in Table E2.

To further examine this phenomenon, we record the time per solve and number of DCA-BL iterations to convergence for $n \in \{100, 200, 500, 1000, 2000\}$; see Figure 3 (left), which shows the average time per DCA-BL iteration for varying LCP densities, computed as the total computational time divided by the number of iterations for each problem instance. As such, this quantity represents average time to solve one convex subproblem of the form (11). The upward trend in average time per solve for sparse problems (density less than 0.4) indicates that MOSEK is presumably successfully exploiting the sparsity structure of the subproblem. However, the time per solve levels out for more dense problems and the average time per solve for a 50 % dense subproblem is longer or the same as that of a fully dense instance. Conversely,

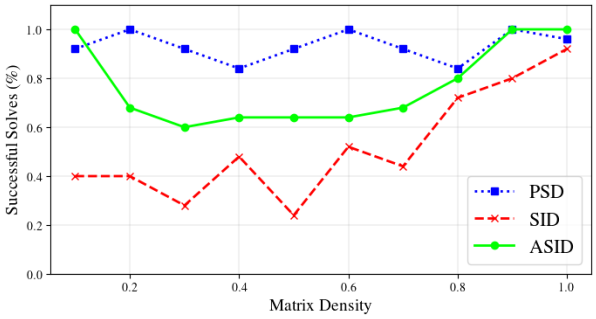

FIGURE 4. Successful solve rates of DCA-BL for random symmetric positive semi-definite (PSD), symmetric indefinite (SID), and asymmetric indefinite (ASID) LCP matrices of size $n = 500$.

we see a strong decreasing trend when we look at the average number of DCA-BL iterations versus density in Figure 3 (right)—DCA-BL converges in fewer iterations as the density of the LCP coefficient matrix increases. This indicates that the faster DCA-BL convergence as LCP density increases (as seen in Figure 2) is due to fewer DCA-BL iterations, rather than decreased time per solve. In addition to computing solutions with CPU time comparable to the benchmark (PATH), DCA-BL exhibits an interesting and unexpected property—namely faster convergence on more dense problems—which indicates a promising advantage over conventional LCP methods in solving dense LCPs.

5.2.3. *LCP Matrix Eigenvalue & Symmetry Properties.* Next, we explore the impact of varying the spectrum of the LCP matrix M on the DCA-BL performance. This exploration is motivated by conventional QP approaches to solving MCPs, where a quadratic complementarity term of the form $f(x) = x^\top(Mx+q)$ is minimized in the objective, with M being positive semi-definite. DCA1 and DCA2 from Le Thi and Pham Dinh (2011), are based on this QP approach, see Appendix C of the online supplement.

One feature of DCA-BL is that the LCP matrix M appears in the linear constraints of the DC program (**??**) rather than in the objective function. The potential computational benefit is that no iterative approximation of M is needed to ensure that the quadratic DCA subproblem objective (11) is convex, such as $M + \rho I_n$ in DCA1 and DCA-QP.

Selected results on random LCPs are presented in Figure 4, which displays the successful solve rates by density with PSD, SID, and ASID matrices for problem instances of size $n = 500$. Notice the trend that DCA-BL is most successful in solving LCPs with PSD matrices. Across all densities, the minimum successful solve rate for DCA-BL was 82 %, and the mean solve rate was 92 % for all 250 instances. Next, observe that DCA-BL applied to SID LCPs exhibits the worst successful solve rates for all densities, but increases to 90 % successful solves for fully dense matrices. This is a counter-intuitive result, as one would expect LCPs with symmetric matrices to be easier to solve. Here, it appears that the DCA-BL performance improves as the SID coefficient matrix becomes more dense. In the case of ASID LCPs, DCA-BL achieves a successful solve rate over 60 % for each density considered and much higher rates for more dense problems (e.g., for densities of at least 80 %). Numerical experiments indicate that LCP-tailored solvers such as PATH are well suited to solve SID matrices due to their symmetry. However, with DCA-BL we see the opposite trend: DCA-BL solves more asymmetric and indefinite LCPs successfully compared to symmetric and indefinite ones.

It should be noted that DCA-BL solves over 90 % of all problem instances, including PSD, SID, and ASID matrices, for fully dense matrices. The LCP test problems presented in Le Thi and Pham Dinh (2011) and described above in Section 5.1.1 have positive semi-definite matrices M, as noted in their original appearance in the literature (Byong-Hun 1983; Fathi 1979; Geiger and Kanzow 1996; Murty

TABLE 3. Successful solve rate, defined as the percentage of solves with $(Mx+q)^\top x < \varepsilon = 10^{-6}$, out of 200 random, asymmetric, and indefinite LCP problem instances with varying coefficient matrix densities.

n	PATH	DCA1	DCA2	DCA-BL
100	100 %	0.0 %	10.0 %	60.0 %
200	100 %	0.0 %	6.5 %	48.5 %
500	100 %	0.0 %	2.5 %	42.0 %
Total	100 %	0.0 %	6.3 %	50.2 %

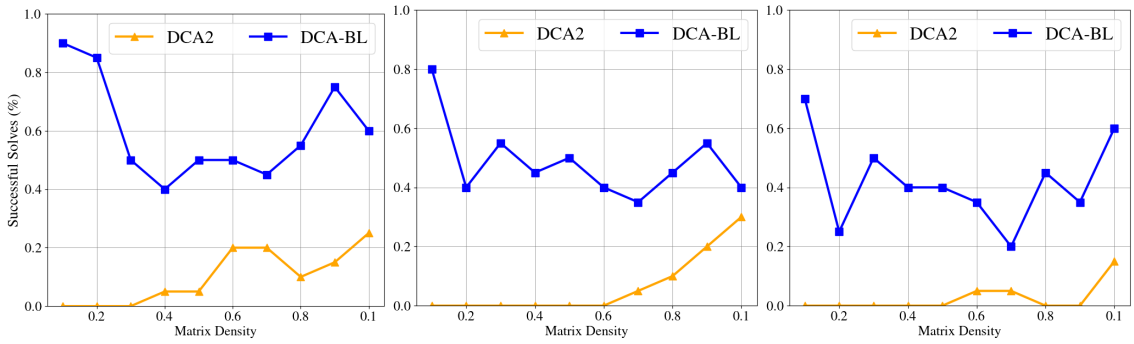

FIGURE 5. Successful solve rate for DCA2 and DCA-BL on random asymmetric indefinite (ASID) LCPs of size (A) $n = 100$, (B) $n = 200$, and (C) $n = 500$. Note that Lemke's method and DCA1 both exhibited a 0 % successful solve rate, hence these methods are omitted from the figure. PATH solved 100 % of instances for each problem size and thus is omitted for readability.

1988). To extend our numerical comparison and test the DCA-BL approach on more challenging LCPs, we build a set of randomly generated problem instances with asymmetric and indefinite matrices.

When compared to other benchmarks, the numerical results indicate that DCA-BL outperforms existing DC-based LCP methods when solving LCPs with asymmetric indefinite coefficient matrices. Table 3 shows the percentage of problem instances successfully solved by each solution method. The PATH solver successfully computes a solution for all problem instances, an expected result given that this method is a professional and commercial solver package that represents state-of-the-art complementarity algorithms. Lemke's method and DCA1 do not solve any problem instances successfully. Lemke's method potentially struggles due to the degeneracy property of the randomly generated SID matrices (i.e., determinant equals zero); noting Theorem 4.4.4 in Cottle et al. (2009), finite termination of Lemke's method is not guaranteed. In most cases, the iterates computed by DCA1 converge to a solution that does not satisfy the criterion $(Mx+q)^\top x < 10^{-6}$. An interesting comparison arises when considering the performance of DCA2 and DCA-BL. In general, we see that DCA-BL solved more problem instances than DCA2. Specifically, DCA-BL solved 50.17 % of the 600 problem instances successfully, while DCA2 only solved 6.33 %. Figure 5 shows the percentage of successful solves by coefficient matrix density. Observe that when the LCP coefficient matrix has low density, DCA2 is unable to solve any problem instances, and its successful solve rate increases with the density of the coefficient matrix. Conversely, as we explore in more detail in Section 5.2.2, the successful solve rate of DCA-BL is largely unaffected by the density of the LCP coefficient matrix, which further suggests that DCA-BL offers significant computational gain over other DC-based method when solving dense LCPs.

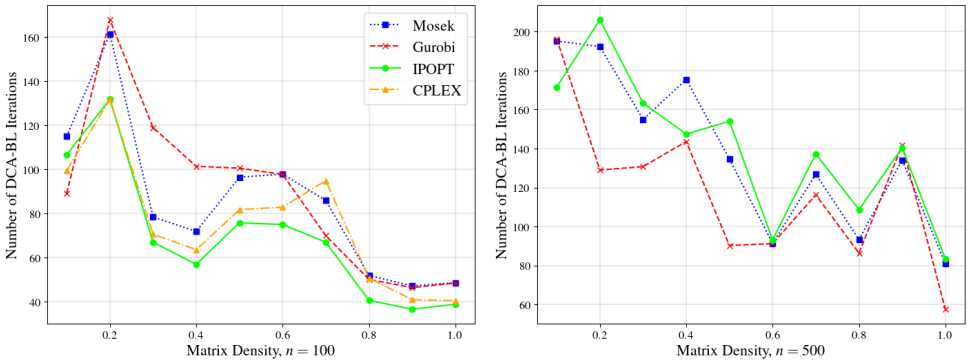

FIGURE 6. Average number of DCA-BL iterations required for convergence on random positive semi-definite LCPs of size $n = 100$ (left) and $n = 500$ (right) with different QCP solvers. Note that CPLEX did not terminate in a reasonable amount of time for the $n = 500$ case, thus its results are omitted in the bottom figure.

5.2.4. *Alternative Solvers.* The result that DCA-BL converges more rapidly as the underlying LCP becomes more dense is surprising. We experimented with the solver used to solve the convex DCA-BL subproblems. We use MOSEK (version 10.1.27) in our implementation due to its better performance in solving the subproblems compared to other popular solvers such as Gurobi, CPLEX, and IPOPT. In particular, MOSEK is particularly known for its exceptional performance in solving large-scale sparse convex programs efficiently (Cafieri et al. 2006). Now, we solve the same set of randomly generated, symmetric, and positive semi-definite LCPs in Section 5.2.2 with Gurobi (version 10.0.3), CPLEX (version 22.1.1), and IPOPT (version 3.14.16). The results for the randomly generated LCPs of size $n = 100$ and $n = 500$ from Section 5.2.2 are shown in Figure 6. While the time per solve varied for each solver, the average number of DCA-BL iterations for convergence followed the same trend regardless of the QCP solver used. We note that CPLEX exhibits significantly slower performance compared to MOSEK, Gurobi, and IPOPT for $n = 500$ and does not terminate in a reasonable amount of time, so its results are omitted in Figure 6 (right). Nonetheless, Figure 6 provides evidence that the results in Section 5.2.2 are not unique to the subproblem solver, but seems to be due to the properties of the DCA-BL framework. Additional numerical experiments were performed on larger instances and indicate a similar trend: the solver used has little impact on the number of DCA-BL iterations required for convergence. While analyzing the cause of the phenomenon is out of scope of this paper, the result provides an interesting insight into the use of DCA-BL in solving dense LCPs that requires further exploration.

An explanation for the results in Figure 4 could be the numerical aspects of the randomly generated ASID and SID matrices and the problem's dimension, rather than a property of the DCA-BL routine. To further investigate this, we conduct the same experiment using Gurobi and CPLEX to solve the convex subproblem. The successful solve rate for random ASID and SID $n = 100$ matrices is presented in Figure 7, which permits a comparison between these solvers and MOSEK 10.27.1, used for initial experimentation. The trend in successful solve rate is similar for each solver, but Gurobi is slightly better for less dense problems. An interesting observation from Figure 7 is that the variance between solve performance diminishes as the density of the coefficient matrix increase. This is likely due to less flexibility in the presolve phase of the convex solver for dense matrices. Nonetheless, the results in Figure 7 confirm that the high successful solve rate for dense, indefinite LCP matrices is independent of the chosen solver.

5.3. **Results: Market Equilibrium Problem.** Lastly, we analyze the application of DCA-BL to the market equilibrium problem described in Section 5.1.3. The results for the 720 problem instances solved by DCA-BL for the price-taker and price-maker model are summarized in Figure E1. The largest instance corresponds to $n_t = n_p = 15$ for the price-taker and price-maker setup, which leads to LCPs of order

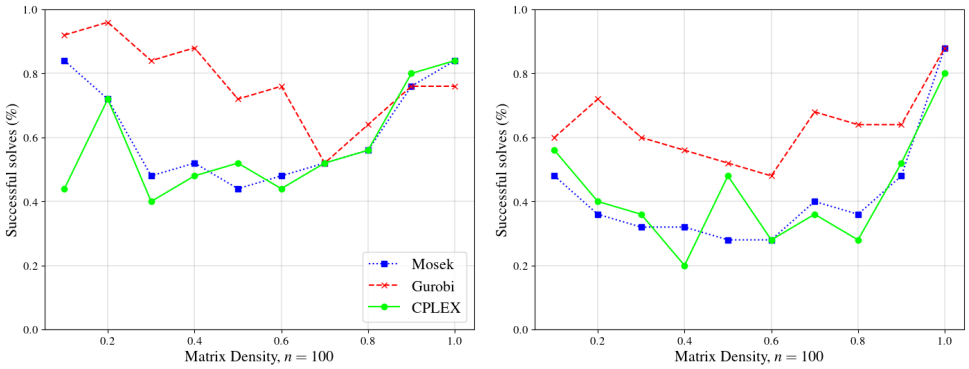

FIGURE 7. Successful solve rates of DCA-BL for LCP matrices of size $n = 100$ with MOSEK, Gurobi, and CPLEX QCP solvers. Left: asymmetric indefinite (ASID) LCP matrices. Right: symmetric indefinite (SID) LCP Matrices.

$n = 480$ and $n = 465$, respectively. First, observe the high successful solve rate for all problem instances, which is 80 % or higher for 57 of the 72 models (36 for price-taker, 36 for price-maker, each with varying number of time periods and players). For smaller problem instances, DCA-BL converges to the correct solution for over 80 % of the instances. In general, DCA-BL solves 85.7 % of the 360 price-taker problem instances and 84.8 % of the price-maker problem instances. The same experiment was also conducted using Gurobi and CPLEX for the convex subproblem. For the price-taker instances, the deviation in solve rate between MOSEK and Gurobi had a median of 10.0% and mean of 13.9%, and the between MOSEK and CPLEX had median 0.0% and mean 3.1%. All three solvers had the same successful solve rate for the price-maker instances. The low deviation suggests that the successful solve rates reported in Figure E1 again do not depend on the solver. The high successful solve rate, especially for small-to-medium sized problem instances, indicates that the DCA-BL approach is effective in solving LCPs that arise from equilibrium problems with multiple quadratic programs and system-wide constraints.

6. Numerical Results for Nonlinear MCPs

We now apply the DCA-BL approach to selected applications that lead to nonlinear MCPs. All experiments were run using the same computing architecture as described in Section 5. In addition to recording the total CPU time and the number of iterations, we again report the time-per-solve, which is computed as the CPU time divided by the number of DCA-BL iterations. As before, the stopping criterion is $\varepsilon = 10^{-6}$.

6.1. Chance-Constrained Optimization via DCA and Second-Order Cone Programming.

Stochastic optimization and equilibrium problems with chance constraints occur in a wide range of applications, and chance-constrained optimization is a powerful approach for decision-making under uncertainty (Ahmed and Xie 2018; Birge and Louveaux 2011; Conejo et al. 2010; Lejeune and Prékopa 2024; Prékopa et al. 2003). The deterministic equivalents of such stochastic problems give rise to convex, second-order cone problems for which the KKT conditions are sufficient and yield nonlinear MCPs. This allows for the simultaneous solution of multiple, second-order cone problems. For instance, in equilibrium problems some players may solve stochastic programs with chance constraints. We briefly review the concept of deterministic equivalents to derive the second-order constraints.

Let $\xi \in \mathbb{R}^n$ denote a Gaussian random vector and let $x \in \mathbb{R}^n$ be a decision vector. We consider chance constraints of the general form

$$\Pr\left(\xi^\top x \geq c\right) \geq \alpha, \tag{13}$$

TABLE 4. Overview of NCP test instances.

Problem Description	Additional Information	% Successful
NCP Multi-Portfolio Equilibrium	Second-order cone constraints Long-only positions model (300 instances) Long-short positions model (300 instances) Varying # of firms $n_p = \{2, 3, 5\}$ Varying # of portfolios n_k Varying # of assets n_j $(\mathcal{J}, \mathcal{A}) = \{(2,3), (3,5), (5,5), (5,7), (6,7)\}$ with $N = n_p n_k n_j$, $12 \leq n \leq 420$	77.6 %
NCP Water Market Equilibrium	Second-order cone constraints Ill-conditioned problem 2 players, 1 watershed 2 land uses (rural, urban) 1 pollutant (sediment) $n = 19$	1/1 problem solved successfully

where $\alpha \in [0, 1]$ is a prescribed probability level and $c \in \mathbb{R}$ is known data. The deterministic equivalent of the chance constraint (13) is given by

$$\mu^\top x + \phi^{-1}(1-\alpha)\sqrt{x^\top V x} \geq c, \tag{14}$$

where $\phi^{-1}(1-\alpha)$ is the $(1-\alpha)$-quantile of the cumulative probability distribution, V is the variance-covariance matrix of the Gaussian random variables $\xi_1, \ldots, \xi_n$ and $\mu_i = \mathbb{E}[\xi_i], i = 1, \ldots, n$. It is well-known that the constraint in (14) is convex as long as $\alpha \geq 0.5$ (Birge and Louveaux 2011) since $\phi^{-1}(1-\alpha) \leq 0$ holds and V is positive semi-definite.

Constraints of the form (14) are special cases of second-order cone constraints. Problems of this form provide an interesting application for the proposed bilinear DCA approach, which, to our knowledge, has yet to be explored in the literature.

Now, consider an equilibrium problem comprised of a set of players $P = \{1, \ldots, n_p\}$. Each player $p \in P$ solves the second-order cone problem with respect to x^p for fixed $x^{-p} = (x^q)_{q \in P \setminus \{p\}}$:

$$\min_{x^p} \quad f^p(x^p, x^{-p}) \tag{15a}$$

$$\text{s.t.} \quad h(x^p) = c^p - \mu^\top x^p - \phi^{-1}(1-\alpha^p)\sqrt{(x^p)^\top V^p x^p} \leq 0, \qquad (\lambda^p) \tag{15b}$$

$$g_i^p(x^p) \leq 0, \quad i = 1, \ldots, m, \qquad (\gamma_i^p) \tag{15c}$$

$$x^p \geq 0, \tag{15d}$$

where $f^p : \mathbb{R}^n \to \mathbb{R}$ is convex and quadratic, $h^p : \mathbb{R}^n \to \mathbb{R}$ is convex, and $g_i^p : \mathbb{R}^n \to \mathbb{R}$ is affine for all $i = 1, \ldots, m$. Also, $c^p \in \mathbb{R}$ and $\mu^p \in \mathbb{R}^n$ are given data and $V^p \in \mathbb{R}^{n \times n}$ is a given symmetric and positive semi-definite variance-covariance matrix. Finally, λ^p and γ_i^p are Lagrange multipliers. The KKT conditions are sufficient for optimality and necessary if a constraint qualification holds (Floudas 1995), giving rise to a nonlinear MCP. The resulting complementarity system may be reformulated as a DC program of the form (5) and can be solved using the DCA-BL framework with convex subproblems as given in (11). A derivation of the DC program for this particular class of nonlinear MCPs is given in Appendix D of the online supplement.

6.2. **Description of Test Problems.** The DCA-BL approach is applied to two equilibrium problems with chance constraints: a stochastic multi-portfolio equilibrium selection problem from Nguyen et al. (2024) and a stochastic water equilibrium resource problem from Boyd (2024). Table 4 shows the complete list of related nonlinear MCP test problems.

6.2.1. *Long-Only Multi-Portfolio Optimization.* We first consider an equilibrium model for multi-portfolio optimization explored by Lampariello et al. (2021) and Nguyen et al. (2024), which is motivated by the probabilistic Markowitz model by Bonami and Lejeune (2009). In this model, we have $P = \{1, \ldots, n_p\}$ firms competing in the same financial market in a non-cooperative random game. The firms invest in a set of portfolios $K = \{1, \ldots, n_k\}$, where each portfolio $k \in K$ is comprised by a set of assets A_k. Firm p must decide the percentage of its capital B_k^p to invest in each asset j of portfolio k, denoted by decision variable x_{kj}^p. The mathematical details of the long-only and long-short multi-portfolio equilibrium problems used in experimentation are described in Appendix B.3 of the online supplement.

To test the computational tractability of DCA-BL on solving the nonlinear multi-portfolio equilibrium problem, we use monthly return data for 450 stocks from Standard & Poor's 500 (S&P 500) index between 2012 and 2022. Using these historical data, we compute the geometric mean of each asset's returns and the variance-covariance matrix of all assets. Then, we build 20 problem instances containing $n_k n_j \in \{15, 25, 35, 42\}$ total assets distributed into $n_k \in \{3, 5, 5, 6\}$ portfolios of equal size n_j (i.e., $n_j = |A_k|$ for all $k \in K$), respectively, by randomly sampling stocks from the S&P 500. With the investment universe fixed, we solve each instance with $n_p \in \{2, 3, 5\}$ competing firms, resulting in 300 total instances. For each firm $p \in P$, we set a reliability level $\alpha^p = 0.9$, a budget of $B^p = n_j n_k$, and a minimum return level R^p as the average return of all assets in the universe. This means that $R^p = \frac{1}{n}\sum_{k,j} r_{k,j}$, where $r_{k,j}$ is the historical return of asset j in portfolio k and $n = n_j n_k$ is the total number of assets considered. Furthermore, the market-impact matrix for each $p \in P$ and $k \in K$ is initialized by computing a random matrix D_k^p whose entries are in $[0, 1]$ and then setting $\Omega_k^p = D_k^p (D_k^p)^\top / n_j$ to ensure that it is positive semi-definite. Lastly, we assume that ξ is normally distributed with mean and covariance matrix given by the historical data. This random data is used to test both the long-only and long-short versions of the model, as described in Appendix B.3 of the online supplement.

6.2.2. *Water Equilibrium Problem.* We proceed by considering the problem of infrastructure investment planning to reduce pollution in river systems from Boyd (2024). The total maximum daily load (TMDL) is the maximum amount of a pollutant allowed to enter a water body and serves as the primary metric used in the US to characterize pollutant-impaired water bodies. A TMDL is said to be implemented when enough investments are made to bring the pollutant load within the allowable amount. Local agencies and municipal governments are responsible for successful TMDL implementation and make pollution-reduction investment decisions within an allocated budget along these lines. The set of possible infrastructure investments, or treatment technologies, are referred to as best management practices (BMPs). In the US, nutrient trading of pollutants, such as nitrogen and phosphorus, is currently practiced to find cost-effective solutions to achieving water-quality goals. In addition, the degree of water pollution is inherently uncertain due to the effect of weather and human activity on the transport of pollutants throughout a river basin. Together, these considerations yield an interesting chance-constrained equilibrium model that analyzes infrastructure investment decisions and their effect on water quality. The mathematical details of the water equilibrium model are described in Appendix B.4 of the online supplement.

To apply the model, we build a stylized problem instance consisting of two players and a single watershed as described in Section 6 of Boyd (2024). In this small illustrative example, we consider two land-uses—rural and urban—and a single pollutant, sediment generated from erosive runoff from paved areas. Further details on how the problem data was generated is available in Boyd (2024).

6.3. **Multi-Portfolio Optimization and DCA-BL Scalability.** We apply DCA-BL to the long-only and long-short multi-portfolio optimization problems described in Section 6.2.1, with additional details in Appendix B.3 of the online supplement. The composition of portfolios and historical asset returns are the same for both models, but the long-short model has twice as many decision variables as the long-only model, due to the introduction of short positions. DCA-BL was employed on 300 random instances for

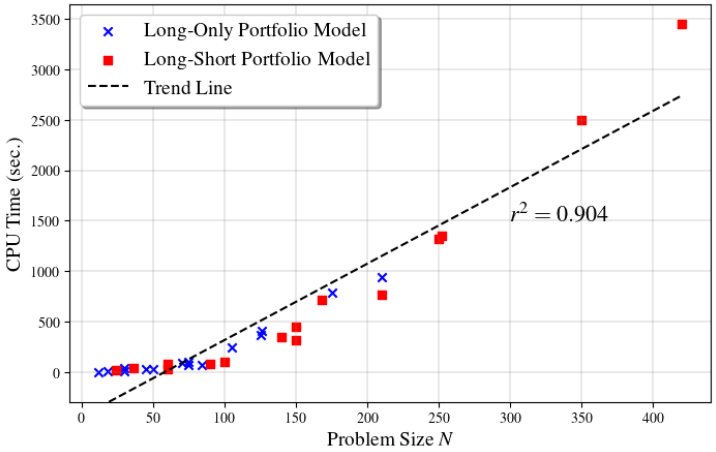

FIGURE 8. Computational time in seconds required for DCA-BL convergence on long-only and long-short multi-portfolio equilibrium problem with varying problem size N. For the long-only portfolio model $N = n_p n_k n_j$, and for the long-short model $N = 2n_p n_k n_j$.

each model, for a total of 600 instances of the nonlinear MCP ranging between $N = 12$ and $N = 420$ total decision variables. For each problem size, DCA-BL successfully converges within the prescribed maximum iteration count of 2500, which is increased from the linear case due to the added computational difficulty of this set of problems, for 77.6 % of the 600 problem instances, as indicated by the successful solve rates reported in Tables E3 and E4. The largest instance solved has 42 total assets and a total of 420 primal variables and 95 dual variables. This greatly exceeds those previously considered in the literature; see Lampariello et al. (2021), where the authors solve a model with 25 assets in the deterministic setting. Moreover, Nguyen et al. (2024) present a small computational example in a stochastic setting consisting of 9 assets.

The average computational performance across all problem instances is shown in Tables E3 and E4, and the trend in computation time for each problem size is illustrated in Figure 8. As expected, the computational time increases with the problem size—ranging from around 1 second (long-only positions, 2 firms, 6 assets) for the smallest instance to 1 hour for the largest one (long-short positions, 5 firms, 42 assets), which corresponds to an increase in both DCA-BL iterations and time-per-solve as the problem size grows. An interesting observation is that for each number of firms considered, the lowest successful solve rate corresponds to the scenario with 5 portfolios and 5 assets per portfolio, likely due to attributes of the randomly generated data for these instances. Nonetheless, Figure 8 indicates that the computational time required by DCA-BL increases almost linearly with the problem size of the nonlinear MCP. This observation is supported by the dashed trend line in Figure 8, which yields an r^2 value of 0.904, indicating a strong linear correlation.

6.4. **The Water Equilibrium Problem: Improved Performance with Domain Tightening.** Lastly, we consider the water equilibrium problem described in Section 6.2.2 , with additional details in Appendix B.4 of the online supplement. We note that convergence of DCA-BL was not achieved within 5000 iterations in the initial implementation, potentially due to the poor conditioning of the problem data. Specifically, the variance-covariance matrix used to implement the deterministic equivalent (B.9b) has a condition number of order 10^{20}. To address the initial difficulties in implementing DCA-BL on this ill-posed problem, we induce a domain reduction on the primal nonlinear MCP variables $I_{p,j}$ for each player. Specifically, we add the constraint

$$\sum_{j \in J} I_{p,j} \leq C_p \tag{16}$$

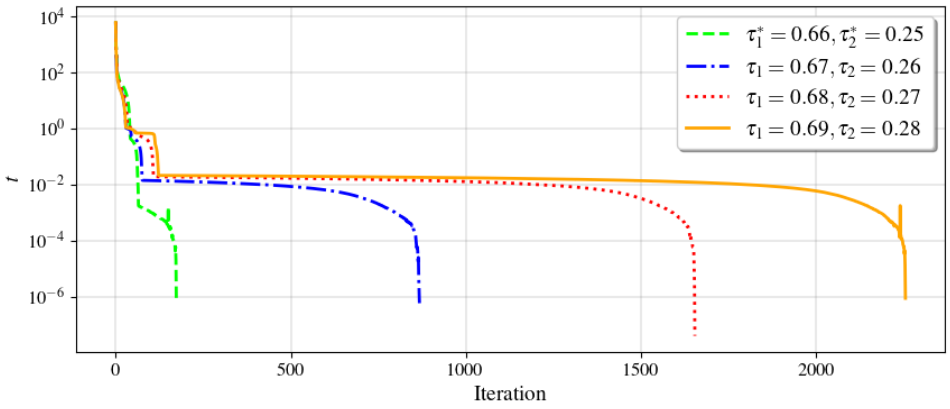

FIGURE 9. Convergence of water equilibrium nonlinear MCP using DCA-BL with varying bounds on primal nonlinear MCP variables, where t is the auxiliary variable minimized in the objective. The domain tightening constraint for player p is given by (16), with constant $C_p = \tau_p I_p^{\max}$. The parameters τ_1^* and τ_2^* are defined according to (18).

to the chance-constrained problem (B.9) of player p, where C_p is the upper-bound on player p's primal decision variables, which is generated a priori. Using problem specific data for the water equilibrium problem, we observe that the investment level of player p in land segment j is upper bounded by $i_{p,j}^{\text{up}}$ in (B.9c). Using the constant

$$I_p^{\max} = \sum_{j \in J} i_{p,j}^{\text{up}} \tag{17}$$

for all players $p \in P$, a natural choice for the domain tightening constraint is $C_p = I_p^{\max}$, or the sum of the upper bounds on player p's investments across land segments. In general, the application of domain tightening is specific to the problem considered and may rely on problem-specific data.

To experiment with the effect of domain tightening on DCA-BL convergence when solving the nonlinear water equilibrium MCP, we vary the constant C_p. Specifically, we define $C_p = \tau_p I_p^{\max}$, where $\tau_p \in [0, 1]$ is a measure of the tightness of player p's domain tightening constraint (16). We expect faster DCA-BL convergence the closer C_p is to the sum of optimal decision variables $C_p^* = \sum_{j \in J} I_{p,j}^*$. First, we perform an initial grid search on τ_p by solving the model with DCA-BL with the 100 combinations of $\tau_1 \in \{0.1, 0.2 \ldots, 1.0\}$ and $\tau_2 \in \{0.1, 0.2, \ldots, 1.0\}$. The results for each trial are reported in Table E5 in the online supplement. Based on the optimal solution computed by DCA-BL in these trials, we determine that (16) is tight at optimality when $\tau_1^* = 0.66$ and $\tau_2^* = 0.25$ with τ_p^* chosen such that

$$\sum_{j \in J} I_{p,j}^* = \tau_p^* I_p^{\max} \tag{18}$$

holds, where $I_{p,j}^*$ is the optimal value of primal variable $I_{p,j}$ for all $p \in P$ and $j \in J$.

Based on these domain tightening parameters, we increase the granularity of the grid search on τ_1 and τ_2 to observe the effect of slight perturbations of the constant C_p on DCA-BL convergence. In particular, we solve the water equilibrium problem with the 25 combinations of $\tau_1 \in \{0.66, 0.67, 0.68, 0.69\}$ and $\tau_2 \in \{0.25, 0.26, 0.27, 0.28, 0.29\}$ and record the infeasibility measure t at each iteration. The respective curves for four of these combinations are shown in Figure 9. The optimal domain tightening parameters are given by $C_p^* = \tau_p^* I_p^{\max}$ for τ_p^* defined according to (18) for $p \in \{1, 2\}$ in this stylized example. When using these parameters, DCA-BL converges in 173 iterations with a CPU time of 5.57 seconds. However, as the domain constraint is relaxed, corresponding to increasing values of τ_1 and τ_2, the number of iterations required for convergence increases as well. In each case, the solution to which DCA-BL converges matches the one computed by PATH. Overall, the application of DCA-BL to this water equilibrium problem

indicates that the general framework is capable of solving complex NCPs with interesting engineering-economic applications. However, numerical difficulties may arise if the underlying nonlinear MCP is poorly conditioned, which may be mitigated via domain tightening using problem specific knowledge.

7. Conclusion

We presented a novel difference of convex functions approach for solving linear and nonlinear complementarity problems. The approach is based on the reformulation of general MCPs as bilinear problems, which can be recast as equivalent DC programs with DC constraints. Applying standard DCA techniques, the reformulation can be solved through a sequential linear approximation of the concave terms in the DC constraints. In the case of MLCPs, the convex subproblem is a convex and quadratically-constrained problem and the approach may be thought of as some kind of sequential quadratic programming. The technique is general enough to handle nonlinear MCPs, and we present a DC reformulation of nonlinear MCPs that arise from a class of probabilistic equilibrium problems. The proposed framework is flexible and does not rely on an approximation of the underlying MCP but on a reformulation of bilinear terms as a difference of two scalar, convex, and quadratic terms, whose second quadratic function is successively approximated in a linear way. Potential inconsistencies due to the linearization are tackled via successive penalty parameter updates.

The proposed DCA-BL approach is tested extensively on a set of random LCPs. Experiments on these instances provide a number of interesting insights into the numerical properties of the approach. In particular, we find that DCA-BL converges more rapidly if the LCP matrix is more dense. We test this finding with different convex solvers to show that the result is not solver specific but an encouraging feature of the DCA-BL approach for solving LCPs that offers potential computational gains over conventional LCP solution methods. This is an interesting and surprising phenomenon that requires further exploration, yet highlights a potential advantage in the use of DCA-BL for solving large-scale LCPs. Moreover, we also vary the eigenvalue distribution of the LCP matrix and compare the performance with conventional solution techniques for complementarity problems. In particular, since the DCA-BL approach does not rely on a quadratic objective function that minimizes complementarity, the method performs particularly well against benchmark methods in the case of indefinite and asymmetric LCP matrices. Additionally, we employ DCA-BL on LCPs instances up to size $n = 50\,000$, exhibiting the utility of DCA-BL in solving large-scale LCPs. Moreover, experiments on large LCP instances highlight the linear-time update rule of DCA-BL, which leads to better performance on dense problem instances compared to other DC-based complementarity approaches such as DCA2 and DCA-QP.

In addition, we apply the approach to a class of probabilistic equilibrium problems in which each player solves a chance-constrained optimization problem, reformulated as a second-order cone program. Concatenating the first-order optimality conditions of each player gives rise to a nonlinear MCP, the solution to which represents a Nash equilibrium of the non-cooperative game. We apply DCA-BL to two such equilibrium problems in water infrastructure investment and multi-portfolio optimization. Moreover, experimentation indicates that DCA-BL is a feasible approach for solving nonlinear complementarity problems that scales linearly, and accelerated performance is seen when domain tightening techniques are employed on the primal variables. The performance of DCA-BL in solving nonlinear problems is closely tied to the conditioning of the problem, and further exploration into its numerical properties are needed to make the method competitive with modern, nonlinear MCP solvers. Nonetheless, the proposed approach offers a robust and novel framework for solving a wide class of complementarity problems that shows promising potential.

Acknowledgments

Steven A. Gabriel was supported by the National Science Foundation Grant #2113891. Steven A. Gabriel and Trine K. Boomsma were supported by the Independent Research Fund Denmark, Project #0217-00009B. Steven A. Gabriel and Dominic Flocco were supported by a grant from Petrobras #4324713. M. A. Lejeune was supported by the National Science Foundation Grant #DMS-2318519 and Grant #ECCS-2114100.

References

Ahmed, S. and Xie, W. (2018). "Relaxations and Approximations of Chance Constraints under Finite Distributions." In: *Mathematical Programming* 170.1, pp. 43–65.

An, L. T. H. and Tao, P. D. (2005). "The DC (Difference of Convex Functions) Programming and DCA Revisited with DC Models of Real World Nonconvex Optimization Problems." In: *Annals of Operations Research* 133.1, pp. 23–46.

ApS, M. (2019). *MOSEK Optimization Suite*.

Birge, J. R. and Louveaux, F. (2011). *Introduction to Stochastic Programming*. Springer Science & Business Media.

Bonami, P. and Lejeune, M. A. (2009). "An exact solution approach for portfolio optimization problems under stochastic and integer constraints." In: *Operations Research* 57.3, pp. 650–670.

Boyd, N. (2024). "Equilibrium Programming for Improved Management of Water-Resource Systems." PhD thesis. University of Maryland, College Park, Maryland USA.

Bynum, M. L., Hackebeil, G. A., Hart, W. E., Laird, C. D., Nicholson, B. L., Siirola, J. D., Watson, J.-P., and Woodruff, D. L. (2021). *Pyomo–optimization modeling in python*. Third. Vol. 67. Springer Science & Business Media.

Byong-Hun, A. (1983). "Iterative methods for linear complementarity problems with upper bounds on primary variables." In: *Mathematical Programming* 26, pp. 295–315.

Cafieri, S, D'Apuzzo, M, Marino, M., Mucherino, A, and Toraldo, G. (2006). "Interior-point solver for large-scale quadratic programming problems with bound constraints." In: *Journal of Optimization Theory and Applications* 129, pp. 55–75.

Conejo, A. J., Carrión, M., Morales, J. M., et al. (2010). *Decision making under uncertainty in electricity markets*. Vol. 1. Springer.

Constante-Flores, G, Conejo, A., and Constante-Flores, S (2022). "Solving certain complementarity problems in power markets via convex programming." In: *TOP*, pp. 1–27.

Cottle, R. W., Pang, J.-S., and Stone, R. E. (2009). *The Linear Complementarity Problem*. SIAM.

Dinh Tao, P. and Le An, T. H. (1997). "Convex Analysis Approach to D.C. Programming: Theory, Algorithms and Applications." In: *Acta Mathematica Vietnamica* 22.1, pp. 289–355.

Dirkse, S. P. and Ferris, M. C. (1995). "The PATH solver: a nommonotone stabilization scheme for mixed complementarity problems." In: *Optimization Methods and Software* 5.2, pp. 123–156.

Fathi, Y. (1979). "Computational complexity of LCPs associated with positive definite symmetric matrices." In: *Mathematical Programming* 17, pp. 335–344.

Ferris, M. C. and Munson, T. S. (1999). "Interfaces to PATH 3.0: Design, Implementation and Usage." In: *Computational Optimization: A Tribute to Olvi Mangasarian Volume I*, pp. 207–227.

Fletcher, R. and Leyffer, S. (2004). "Solving mathematical programs with equilibrium constraints as nonlinear programs." In: *Optimization Methods and Software* 19.1, pp. 15–40.

Floudas, C. A. (1995). *Nonlinear and Mixed-Integer Optimization: Fundamentals and Applications*. Oxford University Press.

Gabriel, S., Flocco, D., Boomsma, T., Schmidt, M., and Lejuene, M. (2025a). *A Heuristic for Complementarity Problems Using Difference of Convex Functions*. Available for download at https://github.com/INFORMSJoC/2024.0822.

Gabriel, S. A. (2017). "Solving discretely constrained mixed complementarity problems using a median function." In: *Optimization and Engineering* 18, pp. 631–658.

Gabriel, S. A., Conejo, A. J., Fuller, J. D., Hobbs, B. F., and Ruiz, C. (2012). *Complementarity Modeling in Energy Markets*. Vol. 180. Springer Science & Business Media.

Gabriel, S. A., Flocco, D. C., Mazzini, F. F., Sotelo, D., Castor, K. d. S., and Levorato, M. (2025b). "Theoretical results for gas market equilibrium modeling with application to Brazil." In: *Computational Management Science* 22.1, p. 2.

Gabriel, S. A., García-Bertrand, R., Sahakij, P., and Conejo, A. J. (2006). "A practical approach to approximate bilinear functions in mathematical programming problems by using Schur's decomposition and SOS type 2 variables." In: *Journal of the Operational Research Society* 57.8, pp. 995–1004.

Geiger, C. and Kanzow, C. (1996). "On the resolution of monotone complementarity problems." In: *Computational Optimization and Applications* 5, pp. 155–173.

Harker, P. T. and Pang, J.-S. (1990). "Finite-dimensional variational inequality and nonlinear complementarity problems: a survey of theory, algorithms and applications." In: *Mathematical Programming* 48.1-3, pp. 161–220.

Hart, W. E., Watson, J.-P., and Woodruff, D. L. (2011). "Pyomo: modeling and solving mathematical programs in Python." In: *Mathematical Programming Computation* 3.3, pp. 219–260.

Hoai An, L. T., Tao, P. D., Nguyen Canh, N., and Van Thoai, N. (2009). "DC programming techniques for solving a class of nonlinear bilevel programs." In: *Journal of Global Optimization* 44, pp. 313–337.

Jara-Moroni, F., Pang, J.-S., and Wächter, A. (2018). "A study of the difference-of-convex approach for solving linear programs with complementarity constraints." In: *Mathematical Programming* 169.1, pp. 221–254.

Labbé, M. and Violin, A. (2013). "Bilevel programming and price setting problems." In: *4OR* 11.1, pp. 1–30.

Lampariello, L., Neumann, C., Ricci, J. M., Sagratella, S., and Stein, O. (2021). "Equilibrium selection for multi-portfolio optimization." In: *European Journal of Operational Research* 295.1, pp. 363–373.

Le Thi, H. A. (2020). "DC programming and DCA for supply chain and production management: state-of-the-art models and methods." In: *International Journal of Production Research* 58.20, pp. 6078–6114.

Le Thi, H. A. and Dinh, T. P. (2001). "A continuous approch for globally solving linearly constrained quadratic." In: *Optimization* 50.1-2, pp. 93–120.

Le Thi, H. A., Ho, V. T., and Pham Dinh, T. (2019). "A unified DC programming framework and efficient DCA based approaches for large scale batch reinforcement learning." In: *Journal of Global Optimization* 73, pp. 279–310.

Le Thi, H. A., Huynh, V. N., and Dinh, T. P. (2014). "DC programming and DCA for general DC programs." In: *Advanced Computational Methods for Knowledge Engineering: Proceedings of the 2nd International Conference on Computer Science, Applied Mathematics and Applications (ICCSAMA 2014)*. Springer, pp. 15–35.

Le Thi, H. A., Nguyen, T. M. T., and Dinh, T. P. (2023). "On solving difference of convex functions programs with linear complementarity constraints." In: *Computational Optimization and Applications* 86.1, pp. 163–197.

Le Thi, H. A. and Pham Dinh, T. (2011). "On solving linear complementarity problems by DC programming and DCA." In: *Computational Optimization and Applications* 50.3, pp. 507–524.

– (2018). “DC programming and DCA: thirty years of developments.” In: *Mathematical Programming* 169.1, pp. 5–68.
– (2023). “Open issues and recent advances in DC programming and DCA.” In: *Journal of Global Optimization*.

Lejeune, M. and Prékopa, A. (2024). “Relaxations for Probabilistically Constrained Stochastic Programming Problems: Review and Extensions.” In: *Annals of Operations Research*. Accepted.

Murty, K. G. (1988). “Linear complementarity.” In: *Linear and Nonlinear Programming*.

Nguyen, H. N., Lisser, A., and Singh, V. V. (2024). “Random games under normal mean–variance mixture distributed independent linear joint chance constraints.” In: *Statistics & Probability Letters* 208, p. 110036.

Oliveira, W. de (2020). “The ABC of DC programming.” In: *Set-Valued and Variational Analysis* 28, pp. 679–706.

Pang, J.-S. and Gabriel, S. A. (1993). “NE/SQP: A robust algorithm for the nonlinear complementarity problem.” In: *Mathematical Programming* 60.1-3, pp. 295–337.

Pham Dinh, T., Le Thi, H. A., and Akoa, F. (Aug. 2008). “Combining DCA (DC Algorithms) and interior point techniques for large-scale nonconvex quadratic programming.” In: *Optimization Methods and Software* 23.4, 609â–629.

Prékopa, A., Ruszczyński, A., and Shapiro, A. (2003). “Probabilistic Programming Models.” In: *Handbooks in Operations Research and Management Science 10*. Elsevier, pp. 267–351.

Scheel, H. and Scholtes, S. (2000). “Mathematical Programs with Complementarity Constraints: Stationarity, Optimality, and Sensitivity.” In: *Mathematics of Operations Research* 25.1, pp. 1–22.

Siddiqui, S. and Gabriel, S. A. (2013). “An SOS1-based approach for solving MPECs with a natural gas market application.” In: *Networks and Spatial Economics* 13, pp. 205–227.

(S. A. Gabriel) (A) University of Maryland, College Park, Maryland 20742, USA, (B) Norwegian University of Science and Technology, Trondheim, Norway, (C) Aalto University, Espoo, Finland
Email address: sgabriel@umd.edu

(D. Flocco) University of Maryland, Department of Mathematics, College Park, Maryland 20742, USA
Email address: dflocco@umd.edu

(T. K. Boomsma) University of Copenhagen, Denmark
Email address: trine@math.ku.dk

(M. Schmidt) Trier University, Department of Mathematics, Universitätsring 15, 54296 Trier Germany
Email address: martin.schmidt@uni-trier.de,

(M. A. Lejeune) George Washington University, Washington, DC, USA
Email address: mlejeune@gwu.edu

Online Supplement to "A Heuristic for Complementarity Problems Using Difference of Convex Functions"

Steven A. Gabriel, Dominic Flocco, Trine K. Boomsma,
Martin Schmidt, Miguel A. Lejeune

This supplement provides additional materials to accompany the article "A Heuristic for Complementarity Problems". Appendix A contains convergence analysis and proofs left out of the main body. Appendix B provides a detailed description of the linear and nonlinear complementarity test problems used in computational experiments. Appendix C describes other DC-based complementarity methods used in benchmark tests. Appendix D details how second-order cone constrained programs can be reformulated as an equivalent DC program. Lastly, Appendices E and F present supplementary numerical results. This supplement is also accompanied by the code and data used to conduct computational experiments, which is available in the GitHub repository: (Gabriel et al. 2025a).

Date: October 7, 2025.
Key words and phrases. DC programming, Complementarity problems, Equilibrium problems, Infrastructure modeling, Portfolio selection.
Supplementary Material. Replication files, including data and code, are provided in the GitHub repository (Gabriel et al. 2025a).

Appendix A. Convergence Analysis

A.1. **Constraint Qualification.** Let f_i, $i = 1, \ldots, n$, be continuously differentiable functions and let $I(z) = \{i \in \{1, \ldots, n\} : f_i(z) = 0\}$. The Mangasarian–Fromowitz constraint qualification (MFCQ) is satisfied at z if there exists a vector d such that $\nabla f_i(z)^\top d < 0$ for all $i \in I(z)$.

In the presence of complementarity $xy = 0$, which is equivalent to constraints $xy \leq 0, x, y \geq 0$, the MFCQ requires the existence of a $d = (d_1, d_2)^\top$ such that

$$-d_1 < 0, \quad -d_2 < 0, \quad d_1 y + d_2 x < 0,$$

which is clearly impossible for $x, y \geq 0$.

A.2. **Proof of Theorem 2.** With $f_i = g_i - h_i$ representing each of the DC functions in (11), we prove a slightly more general version of Theorem 2, adapted from (Le Thi et al. 2014).

Theorem 2. *Suppose that $f_i = g_i - h_i$ and g_i, h_i are continuously differentiable and convex functions for all $i \in \{1, \ldots, n\}$. Let $\{z^k\}$ be a sequence produced by GDCA2. Then, either the algorithm stops after finitely many iterations at a solution z^k to (8) or it generates an infinite sequence of points. If there exists an iteration k_0 such that $\rho_k = \rho_{k_0}$ and $\|z^{k+1} - z^k\|_2 \leq (\|\lambda^{k+1}\|_1 + \delta_1)^{-1}$ for all $k \geq k_0$, then every limiting point z^∞ is a solution to (8).*

Proof. Suppose that GDCA2 stops in iteration k with $z^{k+1} = z^k$ and $t^{k+1} = 0$. Then, $f_i(z^{k+1}) = g_i(z^{k+1}) - h_i(z^{k+1}) = g_i(z^{k+1}) - h_i(z^k) - \nabla h_i(z^k)^\top (z^{k+1} - z^k) \leq t^{k+1} = 0$ for $i = 1, \ldots, n$, and hence, z^{k+1} is a solution to (8). Now, suppose that $\{z^k\}$ is an infinite sequence produced by GDCA2. Assume that z^∞ is a limit point of $\{z^k\}$, i.e., there exists a subsequence (w.l.o.g. indexed with k again) such that $\lim_{k\to\infty} z^k = z^\infty$.

The Karush–Kuhn–Tucker conditions of (10) are necessary for optimality, since Slater's constraint qualification is satisfied because we can always choose t being sufficiently large. Thus, (z^{k+1}, t^{k+1}) satisfies

$$0 = \sum_{i=1}^{n} \lambda_i^{k+1} (\nabla g_i(z^{k+1}) - \nabla h_i(z^k)), \tag{A.1}$$

$$\rho_k - \sum_{i=1}^{n} \lambda_i^{k+1} - \mu^{k+1} = 0, \tag{A.2}$$

$$0 \leq \lambda_i^{k+1} \perp t^{k+1} - g_i(z^{k+1}) + h_i(z^k) + \nabla h_i(z^k)^\top (z^{k+1} - z^k) \geq 0, \quad i = 1, \ldots, n, \tag{A.3}$$

$$0 \leq \mu^{k+1} \perp t^{k+1} \geq 0. \tag{A.4}$$

If there exists an iteration k_0 such that $\rho_k = \rho_{k_0}$ and $\|z^{k+1} - z^k\|_2 \leq (\|\lambda^{k+1}\|_1 + \delta_1)^{-1}$ for all $k \geq k_0$, then $\rho_k \geq r_k = \|\lambda^{k+1}\|_1 + \delta_1 > \|\lambda^{k+1}\|_1$ for all $k \geq k_0$. By (A.2), we have $\mu^{k+1} > 0$ and (A.4) implies $t^{k+1} = 0$ and $g_i(z^{k+1}) - h_i(z^k) - \nabla h_i(z^k)^\top (z^{k+1} - z^k) \leq 0$ for all $k \geq k_1$. Since g_i, h_i are continuous, we have that $\lim_{k\to\infty}(g_i(z^{k+1}) - h_i(z^k) - \nabla h_i(z^k)^\top (z^{k+1} - z^k)) = g_i(z^\infty) - h_i(z^\infty) = f_i(z^\infty) \leq 0$, i.e., z^∞ solves (8).

Appendix B. Description of MCP Test Problems

B.1. **LCP Test Problems from the Literature.** Le Thi and Pham Dinh (2011) compare the performance of DCA1 and DCA2 with Lemke's method on a set of positive semi-definite LCPs from the literature. The test problems considered in our numerical experiments are the following:

- LCP6: This example is takes from Fathi (1979) and is given by

$$M_6 = \tilde{M}_6 \tilde{M}_6^\top \quad \text{with} \quad \tilde{M}_6 = \begin{bmatrix} 1 & 2 & 2 & \cdots & 2 \\ 0 & 1 & 2 & \cdots & 2 \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & \cdots & 1 \end{bmatrix} \quad \text{and} \quad q = -e = -(1, \ldots, 1)^\top.$$

  The matrix M_6 is symmetric and positive definite. Moreover, as noted in (Fathi 1979), M_6 is a P-matrix and is fully dense.

- LCP7: This example is taken from Byong-Hun (1983) and is given by

$$M_7 = \begin{bmatrix} 4 & -2 & 0 & \cdots & 0 \\ 1 & 4 & -2 & \cdots & 0 \\ \vdots & \ddots & \ddots & & \vdots \\ 0 & 0 & \cdots & \cdots & 4 \end{bmatrix} \quad \text{and} \quad q = -e.$$

  The matrix M_7 is a non-symmetric and tridiagonal H-matrix (i.e., its comparison matrix is an M-matrix). Hence, M_7 is positive semi-definite.
- LCP8: This example is taken from Geiger and Kanzow (1996). It holds

$$M_8 = \begin{bmatrix} 4 & -1 & 0 & \cdots & 0 \\ -1 & 4 & -1 & \cdots & 0 \\ \vdots & \ddots & \ddots & & \vdots \\ 0 & 0 & \cdots & -1 & 4 \end{bmatrix} \quad \text{and} \quad q = -e.$$

  The matrix M_8 is symmetric and positive definite.
- LCP9: This example is taken from Murty (1988) and is defined by

$$M_9 = \begin{bmatrix} 2 & 2 & 2 & \cdots & 2 \\ 0 & 1 & 2 & \cdots & 2 \\ \vdots & \ddots & \ddots & & \vdots \\ 0 & 0 & \cdots & 0 & 1 \end{bmatrix} \quad \text{and} \quad q = -e.$$

  The matrix M_8 is positive semi-definite and $M_8 + M_8^\top$ has only one nonnegative eigenvalue.

B.2. **LCP Application: Market Equilibrium Problems.** Consider a set of players $P = \{1, \ldots, n_p\}$, each making their decision to maximize profit over a set of time periods $T = \{1, \ldots, n_t\}$. Accordingly, player $p \in P$ makes decisions $x^p = (x_t^p)_{t \in T}$ by solving the quadratic program with linear constraints given by (12). The necessary and sufficient KKT conditions for (12) are the following:

$$0 \leq Q^p x^p + c^p - (A^p)^\top \lambda^p - \pi \perp x^p \geq 0, \tag{B.1a}$$

$$0 \leq d^p - A^p x^p \perp \lambda^p \geq 0. \tag{B.1b}$$

In modeling the market, we consider an affine inverse demand function $\pi_t : \mathbb{R} \to \mathbb{R}$ given by

$$\pi_t(X_t) = a_t - b_t X_t, \tag{B.2}$$

with price intercept $a_t > 0$ and slope $b_t > 0$ for all $t \in T$ (Gabriel et al. 2012). In a price-taker model, the market price is formed by the market-clearing constraints:

$$0 \leq \sum_{p \in P} x_t^p - X_t \perp \pi_t \geq 0, \quad t \in T, \tag{B.3}$$

ensuring that supply equals demand at positive market prices and that prices are zero if supply exceeds demand. By concatenating the KKT conditions (B.1) for $p \in P$ and the market clearing (B.3), we arrive at a Nash equilibrium problem, expressed as an LCP of order $n = n_p(2n_t + 1) + n_t$; the complementarity system (B.1) has dimension $n_t + (n_t + 1)$ for each player plus n_t market-clearing conditions (B.3). The price-taker LCP has the form (2a), where $M \equiv M_{1,1}$ is an asymmetric and positive semi-definite matrix (Gabriel et al. 2025b). In a price-maker model, the inverse demand function (B.2) replaces the market price π in (12a) and player $p \in P$ makes decisions x^p in anticipation of its impact on prices for fixed decisions of the other players $(x^q)_{q \in P \setminus \{p\}}$:

$$\max_{x^p} \quad f_p(x^p, x^{-p}) = \pi(x^p + x^{-p})^\top x^p - \frac{1}{2}(x^p)^\top Q^p x^p + (c^p)^\top x^p \tag{B.4a}$$

$$\text{s.t.} \quad A^p x^p \leq d^p, \quad (\lambda^p) \tag{B.4b}$$

$$x^p \geq 0. \tag{B.4c}$$

The price-maker LCP is of order $n = n_p(2n_t + 1)$ (substituting for the market-clearing conditions) and likewise of the form (2a), although with a different positive semi-definite LCP matrix; see Gabriel et al. (2025b).

To analyze the performance of DCA-BL for this class of market equilibrium problems, we build random instances of the price-taker and price-maker models with varying problem sizes. Specifically, we build 10 random problem instances for the 36 combination of number of players $n_p \in \{2, 3, 5, 10, 12, 15\}$ and number of time steps $n_t \in \{2, 3, 5, 10, 12, 15\}$, leading to a total of 360 problem instances. Let $U(a, b)$ denote a uniform distribution over the open interval (a, b). For the price-taker problem, each player p's problem data is generated as follows. The matrices $Q^p \in \mathbb{R}^{n_t \times n_t}$ are generated such that they are symmetric and positive definite with eigenvalues sampled from $U(0, 1)$. The vectors $c^p \in \mathbb{R}^{n_t}$ have entries sampled from $U(0, 1)$. Furthermore, $d^p \in \mathbb{R}^{n_t+1}$ is composed of volumetric capacity $x^{p,\text{total}}$ and production capacities $x_t^{p,\text{rate}}$, where $x^{p,\text{total}} \sim U(0, n_t)$ and $x_t^{p,\text{rate}} \sim U(0, 1)$ for each $t = 1, \ldots, n_t$. The volumetric capacity for producer p bounds total production across time periods, i.e., $\sum_{t \in T} x_t^p \leq x^{p,\text{total}}$. Moreover, the production capacity of player p in time period t enforces $x_t^p \leq x_t^{x,\text{rate}}$ for all $t \in T$. The constraint matrix $A^p \in \mathbb{R}^{(n_t+1)\times(n_t+1)}$ is defined accordingly. Finally, the intercept and slope of the inverse demand curve are sampled as $a_t \sim U(0, n_p)$ and $b_t \sim (0, 0.1)$ for all $t = 1, \ldots, n_t$, respectively. Data for the price-maker problem is generated analogously, but Q^p is replaced with the diagonal matrix $D = \text{diag}(b_1, \ldots, b_{n_t})$ with entries $b_t \sim U(0, 0.1)$ for all $t \in T$. Note that the market equilibrium problem given by the LCP (2a) has a unique solution under mild conditions, which is important for planning purposes in, e.g., energy market equilibrium problems; see Gabriel et al. (2025b) for the details.

B.3. **Nonlinear LCP Application: Multi-Portfolio Optimization.**

B.3.1. *Long-Only Multi-Portfolio Optimization.* We consider an equilibrium model for multi-portfolio optimization explored by Lampariello et al. (2021) and Nguyen et al. (2024), which is motivated by the probabilistic Markowitz model by Bonami and Lejeune (2009). In this model, we have $P = \{1, \ldots, n_p\}$ firms competing in the same financial market in a non-cooperative random game. The firms invest in a set of portfolios $K = \{1, \ldots, n_k\}$, where each portfolio $k \in K$ is comprised by a set of assets A_k. Firm p must decide the percentage of its capital B_k^p to invest in each asset j of portfolio k, denoted by decision variable x_{kj}^p. For brevity, let $x_k^p = (x_{kj}^p)_{j \in A_k}$ be the investment vector of firm p for portfolio k and let $x^p = (x_k^p)_{k \in K}$ be the strategy of firm p. In this model, firms take long positions only, and so, $x^p \geq 0$. Each firm attempts to minimize their transaction cost incurred due to trades from competing firms and maximize their expected return. Given a strategy profile (x^p, x^{-p}), the quadratic market-impact cost of firm p, originally discussed by Lampariello et al. (2021) and Nguyen et al. (2024), reads

$$\text{TC}^p(x^p, x^{-p}) = \frac{1}{2} \sum_{k \in K} (B_k^p x_k^p + B_k^{-p} x_k^{-p})^\top \Omega_k^p (B_k^p x_k^p + B_k^{-p} x_k^{-p}), \tag{B.5}$$

where Ω_k^p is the symmetric and positive semi-definite market impact matrix for portfolio k and firm p, whose entry at position (i, j) is the impact of the liquidity of asset i on the liquidity of asset j. This quadratic market-impact cost connects investment decisions between players by capturing the influence of asset liquidity on the cost of trading in the market.

Now, firm p solves the chance-constrained quadratic optimization problem

$$\min_{x^p} \quad \text{TC}^p(x^p, x^{-p}) - \sum_{k \in K} B_k^p \mu^\top x_k^p \tag{B.6a}$$

$$\text{s.t.} \quad \Pr\left(\xi^\top x^p \geq R^p\right) \geq \alpha^p, \tag{B.6b}$$

$$\sum_{j \in A_k} x_{kj}^p = 1, \quad k \in K, \tag{B.6c}$$

$$x_{kj}^p \geq 0, \quad j \in A_k, k \in K, \tag{B.6d}$$

where ξ is a Gaussian random vector with mean $\mathbb{E}[\xi] = \mu$, R^p is the minimum return level for firm p and $\alpha^p \in [0.5, 1)$ is the reliability level for firm p. The objective (B.6a) of each firm is to maximize profit by minimizing their transaction cost (B.5) and maximizing their expected return. Constraint (B.6b) is the

chance constraint that ensures firm p's return exceeds its prescribed minimum R^p with probability α^p and (B.6c) says that firm p must invest all its capital. The deterministic equivalent of (B.6) is a convex-quadratic optimization problem and its KKT conditions are derived in detail in the appendices of the online supplement; see (D.3) in Appendix D of the online supplement. Concatenating the KKT conditions for each firm yields a nonlinear MCP that can be solved via DCA-BL.

B.3.2. *Long-Short Multi-Portfolio Optimization.* We also consider a variant of the above chance-constrained portfolio optimization problem in which we allow for short-sales; see, e.g., Jacobs et al. (2005), Le Thi and Moeini (2014), and Riegler-Rittner (2009). Shorting a stock is to borrow this stock from another party and is equivalent to holding a negative position. The investor can sell the acquired stock on the open market at the current price and receives the cash for the trade. A popular shorting approach is the so-called "130/30" investment strategy from Riegler-Rittner (2009) in which up to 130 % of the starting capital can be allocated to the long portfolio and no more than 30 % of the capital can be shorted.

The implementation of the long-short portfolio model is described in (Jacobs et al. 2005, 2006) for the deterministic case of a single firm. The extension requires only slight modification to firm p's quadratic optimization (B.6). Specifically, we double the number of variables by adding the nonnegative decision $\tilde{x}^p_{kj}$ to represent the firm p's short sales in asset j of portfolio k. To implement the "130/30" investment strategy in each portfolio, we add upper bound constraints

$$\sum_{j\in A_k} x^p_{kj} \le U \quad \text{and} \quad \sum_{j\in A_k} \tilde{x}^p_{kj} \le \tilde{U}, \quad k \in K, \tag{B.7}$$

which enforce nonnegative capacities U and $\tilde{U}$ on the long and short positions in portfolio k, respectively. For experimentation, we take $U = 1.3$ and $\tilde{U} = 0.3$ to implement the popular 130/30 investment strategy from Riegler-Rittner (2009). Given the Gaussian random vector ξ, the firm p's returns are given by $\xi^\top(x^p - \tilde{x}^p)$. Putting this together, firm p solves the following chance-constrained quadratic optimization problem in the long-short model:

$$\min_{x^p,\tilde{x}^p} \quad \mathrm{TC}^p(x^p + \tilde{x}^p, x^{-p} + \tilde{x}^{-p}) - \sum_{k\in K} B^p_k \mu^\top (x^p_k - \tilde{x}^p_k) \tag{B.8a}$$

$$\text{s.t.} \quad \Pr\left(\xi^\top(x^p - \tilde{x}^p) \ge R^p\right) \ge \alpha^p, \tag{B.8b}$$

$$\sum_{j\in A_k} x^p_{kj} - \tilde{x}^p_{kj} = 1, \quad k \in K, \tag{B.8c}$$

$$\text{(B.7)};\ x^p_{kj}, \tilde{x}^p_{kj} \ge 0, \quad j \in A_k, k \in K. \tag{B.8d}$$

As in the long-only case, concatenating the KKT conditions for each firm yields a nonlinear MCP that can be solved via DCA-BL. To further test the computational tractability of DCA-BL on solving nonlinear MCPs, we use the same randomly generated data as in the long-only multi-portfolio problem described in Section 6.2.1.

B.4. **Nonlinear MCP Application: Water Equilibrium Problem.** The model consists of a set P of players, which represent local municipal governments, who must choose one or more treatment technologies from a set of BMPs to be implemented. The TMDL implementation is developed to reduce the delivery of potentially harmful water-quality constituents $c \in C$ such as sediment, bacteria, or nitrogen, to stream segments in the watershed. The magnitude of a given pollutant c varies according to the land use type $\ell \in L$ and natural characteristics of the sub-watershed $w \in W$. To meet maximum allowable loadings, player p builds an infrastructure investment portfolio by deciding a series of investments $I_{p,t,w,\ell}$ to treat the land segment given by (p, w, ℓ) with technology t. A supply of credits $K_{p,c}$ is generated whenever player p reduces more of pollutant c than required to meet their allowable loading $o^{\mathrm{allw}}_{p,c}$. The other players can then purchase these credits at price π_c to reduce their allowable loading $o^{\mathrm{allw}}_{p,c}$ by the purchased amount. Uncertainty arises in pollutant loading and BMP effectiveness, which is characterized by the initial loading $\hat{o}^{\mathrm{init}}_{p,c}$ and the benefit slope $\hat{s}_{p,c,t,w,\ell}$. For what follows, we consider a single water-quality constituent and hence fix $C = \{c'\}$. For ease of notation, we denote by $J = T \times W \times L$ the set of technology, watershed, and land-use type

combinations. Letting $I_p = (I_{p,j})_{j\in J}$, we arrive at the chance-constrained problem

$$\min_{I_p, K_{p,c'}} \quad \sum_{j\in J} I_{p,j} + \pi_{c'} K_{p,c'} \tag{B.9a}$$

$$\text{s.t.} \quad \Pr\Big(\sum_{j\in J} \hat{s}_{p,c',j} I_{p,j} + K_{p,c'} \geq \hat{o}^{\text{init}}_{p,c'} - o^{\text{allw}}_{p,c'}\Big) \geq \alpha_{p,c'}, \tag{B.9b}$$

$$I_{p,j} \leq i^{\text{up}}_{p,j}, \quad j \in J, \tag{B.9c}$$

$$I_{p,j} \geq 0, \quad j \in J, \tag{B.9d}$$

$$K_{p,c'} \text{ free}, \tag{B.9e}$$

to be solved by each player p, where $i^{\text{up}}_{p,j}$ denotes the nonnegative upper bound for player p's investment in $j = (t, w, \ell)$, $\alpha_{p,c'} \in [0, 1)$ is the reliability level, and all other variables are defined as above. The objective (B.9a) of each player is to minimize the cost of investment. The chance constraint (B.9b) states that the pollutant reductions achieved from investment satisfy the minimum requirement $\hat{o}^{\text{init}}_{p,c'} - o^{\text{allw}}_{p,c'}$ with probability $\alpha_{p,c'}$, where $\hat{o}^{\text{init}}_{p,c'}$ and $\hat{s}_{p,c',j}$ are normally distributed random variables. As a result, the chance constraint (B.9b) may be reformulated as its deterministic equivalent in the general form (14), giving rise to a convex optimization problem with a second-order cone constraint. In addition to each player's optimization problem, we add the system constraint

$$\sum_{p\in P} K_{p,c'} = 0, \quad \pi_{c'} \text{ free},$$

which represents the market-clearing constraints for the stormwater credits. Concatenating the KKT conditions of each player and this system constraint yields a nonlinear MCP, which we solve using the DCA-BL framework.

Appendix C. Other DCA Approaches for Complementarity Problems

Consider the general LCP $0 \leq Mx + q \perp x \geq 0$ and let $\mathcal{P}$ denote the polyhedral, and thus convex, set that defines the feasible region of this LCP:

$$\mathcal{P} = \{x \in \mathbb{R}^n \colon Mx + q \geq 0,\, x \geq 0\}.$$

The methodology of DCA1 relies on the natural optimization model

$$\min_{x\in\mathcal{P}} \; f_1(x) = \frac{1}{2}\langle Bx, x\rangle + \langle q, x\rangle \tag{C.1}$$

for LCPs, where $B = M + M^\top$ is symmetric. Note that this is equivalent to minimizing $\langle Mx + q, x\rangle$ since

$$\frac{1}{2}\langle (M + M^\top)x, x\rangle + \langle q, x\rangle = \frac{1}{2}\left((Mx)^\top x + x^\top Mx\right) + q^\top x = (Mx)^\top x + q^\top x = (Mx + q)^\top x.$$

The algorithm relies on a DC decomposition of $f_1 = g_1 - h_1$ with

$$g_1(x) = \chi_{\mathcal{P}}(x) + \frac{1}{2}\langle(\rho_1 I_n + B)x, x\rangle + \langle q, x\rangle,$$

$$h_1(x) = \frac{1}{2}\rho_1\|x\|^2 \quad \text{and} \quad \rho_1 \geq \rho_{\min} = \max\{0, -\lambda_{\min}\}.$$

Here, $\lambda_{\min}$ denotes the smallest eigenvalue of B and $\chi_{\mathcal{P}}(\cdot)$ is an indicator function for the feasible set $\mathcal{P}$, i.e., $\chi_{\mathcal{P}}(x) = 0$ if $x \in \mathcal{P}$ and ∞ otherwise. The parameter ρ_1 is distinct from the penalty parameter ρ_k in the DCA-BL approach. Here, it is used to perturb the eigenvalues of B so that g_1 is convex over the set $\mathcal{P}$. Hence, DCA1 solves a convex quadratic program at each iteration. The update of iterates η^k, computed according to $y^k = \rho_1 x^k$, is rather simple. The DCA1 scheme is presented in Section 3.2 of (Le Thi and Pham Dinh 2011). Theorem 1 in Le Thi and Pham Dinh (2011) presents a number of useful convergence results regarding DCA1. Namely, that the sequence of iterates $\{x^k\}$ generated by DCA1 is such that $\{f_1(x^k)\}$ is decreasing and, if $f_1(x^*) = 0$ for a limit point x^*, then $x^* \in \mathcal{P}$ is a global minimizer.

The DCA scheme for LCPs that exhibits the best performance in Le Thi and Pham Dinh (2011) is DCA2, which employs a concave separable minimization. Specifically, the minimization problem (C.1) is replaced by

$$\min_{x\in\mathcal{P}} \; f_2(x) := \sum_{i=1}^{n} \min\{x_i, (Mx+q)_i\},$$

which is a concave function, so $-f_2$ is piecewise-linear and convex. Thus, the DC formulation

$$\min_{x\in\mathcal{P}} \; \chi_{\mathcal{P}}(x) - (-f_2(x))$$

us used with $g_2(x) = \chi_{\mathcal{P}}(x)$ and $h_2(x) = -f_2(x)$. Using these definitions, the subdifferential of h_2 at a point x is given by

$$\partial h_2(x) = \sum_{i=1}^{n} \partial \max\{-x_i, -(Mx+q)_i\} = -\sum_{i=1}^{n} \begin{cases} e_i, & \text{if } x_i < (Mx+q)_i, \\ M_i, & \text{if } x_i > (Mx+q)_i, \\ [e_i, M_i], & \text{if } x_i = (Mx+q)_i, \end{cases} \tag{C.2}$$

where e_i is the ith unit vector in $\mathbb{R}^n$, M_i is the ith row of M and $[e_i, M_i]$ is the line segment between e_i and M_i. Using this definition, the DCA2 scheme computes iterates η^k such that $\eta^k \in \partial h_2(x^k)$. Le Thi and Pham Dinh (2011) provide a proof of finite termination, i.e., the method stops with an iterate x^k that is a solution x^*. Moreover, the DC function f_2 has the property that if $f_2(x^*) = 0$, then x^* is a local minimizer. The DCA2 algorithm is presented in Section 3.3 of (Le Thi and Pham Dinh 2011).

Lastly, Pham Dinh et al. (2008) develop a general approach for solving large-scale nonconvex quadratic optimization problems. The DCA methodology is used to solve LCPs via the nonconvex program

$$\min_{x\in\mathcal{P}} \; f_3(x) := \frac{1}{2} x^\top (M + M^\top) x + q^\top x, \tag{C.3}$$

where $M + M^\top$ is assumed to be indefinite so that f_3 is nonconvex. Then, a proximal DC decomposition is applied to the indefinite matrix $B = M + M^\top$: $B_1 = B + \rho I$, $B_2 = \rho I$ with $\rho \geq 0$ such that $B + \rho I$ is positive semidefinite. In practice, we take $\rho = \max(-\lambda_1, 0) + 10^{-3}$, where λ_1 is the smallest eigenvalue of B. We refer to this approach as DCA-QP, given in Algorithm 4.2 of (Pham Dinh et al. 2008)

Appendix D. Chance-Constrained Optimization via DCA and Second-Order Cone Programming

Let $\xi \in \mathbb{R}^n$ denote a Gaussian random vector and let $x \in \mathbb{R}^n$ be a decision vector. We consider chance constraints of the general form

$$\Pr\left(\xi^\top x \geq c\right) \geq \alpha,$$

where $\alpha \in [0, 1]$ is a prescribed probability level and $c \in \mathbb{R}$ is known data. Consider the chance-constrained optimization problem

$$\begin{aligned} \min_{x} \quad & f(x) && && \text{(D.1a)} \\ \text{s.t.} \quad & \Pr(\xi^\top x \geq c) \geq \alpha && && \text{(D.1b)} \\ & g_i(x) \leq 0, \quad i = 1, \ldots, m, && (\gamma_i) && \text{(D.1c)} \\ & x \geq 0. && && \text{(D.1d)} \end{aligned}$$

The deterministic equivalent of (D.1) is

$$\begin{aligned} \min_{x} \quad & f(x) && && \text{(D.2a)} \\ \text{s.t.} \quad & h(x) = c - \mu^\top x - \phi^{-1}(1-\alpha)\sqrt{x^\top V x} \leq 0, && (\lambda) && \text{(D.2b)} \\ & g_i(x) \leq 0, \quad i = 1, \ldots, m, && (\gamma_i) && \text{(D.2c)} \\ & x \geq 0, && && \text{(D.2d)} \end{aligned}$$

where $V \in \mathbb{R}^{n\times n}$ is a symmetric and positive semi-definite variance-covariance matrix, $f : \mathbb{R}^n \to \mathbb{R}$ is convex and quadratic, $h : \mathbb{R}^n \to \mathbb{R}$ is convex, and $g_i : \mathbb{R}^n \to \mathbb{R}$ are affine-linear for all $i = 1, \ldots, m$. Moreover, $c \in \mathbb{R}$ and $\mu \in \mathbb{R}^n$ are given data. Hence, the KKT conditions are sufficient for optimality and necessary if a

constraint qualification holds (Floudas 1995). Constraint (D.2b) is a second-order cone constraint that is convex in x, noting that $\sqrt{x^\top Vx} = \|V^{1/2}x\|_2$. The KKT conditions read

$$0 \leq \nabla f(x) + \lambda \nabla h(x) + \sum_{i=1}^{m} \gamma_i \nabla g_i(x) \perp x \geq 0, \tag{D.3a}$$

$$0 \leq \mu^\top x + \phi^{-1}(1-\alpha)\sqrt{x^\top Vx} - c \perp \lambda \geq 0, \tag{D.3b}$$

$$0 \leq -g_i(x) \perp \gamma_i \geq 0, \quad i = 1, \ldots, m. \tag{D.3c}$$

The gradient of h is given by

$$\nabla h(x) = -\mu - \phi^{-1}(1-\alpha)\frac{Vx}{\sqrt{x^\top Vx}}.$$

Let us now define the auxiliary variable $\sigma := \sqrt{x^\top Vx} = \|V^{1/2}x\|_2$. Under the mild assumption that V is positive definite, we have $\sigma > 0$. Then, the first KKT condition (D.3a) can be written as

$$\begin{aligned}
& 0 \leq \nabla f(x) + \lambda \left(-\mu - \frac{\phi^{-1}(1-\alpha)}{\sigma} Vx \right) + \sum_{i=1}^{m} \gamma_i \nabla g_i(x) \perp x \geq 0 \\
\iff \quad & 0 \leq \sigma \nabla f(x) - \mu\lambda\sigma - \lambda\phi^{-1}(1-\alpha)Vx + \sigma \sum_{i=1}^{m} \gamma_i \nabla g_i(x) \perp x \geq 0 \\
\iff \quad & 0 \leq y = \sigma \underbrace{\left(\nabla f(x) + \sum_{i=1}^{m} \gamma_i \nabla g_i(x) \right)}_{z} - \lambda \underbrace{\left(\mu\sigma + \phi^{-1}(1-\alpha)Vx \right)}_{w} \perp x \geq 0.
\end{aligned}$$

Since f is quadratic and g_i is affine-linear for $i = 1, \ldots, m$, the terms defining the auxiliary variables z and w are linear. Hence, we can express y as the sum of bilinear terms

$$y = \sigma z - \lambda w, \tag{D.4}$$

which may be modeled using a DC approach according to Theorem 1. Furthermore, the definition of the auxiliary variable σ may be implemented as a difference of convex functions:

$$\|V^{1/2}x\|_2^2 - \sigma^2 \leq 0, \tag{D.5a}$$

$$\sigma^2 - \|V^{1/2}x\|_2^2 \leq 0. \tag{D.5b}$$

Note that the inequality (D.5a), which may be written equivalently as $\|V^{1/2}x\|_2 \leq \sigma$, is a second-order cone constraint that is convex (Boyd and Vandenberghe 2004). The term $-\|V^{1/2}x\|_2^2$ is concave, but (D.5b) may be expressed as a difference of convex functions σ^2 and $\|V^{1/2}x\|_2^2$. Using this bilinear formulation, we may solve nonlinear complementarity problems of the form (D.3) via DCA. In the equilibrium setting in which a subset of players solve a second-order cone program of the form (15), these KKT conditions may be concatenated together to find equilibrium solutions with DCA possibly with system-level constraints added as well.

Appendix E. Supplementary Numerical Results

Appendix F. Numerical Experiments with Different Choices of N

Numerical experiments indicate that the choice of N^i can have an impact on algorithm performance. Another practical choice is from (Gabriel et al. 2006), where this matrix is taken to be

$$N^i = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}, \quad i = 1, \ldots, n$$

We have tested DCA-BL with this alternative choice of N. Comparing the results with the original choice of N in Table 2 with the results for this alternative choice of N in Table E6, we see the alternative choice of N provides a slowdown for each instance. This suggests that the choice of N can have an impact DCA-BL performance.

TABLE E1. Breakdown of total time by convex subproblem solution times and update times for the benchmark LCPs with $n = 2000$. Updates for each algorithm occur in iteration k after the convex subproblem is solved. A "⋆" indicates no update was needed since the algorithm converged at this iteration. A "–" indicate an irrelevant iteration since the respective algorithm converged before this point.

ID	k	DCA1			DCA2			DCA-QP			DCA-BL		
		Solve	Update	Total	Solve	Update	Total	Solve	Update	Total	Solve	Update	Total
LCP6	0	65.159	⋆	65.159	7.420	5.600	13.020	15.782	1229.866	1245.648	19.113	0.018	19.131
	1	–	–	0.000	10.984	⋆	10.984	23.527	⋆	23.527	15.860	⋆	15.860
	Total	65.159	⋆	65.159	18.404	5.600	24.004	39.309	1229.866	1269.176	34.973	0.018	34.991
LCP7	0	2.231	0.012	2.243	0.079	4.638	4.717	0.178	2.801	2.978	0.723	0.016	0.739
	1	2.116	⋆	2.116	0.097	⋆	0.097	0.177	⋆	0.177	0.783	0.016	0.789
	2	–	–	–	–	–	–	–	–	–	0.773	0.016	0.790
	3	–	–	–	–	–	–	–	–	–	0.734	⋆	0.734
	Total	4.347	0.012	4.359	0.177	4.638	4.814	0.314	2.801	3.115	3.013	0.049	3.062
LCP8	0	0.171	⋆	0.171	0.090	⋆	0.090	0.179	2.870	3.049	0.719	0.017	0.736
	1	–	–	–	–	–	–	0.132	⋆	0.132	0.861	0.016	0.877
	2	–	–	–	–	–	–	–	–	–	0.796	⋆	0.796
	Total	0.171	0.000	0.171	0.090	0.000	0.090	0.311	2.870	3.181	2.377	0.033	2.409
LCP9	0	53.097	⋆	53.097	6.749	⋆	6.749	5.763	1437.321	1443.084	20.809	0.021	20.829
	1	–	–	–	–	–	–	4.305	⋆	4.305	42.603	⋆	42.603
	Total	53.097	0.000	53.097	6.749	0.000	6.749	10.068	1437.321	1447.389	63.412	0.021	63.433

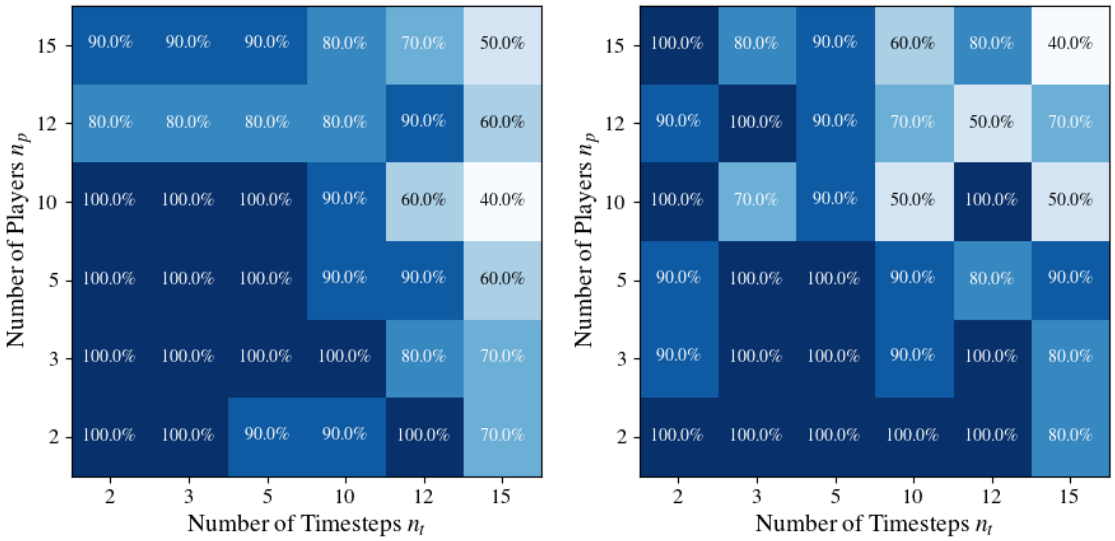

FIGURE E1. Successful solve percentages with DCA-BL algorithm. Left: price-taker. Right: price-maker. The size of the LCP is given by $n = n_p(2n_t + 1) + n_t$ for the price-taker model and $n = n_p(2n_t + 1)$ for the price-maker model.

TABLE E2. Average DCA-BL computational results for positive semi-definite LCPs with varying size and density. Runtime (RT) is reported in seconds and solution error is given by $(Mx+q)^\top x$. The successful solve rate is the percentage of instances for which DCA-BL converges to a solution with error less than 10^{-6} and in the prescribed maximum iteration limit of 500.

n	Density	PATH RT	DCA-BL RT	DCA-BL Iters.	Time/Solve	Error	Solve Rate
100	0.1	0.098	1.995	51.7	0.038	7.4e-06	96.0%
	0.2	0.144	1.735	63.2	0.024	7.8e-08	100.0%
	0.3	0.187	1.338	29.7	0.044	4.7e-09	100.0%
	0.4	0.233	1.603	34.6	0.045	4.3e-08	100.0%
	0.5	0.274	1.824	26.8	0.061	2.8e-09	96.0%
	0.6	0.318	1.442	22.8	0.057	1.6e-08	96.0%
	0.7	0.356	1.309	22.1	0.057	9.7e-11	96.0%
	0.8	0.406	1.934	28.0	0.069	1.1e-08	100.0%
	0.9	0.448	1.505	21.1	0.067	1.7e-09	100.0%
	1.0	0.490	0.934	16.8	0.054	1.1e-09	100.0%
200	0.1	0.300	3.607	44.3	0.078	5.1e-08	92.0%
	0.2	0.502	4.989	45.7	0.110	1.6e-06	76.0%
	0.3	0.713	4.900	30.3	0.163	6.8e-10	84.0%
	0.4	0.932	5.652	29.8	0.189	2.7e-08	96.0%
	0.5	1.131	5.124	28.2	0.178	9.4e-08	92.0%
	0.6	1.360	5.768	25.9	0.225	1.3e-08	96.0%
	0.7	1.584	7.713	36.7	0.193	3.5e-08	92.0%
	0.8	1.767	4.410	18.9	0.230	2.4e-09	80.0%
	0.9	2.013	6.641	21.7	0.297	3.2e-09	88.0%
	1.0	2.206	3.503	21.2	0.174	8.1e-09	92.0%
500	0.1	2.079	35.481	85.3	0.417	4.7e-07	92.0%
	0.2	4.213	50.002	67.8	0.761	5.7e-08	100.0%
	0.3	6.295	73.286	79.7	0.978	2.9e-06	92.0%
	0.4	8.322	52.343	49.9	1.055	1.2e-06	84.0%
	0.5	10.715	46.493	48.2	0.977	1.9e-08	92.0%
	0.6	12.710	34.144	41.4	0.883	3.7e-08	100.0%
	0.7	15.115	26.446	36.5	0.739	6.3e-09	92.0%
	0.8	16.777	34.907	40.9	0.845	1.6e-08	84.0%
	0.9	19.423	27.631	30.6	0.910	2.2e-08	100.0%
	1.0	21.811	20.853	26.1	0.789	2.1e-09	96.0%
1000	0.1	20.747	928.579	238.6	3.271	5.8e-07	100.0%
	0.2	41.550	430.486	128.4	3.490	5.3e-08	100.0%
	0.3	63.147	1028.610	164.7	6.140	9.9e-07	100.0%
	0.4	83.836	424.126	80.8	5.126	1.4e-06	100.0%
	0.5	104.124	1365.180	168.0	6.787	2.6e-08	100.0%
	0.6	124.469	330.971	60.4	5.371	4.2e-08	100.0%
	0.7	144.412	302.538	50.4	5.269	8.6e-09	88.9%
	0.8	164.113	224.491	49.1	4.428	1.6e-09	100.0%
	0.9	185.750	305.563	54.7	4.642	2.9e-08	100.0%
	1.0	204.846	148.523	36.9	4.031	1.2e-09	100.0%
2000	0.1	137.780	3466.751	283.2	11.453	3.7e-07	90.0%
	0.2	263.033	3148.489	221.7	13.880	8.2e-08	90.0%
	0.3	395.138	3257.166	149.9	18.165	1.9e-06	80.0%
	0.4	531.173	6993.530	254.8	23.137	2.1e-07	80.0%
	0.5	676.424	3031.509	116.2	20.230	3.5e-08	90.0%
	0.6	871.467	6174.359	150.5	28.883	5.8e-08	80.0%
	0.7	1021.085	2006.062	83.0	20.858	9.1e-09	90.0%
	0.8	1143.064	2763.111	104.8	21.318	2.9e-09	90.0%
	0.9	1258.207	4360.034	118.6	26.659	1.9e-08	100.0%
	1.0	1392.811	969.936	46.7	19.890	1.1e-09	100.0%

TABLE E3. Average DCA-BL performance on long-only multi-portfolio equilibrium problem instances with n_p firms, n_k portfolios, and n_j assets per portfolio as well as 20 problem instances per problem size. Here, $N = n_p \cdot n_k \cdot n_j$ is the total number of primal decision variables, the complementarity error is computed as $F^1(x)^\top x$, and the solve rate is the percentage of problem instances for which DCA-BL converges within the maximum iteration limit of 2500.

n_p	n_k	n_j	N	Runtime (sec.)	Iters.	Time/Solve	Error	Solve Rate
	2	3	12	0.825	95.4	0.009	3.4e-06	95.0%
	3	5	30	10.039	126.3	0.079	2.1e-05	95.0%
2	5	5	50	23.154	93.5	0.248	6.2e-06	55.0%
	5	7	70	88.981	246.6	0.361	2.1e-05	65.0%
	6	7	84	74.232	274.1	0.271	3.0e-05	90.0%
	2	3	18	3.481	237.1	0.015	8.2e-06	95.0%
	3	5	45	24.068	248.1	0.097	2.2e-05	95.0%
3	5	5	75	64.676	426.5	0.152	2.5e-05	90.0%
	5	7	105	246.885	420.9	0.587	1.8e-05	90.0%
	6	7	126	410.678	558.8	0.735	5.4e-05	90.0%
	2	3	30	35.837	587.9	0.061	3.3e-06	90.0%
	3	5	75	98.784	681.6	0.145	4.8e-06	100.0%
5	5	5	125	363.730	1050.4	0.346	6.7e-06	85.0%
	5	7	175	790.515	980.3	0.806	3.3e-06	95.0%
	6	7	210	935.858	764.8	1.224	6.0e-05	90.0%

TABLE E4. Average DCA-BL performance on long-short multi-portfolio equilibrium problem instances with n_p firms, n_k portfolios, and n_j assets per portfolio as well as 20 problem instances per problem size. Here, $N = 2n_p \cdot n_k \cdot n_j$ is the total number of primal decision variables, the complementarity error is computed as $F^1(x)^\top x$, and the solve rate is the percentage of problem instances for which DCA-BL converges within the maximum iteration limit of 2500.

n_p	n_k	n_j	N	Runtime (sec.)	Iters.	Time/Solve	Error	Solve Rate
	2	3	24	16.308	397.5	0.041	5.6e-06	90.0%
	3	5	60	27.263	407.5	0.067	1.6e-05	95.0%
2	5	5	100	98.662	411.6	0.240	2.2e-05	70.0%
	5	7	140	347.805	721.7	0.482	4.4e-05	55.0%
	6	7	168	717.696	1134.0	0.633	6.5e-05	65.0%
	2	3	36	40.086	488.3	0.082	1.0e-05	85.0%
	3	5	90	80.326	557.8	0.144	2.3e-05	85.0%
3	5	5	150	317.414	717.0	0.443	5.7e-05	70.0%
	5	7	210	767.912	1195.1	0.643	8.9e-05	60.0%
	6	7	252	1347.827	1371.2	0.983	2.3e-05	55.0%
	2	3	60	79.061	1094.6	0.072	1.8e-05	50.0%
	3	5	150	451.481	1329.2	0.340	5.2e-05	75.0%
5	5	5	250	1321.603	1536.8	0.860	2.2e-05	55.0%
	5	7	350	2497.208	2296.5	1.087	2.4e-05	55.0%
	6	7	420	3456.926	2440.0	1.417	2.4e-05	40.0%

TABLE E5. Status of the DCA-BL method for the domain tightening grid for the water equilibrium problem: infeasible (–), no convergence (+), or convergence (C). The domain tightening constant for $p \in \{1, 2\}$ is $C_p = \tau_p I_p^{\max}$, where $I_p^{\max}$ is according to (17) for the stylized network in Boyd (2024).

τ_1 \ τ_2	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
0.1	–	–	–	–	–	–	–	–	–	–
0.2	–	–	–	–	–	–	–	–	–	–
0.3	–	–	–	–	–	–	–	–	–	–
0.4	–	–	–	–	–	–	+	+	+	+
0.5	–	–	–	–	+	+	+	+	+	+
0.6	–	–	–	–	–	+	+	+	+	+
0.7	–	–	C	C	+	+	+	+	+	+
0.8	–	+	C	C	+	+	+	+	+	+
0.9	+	+	+	+	+	+	+	+	+	+
1.0	+	+	+	+	+	+	+	+	+	+

TABLE E6. Numerical results for DCA-BL with N^i, $i = 1, \ldots, n$ given from (Gabriel et al. 2006).

ID	n	Iters.	CPU Time (sec.)	Error
LCP6	1000	2	7.589	1.07E-07
	2000	2	38.662	3.98E-08
	5000	2	330.826	1.10E-08
LCP7	1000	4	1.598	7.57E-07
	2000	4	3.544	1.67E-06
	5000	4	9.076	2.71E-06
LCP8	1000	3	1.084	3.60E-05
	2000	3	2.420	9.43E-05
	5000	3	6.585	6.51E-05
LCP9	1000	2	10.780	4.67E-09
	2000	2	54.236	2.91E-08
	5000	2	542.264	1.70E-06

References

Bonami, P. and Lejeune, M. A. (2009). "An exact solution approach for portfolio optimization problems under stochastic and integer constraints." In: *Operations Research* 57.3, pp. 650–670.

Boyd, N. (2024). "Equilibrium Programming for Improved Management of Water-Resource Systems." PhD thesis. University of Maryland, College Park, Maryland USA.

Boyd, S. P. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press.

Byong-Hun, A. (1983). "Iterative methods for linear complementarity problems with upper bounds on primary variables." In: *Mathematical Programming* 26, pp. 295–315.

Fathi, Y. (1979). "Computational complexity of LCPs associated with positive definite symmetric matrices." In: *Mathematical Programming* 17, pp. 335–344.

Floudas, C. A. (1995). *Nonlinear and Mixed-Integer Optimization: Fundamentals and Applications*. Oxford University Press.

Gabriel, S., Flocco, D., Boomsma, T., Schmidt, M., and Lejuene, M. (2025a). *A Heuristic for Complementarity Problems Using Difference of Convex Functions*. Available for download at https://github.com/INFORMSJoC/2024.0822.

Gabriel, S. A., Conejo, A. J., Fuller, J. D., Hobbs, B. F., and Ruiz, C. (2012). *Complementarity Modeling in Energy Markets*. Vol. 180. Springer Science & Business Media.

Gabriel, S. A., Flocco, D. C., Mazzini, F. F., Sotelo, D., Castor, K. d. S., and Levorato, M. (2025b). "Theoretical results for gas market equilibrium modeling with application to Brazil." In: *Computational Management Science* 22.1, p. 2.

Gabriel, S. A., García-Bertrand, R., Sahakij, P., and Conejo, A. J. (2006). "A practical approach to approximate bilinear functions in mathematical programming problems by using Schur's decomposition and SOS type 2 variables." In: *Journal of the Operational Research Society* 57.8, pp. 995–1004.

Geiger, C. and Kanzow, C. (1996). "On the resolution of monotone complementarity problems." In: *Computational Optimization and Applications* 5, pp. 155–173.

Jacobs, B., Levy, K., and Markowitz, H. (2005). "Portfolio optimization with factors, scenarios, and realistic short positions." In: *Operations Research* 53.4, pp. 586–599.

– (2006). "Trimability and fast-optimization of long-short portfolios." In: *Financial Analysts Journal* 62.2, pp. 36–46.

Lampariello, L., Neumann, C., Ricci, J. M., Sagratella, S., and Stein, O. (2021). "Equilibrium selection for multi-portfolio optimization." In: *European Journal of Operational Research* 295.1, pp. 363–373.

Le Thi, H. and Moeini, A. (2014). "Long-short portfolio optimization under cardinality constraints by difference of convex functions algorithm." In: *Journal of Optimization Theory and Applications* 161, pp. 199–224.

Le Thi, H. A., Huynh, V. N., and Dinh, T. P. (2014). "DC programming and DCA for general DC programs." In: *Advanced Computational Methods for Knowledge Engineering: Proceedings of the 2nd International Conference on Computer Science, Applied Mathematics and Applications (ICCSAMA 2014)*. Springer, pp. 15–35.

Le Thi, H. A. and Pham Dinh, T. (2011). "On solving linear complementarity problems by DC programming and DCA." In: *Computational Optimization and Applications* 50.3, pp. 507–524.

Murty, K. G. (1988). "Linear complementarity." In: *Linear and Nonlinear Programming*.

Nguyen, H. N., Lisser, A., and Singh, V. V. (2024). "Random games under normal mean–variance mixture distributed independent linear joint chance constraints." In: *Statistics & Probability Letters* 208, p. 110036.

Pham Dinh, T., Le Thi, H. A., and Akoa, F. (Aug. 2008). "Combining DCA (DC Algorithms) and interior point techniques for large-scale nonconvex quadratic programming." In: *Optimization Methods and Software* 23.4, 609â–629.

Riegler-Rittner, S. (2009). *Performance of 130/30 Strategies*. Europaischer Hochschulverlag Gmbh & Co.

(S. A. Gabriel) (a) University of Maryland, College Park, Maryland 20742, USA, (b) Norwegian University of Science and Technology, Trondheim, Norway, (c) Aalto University, Espoo, Finland

Email address: sgabriel@umd.edu

(D. Flocco) University of Maryland, Department of Mathematics, College Park, Maryland 20742, USA
Email address: dflocco@umd.edu

(T. K. Boomsma) University of Copenhagen, Denmark
Email address: trine@math.ku.dk

(M. Schmidt) Trier University, Department of Mathematics, Universitätsring 15, 54296 Trier Germany
Email address: martin.schmidt@uni-trier.de,

(M. A. Lejeune) George Washington University, Washington, DC, USA
Email address: mlejeune@gwu.edu