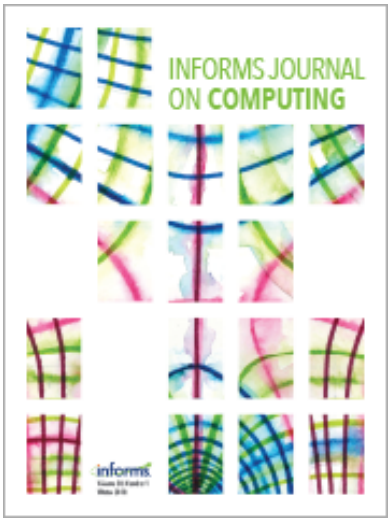


INFORMS Journal on Computing

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

A Computational Framework for Solving Nonlinear Binary Optimization Problems in Robust Causal Inference

Md Saiful Islam, Md Sarowar Morshed, Md. Noor-E-Alam

To cite this article:
Md Saiful Islam, Md Sarowar Morshed, Md. Noor-E-Alam (2022) A Computational Framework for Solving Nonlinear Binary Optimization Problems in Robust Causal Inference. INFORMS Journal on Computing

Published online in Articles in Advance 10 Aug 2022

. https://doi.org/10.1287/ijoc.2022.1226

Published in INFORMS Journal on Computing on August 10, 2022 as DOI: 10.1287/ijoc.2022.1226.
This article has not been copyedited or formatted. The final version may differ from this version.

http://pubsonline.informs.org/journal/ijoc

INFORMS JOURNAL ON COMPUTING
Articles in Advance, pp. 1–19
ISSN 1091-9856 (print), ISSN 1526-5528 (online)

A Computational Framework for Solving Nonlinear Binary Optimization Problems in Robust Causal Inference

Md Saiful Islam,[a] Md Sarowar Morshed,[a] Md. Noor-E-Alam[a,*]

[a] Mechanical and Industrial Engineering, Northeastern University, Boston, Massachusetts 02115
*Corresponding author

Contact: islam.m@northeastern.edu, https://orcid.org/0000-0002-5078-1496 (SI); morshed.m@northeastern.edu (SM); mnalam@northeastern.edu, https://orcid.org/0000-0001-5353-9710 (N-E-A)

Received: January 8, 2021
Revised: November 11, 2021; March 30, 2022; June 8, 2022; June 22, 2022
Accepted: June 29, 2022
Published Online in Articles in Advance: August 10, 2022

https://doi.org/10.1287/ijoc.2022.1226

Abstract. Identifying cause-effect relations among variables is a key step in the decision-making process. Whereas causal inference requires randomized experiments, researchers and policy makers are increasingly using observational studies to test causal hypotheses due to the wide availability of data and the infeasibility of experiments. The matching method is the most used technique to make causal inference from observational data. However, the pair assignment process in one-to-one matching creates uncertainty in the inference because of different choices made by the experimenter. Recently, discrete optimization models have been proposed to tackle such uncertainty; however, they produce 0-1 nonlinear problems and lack scalability. In this work, we investigate this emerging data science problem and develop a unique computational framework to solve the robust causal inference test instances from observational data with continuous outcomes. In the proposed framework, we first reformulate the nonlinear binary optimization problems as feasibility problems. By leveraging the structure of the feasibility formulation, we develop greedy schemes that are efficient in solving robust test problems. In many cases, the proposed algorithms achieve a globally optimal solution. We perform experiments on real-world data sets to demonstrate the effectiveness of the proposed algorithms and compare our results with the state-of-the-art solver. Our experiments show that the proposed algorithms significantly outperform the exact method in terms of computation time while achieving the same conclusion for causal tests. Both numerical experiments and complexity analysis demonstrate that the proposed algorithms ensure the scalability required for harnessing the power of big data in the decision-making process. Finally, the proposed framework not only facilitates robust decision making through big-data causal inference, but it can also be utilized in developing efficient algorithms for other nonlinear optimization problems such as quadratic assignment problems.

History: Accepted by Ram Ramesh, Area Editor for Data Science and Machine Learning.
Funding: This work was supported by the Division of Civil, Mechanical and Manufacturing Innovation of the National Science Foundation [Grant 2047094].
Supplemental Material: The online supplements are available at https://doi.org/10.1287/ijoc.2022.1226.

Keywords: causal inference • big data • discrete optimization • nonlinear optimization • observational study

1. Introduction

In this paper, we consider an emerging data science problem in robust causal inference and develop a unique computational framework consisting of a novel reformulation technique and innovative algorithms to facilitate decision making from large-scale observational data. As a natural process of digitization, we are continuously generating data on our health, behavior, mood, choices, and physical activity, which creates a fertile field of prospective (collecting data that is generated as a natural process over time) or retrospective (experimenting on already collected data) observational experiments. Such experiments on nonrandomized data are being used in identifying cause-effect relationships among variables to make informed policy decisions in public and private sectors (Nikolaev et al. 2013). Policy decisions across different domains are made by intervening in different socioeconomic variables or process parameters (treatment), and measuring their causal effect on the desired outcome. Even though randomized experiments are the gold standard for cause-effect analysis, they are often infeasible due to legal or ethical reasons. In addition, controlled experiments can be expensive and inapplicable to events that have already occurred. For instance, we might be interested in the effect of cloudy weather on bike

Published in INFORMS Journal on Computing on August 10, 2022 as DOI: 10.1287/ijoc.2022.1226.
This article has not been copyedited or formatted. The final version may differ from this version.

rentals or the effect of eating fast food on children's learning ability. Hence, in many decision-making processes, we are constrained to use observational data only, and rapid digitization is making this more prevalent.

Identifying a causal effect or testing a causal hypothesis from observational data is prone to confounding bias because of distributional differences between treated and control samples on the measured covariates (Stuart 2010). A common strategy adopted by researchers across different domains, known as the *matching method*, is to adjust for observed covariates to reduce the confounding bias. The matching method aims to restore the properties of a randomized experiment by finding a control group that is identical to the treatment group in terms of the joint distribution of the measured covariates (Stuart 2010, Nikolaev et al. 2013, Sauppe et al. 2014, Sauppe and Jacobson 2017). Due to the well-established methodological framework, matching methods have been used in many disciplines, including public health (Islam et al. 2019), economics (Dehejia and Wahba 1999), sociology (Gangl 2010), and education (Zubizarreta et al. 2014). However, forming matched pairs by assigning treated samples to control samples is a data mining problem (Morucci et al. 2018). The matched-pair construction is done by minimizing a single criterion between the treated and control samples such as multivariate distance (Rubin 1979), the probability of receiving treatment (Rosenbaum and Rubin 1983), or the χ^2 test statistic (Iacus et al. 2011, Nikolaev et al. 2013). Even so, finding pairs that balance the empirical distributions of the study subgroups is a difficult problem. In the past, this problem was solved with network flow algorithms that pursued covariate balance indirectly and relied heavily on iterative postassignment balance checking (Iacus et al. 2011). These methods involve a significant amount of guesswork, and the experimenter has little to no control over the matching process (Hill 2011). In recent years, discrete optimization models have been developed (Zubizarreta 2012, Nikolaev et al. 2013, Zubizarreta et al. 2014, Zubizarreta 2015) to solve matching problems that directly aim to minimize the imbalance metrics. Use of mixed-integer programming (MIP) models puts the experimenter in control of the matching process and provides the flexibility of including higher-order moments and multivariate moments as the imbalance measure.

Traditional matching techniques, including the recent methods using MIP models, choose a single set of matched pairs, whereas other possible sets of matched pairs may exist with equal or similar match quality. Ignoring other equally or almost equally good sets of matched pairs creates a source of uncertainty in the inference that is dependent on the choice of the experimenter over different matching methods (Morucci et al. 2018). For instance, in an attempt to evaluate the effect of the hospital readmission reduction program (HRRP) (McIlvennan et al. 2015) on nonindex readmission (readmission to a hospital that is different from the hospital that discharged the patient), Islam et al. (2019) showed that with more than 15,000 matched pairs, one experimenter can find HRRP as a cause of higher nonindex readmission, while another experimenter can find the opposite. Such uncertainty becomes more prominent for observational studies involving big data, as the chance of having multiple sets of good matches increases with the increase of the size of the data. Coker et al. (2021) refer to such uncertainty in statistical inference as the *hacking interval* in the context of linear regression and popular machine learning algorithms like the K-nearest neighbor (KNN) and support vector machine (SVM) algorithms. There are a variety of sensitivity analysis techniques available in the causal inference literature; however, they mostly focus on the sensitivity of experiment design (i.e., confounding effect of unmeasured variables) and assumptions. Some of those techniques identify the bound of allowable unmeasured confounding (see chapter 4 in Rosenbaum 2002), whereas others consider the heterogeneity of the effect (Fogarty 2020) or aim to develop nonparametric techniques (Howard and Pimentel 2021). Nonetheless, these sensitivity analysis methods do not consider the uncertainty in the poststudy design phase due to the choices made by an experimenter.

To test a causal hypothesis that is robust to the experimenter's choice of the matching algorithm, Morucci et al. (2018) proposed discrete optimization-based tests for causal hypothesis with binary and continuous outcomes. The robust causal hypothesis tests explore all possible pair assignments by computing the maximum and minimum test statistics given a good set of matches. Whereas the robust tests proposed by Morucci et al. (2018) address the uncertainty in inference, the integer programming models produce nonlinear-binary optimization problems that are difficult to solve, even for small instances.

In general, the nonlinear problem with binary variables is considered one of the challenging problems, yet, such problems abound in science and engineering applications (Murray and Ng 2010, Anthony et al. 2017). Specifically, optimization problems arising in computer vision, machine learning, and statistics are often nonlinear in nature when posed as MIP. On the other hand, Bertsimas et al. (2016) show that if the problem is computationally tractable, then the MIP approach can produce significantly better solutions than the numerical optimization techniques commonly used in data science. Unfortunately, there are very few successful solution algorithms available in the literature, and even the linearly constrained problem with quadratic objective, which is considered the simplest case among

Published in INFORMS Journal on Computing on August 10, 2022 as DOI: 10.1287/ijoc.2022.1226.
This article has not been copyedited or formatted. The final version may differ from this version.

nonlinear problems, is NP-hard (Murray and Ng 2010). In the optimization literature, there are four major approaches to solve general nonlinear integer programming problems: continuous reformulation, quadratic reformulation, linearization with piecewise linear functions, and a heuristic/metaheuristic approach. The continuous reformulation aims to transform the nonlinear binary problem into an equivalent problem of finding global optimum in continuous variables through the use of algebraic, geometric, and analytic techniques (Murray and Ng 2010) or relaxing the binary constraint and adding a penalty in the objective (Lucidi and Rinaldi 2010). However, such reformulation techniques often introduce exponentially large number of variables to the model and can make it even more difficult to solve (Murray and Ng 2010).

Similar to the continuous reformulation, quadratization adds a large number of variables in constructing an equivalent problem. For instance, quadratic reformulation of a nonlinear problem with n binary variables will need at least $2^{n/2}$ auxiliary variables (see theorem 1 in Anthony et al. 2017). Classical linearization techniques also introduce an exponentially large number of additional variables (Anthony et al. 2017) and will be impractical for large-scale problems arising from big data. However, several heuristic algorithms have been successful in solving moderately sized problems with binary variables when the objective function is quadratic (Glover et al. 2002, Boros et al. 2007). Researchers often use metaheuristics like the genetic algorithm (Gopalakrishnan and Kosanovic 2015) and neighborhood search algorithm (Archetti et al. 2020) to solve nonlinear binary optimization problems. In recent years, a variety of commercial nonlinear programming solvers has been developed; however, their applicability is limited to very small-scale instances (see Cafieri and Omheni 2017). A more related work is Islam et al. (2019), which developed scalable and efficient algorithms to solve nonlinear binary problems in robust causal hypothesis tests with binary outcome data. However, the proposed solution approach is very specific to the objective function structure and not generalizable to the problem that deals with continuous outcomes.

On the other hand, modern MIP solvers (e.g., Gurobi, Cplex, Mosek) have gained significant computational power over the last few decades due to the algorithmic development and improvement of hardware capability. For instance, we can solve many MIPs in seconds today that, 25 years ago, would have taken 71,000 years to solve (Bertsimas and Dunn 2019). Usually, MIP solvers use a combination of branch-and-bound, cutting plane, and group-theoretic approaches to solve practical problems (Bixby 2012). Instead of employing specific branching rules, MIP solvers often use a hybrid branching strategy by combining rules like *nonchimerical branching* (Fischetti and Monaci 2012), *reliability branching*, and *inference branching* (Martin et al. 2005). Cutting planes are a pivotal tool for the solvers, especially when solving integer linear programs (ILPs). Quadratic integer programs (QIPs) also take advantage of the branch-and-bound algorithm, where, at each node, a quadratic program is solved using efficient techniques like barrier methods. In addition to these techniques, MIP solvers often use efficient preprocessing methods to tighten the original formulation and reduce the size of the problem (Achterberg and Wunderling 2013). There is a wide range of starting and improvement heuristics available in the solvers that provide a good incumbent solution or a sufficient one when the problem is intractable (see the survey by Fischetti and Lodi 2010).

Apart from the algorithmic developments, modern MIP solvers take advantage of hardware technology by running multiple optimizers concurrently on multiple threads and choosing the best one, as we often do not know which algorithm, branching rule, or their combination would be most efficient for the problem at hand. To further increase the efficiency of MIP solvers, researchers are exploring different machine learning (Alvarez et al. 2014, He et al. 2014) and reinforcement learning (Etheve et al. 2020) techniques for better branching and predicting the subtree size at a node. However, these MIP solvers cannot solve a general nonlinear problem. To leverage the improvements of the MIP solvers, we need a linear or at least a quadratic formulation of the current nonlinear problems in robust causal inference.

In this paper, we consider the nonlinear binary optimization models of robust causal inference tests proposed by Morucci et al. (2018) for continuous outcome data and develop a computational framework that can handle large-scale observational data. First, we propose a unique approach to reformulate the nonlinear binary optimization problems as equivalent and less-restrictive feasibility problems. By exploiting the structure of the feasibility problem, we then develop greedy algorithms to solve the original robust causal hypothesis test problems. The reformulation into a feasibility problem also provides an opportunity to formulate the robust test problems as quadratic integer programming problems that can take advantage of recent developments in commercial MIP solvers. Without this reformulation, we cannot use MIP solvers to get guaranteed optimality. A major advantage of the reformulation approach is that, unlike state-of-the-art quadratization techniques, here, we do not need to add any additional variables. Moreover, the proposed unique reformulation approach can be leveraged to model general nonlinear and quadratic optimization problems (e.g., the quadratic assignment problem) as feasibility problems

Published in INFORMS Journal on Computing on August 10, 2022 as DOI: 10.1287/ijoc.2022.1226.
This article has not been copyedited or formatted. The final version may differ from this version.

and to develop efficient algorithms. Nonetheless, the proposed algorithms are very efficient in solving practical-sized problems and are scalable to very-large-scale problems. Our numerical experiments on real-world data sets show that the proposed computational framework provides the same inference to robust tests while taking a fraction of the time required by the exact method. The time complexity demonstrates that the developed algorithms are scalable enough to harness the power of big data in performing robust causal analysis, which consequently will provide robust policy decisions.

The remainder of the paper is organized as follows. We discuss the matching method and the robust causal hypothesis test with continuous outcomes (i.e., robust Z-test) along with its challenges in Section 2. We provide a feasibility formulation of the robust Z-test in Section 3. In Section 4, we develop greedy algorithms to solve the reformulated Z-test and analyze the properties and complexity of the proposed algorithms. We apply our algorithms to three real-world data sets of varying sizes in Section 5 and compare our result with Gurobi and the ILP-based heuristic proposed by Morucci et al. (2018). Finally, we provide concluding remarks in Section 6.

2. Matching Method and Robust Test

In this section, we discuss the matching method in general and the robust causal hypothesis test with continuous outcomes. We first discuss the matching method and introduce necessary notations and required assumptions to make causal inference from observational data. We then discuss the integer programming model for a robust causal hypothesis test with continuous outcomes, existing methods to solve the robust test, and issues with the current solution method.

2.1. Matching Method

In this paper, we consider the potential outcome framework (Holland 1986) that has led to the development of matching methods. Under the potential outcome framework, treatment effect or causal effect is measured by comparing the counterfactual: the difference in the potential outcomes of a sample unit in both the treated and control scenarios. The fundamental problem in causal inference is that we can only observe one scenario (either treated or control) for each sample. A sample $i \in \mathcal{S}$, where $\mathcal{S} := \{1, 2, \ldots, N\}$ can only have outcome $Y_i = Y_i^1 T_i + Y_i^0(1 - T_i)$, where $T_i \in \{0, 1\}$ (Holland 1986). Here, Y_i^1 represents the outcome under treatment (i.e., $T_i = 1$), and Y_i^0 represents outcome without treatment (i.e., $T_i = 0$). This problem is overcome by considering the average of the treatment effect $\tau = Y^1 - Y^0$ over the study population $\mathcal{S}$. Whereas this strategy works for randomized experiments, experiments with observational data produce biased inference due to the distributional differences of the groups that received treatment (i.e., $\mathcal{T} \subset \mathcal{S}$) and the group which did not (i.e., $\mathcal{C} \subset \mathcal{S}$) on some pretreatment covariates.

The matching method provides an unbiased estimation of the treatment effect by identifying pairs (t, c), where $t \in \mathcal{T}$ and $c \in \mathcal{C}$ or subsets $\mathcal{T} \subset \mathcal{T}$ and $\mathcal{C} \subset \mathcal{C}$ are matched exactly in terms of their covariate set $\mathbf{X} \in \mathcal{X}$ (Rosenbaum and Rubin 1983, Stuart 2010). However, it is almost impossible to find exact matches, even with a small number of covariates (Rosenbaum and Rubin 1983, Zubizarreta 2012). A large number of matching methods have been developed to make (t, c) pairs or $(\mathcal{T}, \mathcal{C})$ subsets as similar as possible in terms of $\mathbf{X}$ (Stuart 2010, Nikolaev et al. 2013). In this paper, we will restrict the scope to one-to-one matching and consider the matching algorithms that identify (t, c) pairs. Some commonly used matching methods are propensity score matching (Rosenbaum and Rubin 1983), nearest neighbor matching (Stuart 2010), optimal matching (Rosenbaum 1989), Mahalanobis distance matching (Rubin 1979), and genetic matching (Diamond and Sekhon 2013). One of the popular methods (if not the most popular) (Stuart 2010, Zubizarreta 2012) is the propensity score matching method (Rosenbaum and Rubin 1983) that employs a logistic model to estimate each sample's propensity of receiving treatment and find the (t, c) pairs that are minimizing the differences in their propensity scores. The matching process is repeated and evaluated iteratively until the desired quality of the matches is achieved. Once a suitable set of matches is identified, we can test the null hypothesis in (1) and (2) to make causal inference. Note that $\mathcal{H}_0^{SATE}$ in (1) tests the zero sample average treatment effect (SATE) hypothesis on the whole sample set, and $\mathcal{H}_0^{SATT}$ in (2) tests the zero sample average treatment effect hypothesis on the treated (SATT) samples. Here, we make the assumptions that are commonly used in causal inference literature (see Appendix 1 of Online Supplement S1):

$$\mathcal{H}_0^{SATE} := \mathbb{E}_{Y|X}[Y^1 - Y^0 | \mathbf{X}] = 0, \tag{1}$$

$$\mathcal{H}_0^{SATT} := \mathbb{E}_{Y|X}[Y^1 - Y^0 | \mathbf{X}, T = 1] = 0 \tag{2}$$

2.2. Uncertainty Due to the Choice of the Experimenter

As we mentioned in Section 2.1, matching methods aim to minimize a specific set of criteria to find the matched pairs (t, c) in one-to-one matching. In the causal inference literature, there are plentiful algorithms to find such matched-pair sets; however, they offer little to

Published in INFORMS Journal on Computing on August 10, 2022 as DOI: 10.1287/ijoc.2022.1226.
This article has not been copyedited or formatted. The final version may differ from this version.

no clear guidance on the choices of matching procedure (Morgan and Winship 2015). A common practice among researchers and practitioners is to use widely adopted and cited techniques and software (Morucci et al. 2018). Therefore, starting with the same data set and the same matching objectives, two experimenters can get two different sets of matched pairs. As the pair assignment process does not consider outcomes, traditional inference techniques do not guarantee the same inference by both experimenters. For instance, treated unit $t \in \mathcal{T}$ can have multiple potential assignments $\{c_1, c_2, \ldots, c_n\} \in \mathcal{C}$ with equal match quality but different outcomes. Each possible assignment will potentially have a different treatment effect: $Y_t^1 - Y_{c_1}^0 \neq Y_t^1 - Y_{c_2}^0 \neq \cdots \neq Y_t^1 - Y_{c_n}^0$, which creates an uncertainty in the inference. Moreover, this phenomenon is more prominent in big data observational studies due to the availability of more pairing options.

2.3. Robust Approach for Causal Hypothesis Tests

To make a causal inference that is robust to the choice of the experimenter, recently, Morucci et al. (2018) proposed a new methodology based on discrete optimization techniques. Before discussing its difference with the classical method of making causal inference (Rosenbaum and Rubin 1985, Holland 1986), we need to define the pair assignment variables $a_{i,j}$ and the set of good matches $\mathcal{M}$.

Definition 1 (A Set of Good Matches). The set of good matches $\mathcal{M}$ includes treated samples $\mathcal{T} \subset \mathcal{S}$ and control samples $\mathcal{C} \subset \mathcal{S}$ that satisfy user-defined matching criteria such that $\mathcal{M} := \{(t_i, c_j) \in (\mathcal{T} \times \mathcal{C}) : t_i \simeq c_j | \mathbf{X}\}$.

The set of good matches $\mathcal{M}$ can be stored in a logical matrix $D^{|\mathcal{T}|\times|\mathcal{C}|}$. Here, an element in D, $d_{i,j} = 1$ if treated unit i is a good match to a control unit j, and 0 otherwise.

Definition 2 (Pair Assignment Operator). We have that $a_{ij} \in \{0, 1\}$ is the pair assignment operator, where $a_{ij} = 1$ if treated unit $t_i \in \mathcal{T}$ is paired with a control unit $c_j \in \mathcal{C}$ and the pair $(t_i, c_j) \in \mathcal{M}$; $a_{ij} = 0$ otherwise.

Here, we are interested in testing the causal hypothesis (1) or (2) given a good set of matches $\mathcal{M}$. The classical approach selects one matched-pair set from $\mathcal{M}$, calculates the test statistic Λ, and makes inference based on Λ or the corresponding p-value. On the other hand, the robust approach explores all possible pair assignments within $\mathcal{M}$ without replacement by calculating the maximum and minimum test statistics $(\Lambda_{\max}, \Lambda_{\min})$. Using $(\Lambda_{\max}, \Lambda_{\min})$, a robust test can be defined as the following.

Definition 3 (Robust Test). Given a level of significance α to test the hypothesis $\mathcal{H}_0$ and $(\Lambda_{\max}, \Lambda_{\min})$ calculated from $\mathcal{M}$, the test is called *α-robust* if $|p\text{-value}(\Lambda_{\max}) - p\text{-value}(\Lambda_{\min})| \leq \alpha$. The test $\mathcal{H}_0$ is called *absolute-robust* when $p\text{-value}(\Lambda_{\min}) = p\text{-value}(\Lambda_{\max})$.

However, computing the test statistics requires solving binary optimization problems, and their structure depends on the nature of the test statistics. For example, McNemar's test (McNemar 1947) and the Z-test (Low et al. 2016) proposed by Morucci et al. (2018) for binary and continuous data, respectively, produce nonlinear binary optimization problems. Whereas they ensure robustness in inference, nonlinear-binary optimization problems are extremely difficult to solve, even for smaller instances. Islam et al. (2019) developed efficient algorithms for McNemar's test with binary outcomes by converting the original nonlinear problem into a counting problem. The robust tests with the continuous outcome, on the other hand, remain difficult to solve for practical-sized problems. In the following section, we discuss the robust Z-test model, the current solution approach, and challenges with the current approach.

2.4. Robust Z-Test and Challenges

To test the zero causal effect hypothesis with continuous outcomes, one can consider the canonical Z-test (Morucci et al. 2018) with the test statistics $\Lambda := Z(\mathbf{a}) = \frac{\sqrt{n}(\hat{\tau}-0)}{\hat{\sigma}}$. Here, n is the number of matched pairs, $\hat{\tau} = \frac{1}{n}\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t - y_j^c)a_{i,j}$ is the average treatment effect among the matched pairs, and $\hat{\sigma}$ is the sample standard deviation of the treatment effect. Given a set of good matches $\mathcal{M}$, an integer programming formulation of the Z-test proposed by Morucci et al. (2018) is provided in (3)–(8):

$$\max/\min \; Z(\mathbf{a}) = \frac{\frac{1}{\sqrt{n}}\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t - y_j^c)a_{i,j}}{\sqrt{\frac{1}{n}\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}[(y_i^t - y_j^c)a_{i,j}]^2 - \left(\frac{1}{n}\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t - y_j^c)a_{i,j}\right)^2}} \tag{3}$$

$$\text{subject to:} \quad \sum_{i\in\mathcal{T}} a_{ij} \leq 1 \quad \forall j, \tag{4}$$

$$\sum_{j\in\mathcal{C}} a_{ij} \leq 1 \quad \forall i, \tag{5}$$

$$a_{i,j} \leq d_{i,j} \quad \forall i, j, \tag{6}$$

$$\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}} a_{ij} = n, \tag{7}$$

$$a_{ij} \in \{0, 1\} \quad \forall i, j. \tag{8}$$

Here, the objective function $Z(\mathbf{a})$ represents the test statistic Λ, and the denominator in the objective function (Equation (3)) is the sample standard deviation $(\hat{\sigma})$ of the treatment effect. As we can see, to compute the test statistic, we have to solve the above nonlinear binary optimization problems. The current solution approach proposed in Morucci et al. (2018) linearizes

Published in INFORMS Journal on Computing on August 10, 2022 as DOI: 10.1287/ijoc.2022.1226.
This article has not been copyedited or formatted. The final version may differ from this version.

the above model by imposing an upper bound on $\hat{o}$, $\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}[(y_i^t - y_j^c)a_{i,j}]^2 \le b_l$, and by replacing the objective with max/min $\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t - y_j^c)a_{i,j}$. Therefore, the linearized Z-test model takes the following form:

$$\max/\min \quad \sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t - y_j^c)a_{i,j} \tag{9}$$

$$\text{subject to}: \quad \sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}[(y_i^t - y_j^c)a_{i,j}]^2 \le b_l \quad \text{and}$$
$$\text{Constraints}(4)-(8) \tag{10}$$

The integer linear program (ILP)–based heuristic of Morucci et al. (2018) solves the model (9)–(10) iteratively by changing the upper bound of standard deviation b_l on a grid. Starting with a coarse grid of b_l, the proposed algorithm solves a series of ILPs and creates a new, refined mesh at each iteration. The iterative process continues on the grid of b_l until the desired level of tolerance on the upper bound b_l is achieved. Whereas this innovative approach provides a working solution to a complex problem, it faces several challenges in practice. The first challenge is that, at each grid point, we need to solve an ILP, and, after refining the grid of b_l, we have to solve the ILP with updated constraints. As we have to iterate over the grid of b_l many times, the ILP has to be solved hundreds of times if not thousands. One can solve small ILP instances with commercial MIP solvers efficiently; however, in today's big data world, such smaller problems are highly unlikely. In addition, solving hundreds of ILPs adds a significant computational burden. The second challenge is that the range of b_l can be very large, which significantly increases the number of ILPs that we have to solve to calculate the test statistics. For instance, in a case study with Bikeshare data (Fanaee-T and Gama 2014), the range of b_l is 1.12 million to 26.12 million.

To overcome these challenges, in this paper, we reformulate the nonlinear-binary formulation of the Z-test into a less-restricted feasibility problem. By leveraging the structure of the feasibility problem, we develop greedy algorithms that are very efficient and scalable to big data observational experiments. The feasibility formulation also allows us to convert the Z-test problems into a quadratic integer program to take advantage of recent developments in MIP solvers.

3. Reformulation of the Z-Test

The optimization model discussed for the robust Z-test in (3)–(8) has a fractional objective in the form of $Z(\mathbf{a}) = \frac{f(x)}{g(x)}$. Both $f(x)$ and $g(x)$ can be written as a function of treatment effect between treated sample i and control sample j: $(y_i^t - y_j^c)a_{i,j}$. In addition, we can bound $Z(\mathbf{a})$ in the range of $Z(\mathbf{a}) \le |4|$, as, beyond this range, we will have approximately zero area under the standard normal distribution curve, and optimizing further beyond this range will not make any difference in the inference. For instance, let's assume that we find a suboptimal solution for the maximization problem, a set of treated-control pair assignments for which $Z(\mathbf{a}) = 4.10$. In theory, we can find a global optimal solution better than the current solution; however, improving the solution quality further will not change the robust inference of the hypothesis test. Using this property of the optimization model, we reformulate the robust Z-test problem as a feasibility problem.

To reformulate the robust Z-test as a feasibility problem, let's assume that, for the minimization model, $\exists\, a_{i,j}$ such that $Z(\mathbf{a}) \le \gamma$, where γ is a scalar parameter. We can iterate over different values of γ to find its optimal value. Then we have the following:

$$\frac{1}{\sqrt{n}}\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t - y_j^c)a_{i,j} \le \gamma \times \sqrt{\frac{1}{n}\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}[(y_i^t - y_j^c)a_{i,j}]^2 - \left(\frac{1}{n}\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t - y_j^c)a_{i,j}\right)^2}, \tag{11}$$

$$\frac{1}{n}\left(\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t - y_j^c)a_{i,j}\right)^2 \le \gamma^2\frac{1}{n}\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}[(y_i^t - y_j^c)a_{i,j}]^2 - \gamma^2\left(\frac{1}{n}\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t - y_j^c)a_{i,j}\right)^2. \tag{12}$$

It is important to note that, in Inequality (12), we are taking the square of an inequality for further simplification. Both sides of Inequality (11) can be negative and nonnegative depending on the intervals of sum of treatment effects ($\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t - y_j^c)a_{i,j}$) and γ. Therefore, to ensure the correct direction of the inequality, we must consider four possible combinations of the sum of treatment effects and γ. We consider these combinations in four cases and discuss them in the following subsection. Before we discuss the four cases, let us introduce the proposed feasibility formulation by assuming that Inequality (12) is a valid inequality. Simplifying Inequality (12) further, we get the following quadratic constraint:

$$\left(1 + \gamma^2\frac{1}{n}\right)\left(\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t - y_j^c)a_{i,j}\right)^2 - \gamma^2\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}[(y_i^t - y_j^c)a_{i,j}]^2 \le 0. \tag{13}$$

Therefore, the nonlinear-binary optimization problem can be framed as the following binary feasibility problem: $\exists$ a set of assignments $a_{i,j}$ so that $Z(\mathbf{a}) \le \gamma$ with

Published in INFORMS Journal on Computing on August 10, 2022 as DOI: 10.1287/ijoc.2022.1226.
This article has not been copyedited or formatted. The final version may differ from this version.

Constraints (14)–(20):

$$\left(1+\gamma^2\frac{1}{n}\right)\left(\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t-y_j^c)a_{i,j}\right)^2-\gamma^2\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}[(y_i^t-y_j^c)a_{i,j}]^2\le 0, \tag{14}$$

$$\sum_{i\in\mathcal{T}}a_{ij}\le 1 \quad \forall j, \tag{15}$$

$$\sum_{j\in\mathcal{C}}a_{ij}\le 1 \quad \forall i, \tag{16}$$

$$a_{i,j}\le d_{i,j} \quad \forall i,j, \tag{17}$$

$$\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}a_{ij}=n, \tag{18}$$

Additional constraints to validate Inequality (14) (19)

$$a_{ij}\in\{0,1\} \quad \forall i,j. \tag{20}$$

3.1. Advantage of the Proposed Feasibility Formulation

The reformulation (14)–(20) converts a nonlinear optimization problem into an equivalent and less-restrictive feasibility problem that offers several advantages.

First, it presents an opportunity to formulate the robust Z-test problem as a quadratic integer program (QIP) (see Section 4). Whereas general nonlinear integer optimization problems are difficult to solve, QIP formulation can benefit from the recent developments in MIP solvers.

Second, feasibility formulation facilitates new algorithmic development. For example, we can show that the reformulated problem (14)–(20) is a convex-feasibility problem with binary variables. Solving the feasibility problem for any γ and finding an optimal γ in the range of $-4\le\gamma\le 4$ with a binary search algorithm can solve the robust Z-test problem. Note that the range of γ is fixed for all data sets and is very small compared with b_l used in the ILP-based approach, as beyond $Z(\mathbf{a})=|4|$ will have approximately zero area under the standard normal distribution curve. Similar to the Z-test, most parametric hypothesis tests (i.e., Student's *t*-test, Welch's *t*-test, *F*-test, χ^2-test) use test statistics in the $\frac{f(x)}{g(x)}$ form, where both $f(x)$ and $g(x)$ are functions of data, and the test statistics follow certain distributions. If those tests are conducted in the spirit of the *hacking interval* proposed by Coker et al. (2021), then, regardless of the application domain, we can use the bounded nature of the distribution of test statistics and its structure to create a feasibility formulation. The proposed feasibility reformulation scheme will provide guidance to develop scalable solution techniques for other robust hypothesis tests. With regard to solving the feasibility problem, convex feasibility problems with continuous variables have been studied extensively in the optimization literature (Escalante and Raydan 2011), and iterative projection-based algorithms (Zhao et al. 2018, De Bernardi et al. 2019, Necoara et al. 2019, Li et al. 2019, Morshed et al. 2021) have proved to be very efficient in solving such problems. Unfortunately, to our best knowledge, there is no efficient algorithm available to solve convex feasibility problems with binary variables. Recently, projection-based algorithms have been developed (Chubanov 2012, 2015; Basu et al. 2014) to solve linear feasibility problems with binary variables. We believe that new algorithms can be developed for solving convex feasibility problems with integer variables. Apart from the causal hypothesis test problem, other types of optimization problems such as the quadratic assignment problem (QAP) (Pitsoulis and Pardalos 2009) can be formulated as a feasibility problem and can be solved with iterative projection-based algorithms.

Finally, this unique reformulation simplifies the structure of the problem. By leveraging the structure, we can develop algorithms to solve such a computationally expensive problem. In that vein, we develop several greedy schemes to solve the robust Z-test problems that are efficient and scalable to harness the power of big data in causal analysis.

3.2. Cases for the Minimization Problem

To simplify the test statistic $Z(\mathbf{a})$ and formulate it as a feasibility problem, we took the square of an inequality in (12). To ensure the validity of this inequality, we need to add additional constraints that lead to four possible cases. The resulting cases along with the constraints are presented below. It is important to note that $\hat{\sigma}=\sqrt{\frac{1}{n}\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}[(y_i^t-y_j^c)a_{i,j}]^2-\left(\frac{1}{n}\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t-y_j^c)a_{i,j}\right)^2}>0$, therefore, will not influence the cases.

Case 1. For this case, we consider $\gamma\ge 0$ and $\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t-y_j^c)a_{i,j}\ge 0$. Now, Inequality (11) can be written as $\frac{1}{\sqrt{n}}\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t-y_j^c)a_{i,j}\le\gamma\hat{\sigma}$. Here, $\gamma\ge 0,\hat{\sigma}>0$ and $\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(T_i-C_j)a_{i,j}\ge 0$. So, both sides of the inequality are positive. Taking the square will not change the sign of the inequality. Hence, we will have the following constraints:

$$\left(1+\gamma^2\frac{1}{n}\right)\left(\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t-y_j^c)a_{i,j}\right)^2-\gamma^2\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}[(y_i^t-y_j^c)a_{i,j}]^2\le 0, \tag{21}$$

$$\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t-y_j^c)a_{i,j}\ge 0. \tag{22}$$

Case 2. For this case, we consider $\gamma\le 0$ and $\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t-y_j^c)a_{i,j}\le 0$. As both sides of Inequality (11) are negative, the sign of the quadratic constraint will

Published in INFORMS Journal on Computing on August 10, 2022 as DOI: 10.1287/ijoc.2022.1226.
This article has not been copyedited or formatted. The final version may differ from this version.

change, and we will have the following constraints:

$$\left(1+\gamma^2\frac{1}{n}\right)\left(\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t-y_j^c)a_{i,j}\right)^2-\gamma^2\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}[(y_i^t-y_j^c)a_{i,j}]^2\geq 0, \tag{23}$$

$$\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t-y_j^c)a_{i,j}\leq 0. \tag{24}$$

Case 3. For this case, we consider $\gamma\geq 0$ and $\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t-y_j^c)a_{i,j}\leq 0$. As $\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t-y_j^c)a_{i,j}\leq 0$, the left-hand side of Inequality (11) is nonpositive, but $\gamma\geq 0$ and $\hat{\sigma}>0$, which makes the right-hand side nonnegative. Since this is true for all $\gamma\in\mathbb{R}$, $\gamma\geq 0$, we have that

$$\frac{1}{\sqrt{n}}\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t-y_j^c)a_{i,j}\leq\min_{\gamma\in\mathbb{R}}(\gamma\hat{\sigma})=0. \tag{25}$$

Therefore, we will have the following constraints. For this case, our nonlinear feasibility problem becomes a linear feasibility problem:

$$\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t-y_j^c)a_{i,j}\leq 0. \tag{26}$$

Case 4. For this case, we consider $\gamma\leq 0$ and $\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t-y_j^c)a_{i,j}\geq 0$. As the left-hand side of Inequality (11) is nonnegative and the right-hand side is nonpositive, the above equation only holds at equality: $\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t-y_j^c)a_{i,j}=0$. However, this specific case is redundant, as it is already considered in the other cases. All of the above-mentioned cases and resulting constraints are summarized in Table 1.

Each of the above cases has two conditions, one on the sum of treatment effect ($\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t-y_j^c)a_{i,j}$) and another on γ. The condition of the sum of treatment effect $\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t-y_j^c)a_{i,j}\geq 0$ (or ≤ 0) depends on the assignment variable; therefore, this condition is included in the optimization model as a constraint in (19). For Case 1 in the minimization problem, this constraint takes the form of Inequality (22). Moreover, γ is not dependent on the assignment variable, so we do not include it in the optimization model. The condition on γ is considered after solving the assignment problems, as discussed in the following section.

The maximization model follows the same argument. For completeness, we provide the corresponding cases of the maximization problem in Appendix 2 of Online Supplement S1.

4. Algorithmic Approach

In this section, we develop greedy algorithms to solve the derived cases of the robust Z-test by exploiting the structure of the feasibility problem formulated in the previous section. First, we consider Case 1 of the minimization problem. In Case 1, apart from the assignment constraints, we have to satisfy Constraints (21) and (22) to calculate γ^* in the range $\gamma\geq 0$. Simplifying Constraint (21) will result in the following:

$$\left(\frac{n\gamma^2}{n+\gamma^2}\right)\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}\left[(y_i^t-y_j^c)a_{i,j}\right]^2\geq\left[\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t-y_j^c)a_{i,j}\right]^2. \tag{27}$$

As our objective is to find a set of assignments $a_{i,j}$ that produces the smallest value of γ while satisfying Constraints (27) and (22), we can exploit the structure of Inequality (27). Note that in Inequality (27), for any number of samples (n), the minimum possible value of γ is possible when $\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}[(y_i^t-y_j^c)a_{i,j}]^2$ is maximized and $[\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t-y_j^c)a_{i,j}]^2$ is minimized, which can be written as the following optimization problem:

$$\max_{a_{i,j}\in\mathcal{M}}\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}\left[(y_i^t-y_j^c)a_{i,j}\right]^2-\left[\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t-y_j^c)a_{i,j}\right]^2$$
$$\text{subject to : Constraints (4)–(8), and (22)} \tag{28}$$

The above quadratic integer program (QIP) (28) can be solved with any commercial MIP solver. Using the solution of (28) and a prespecified n, we can calculate the optimal solution γ^* by solving the quadratic Equation (29) for γ. As the condition for Case 1 of the minimization problem is $\gamma\geq 0$, γ^* would be the minimum nonnegative solution of γ from Equation (29). Similarly, we can develop QIPs for Cases 1 and 2 of both the minimization and maximization problems (see Table 2). The details on QIP formulation are provided in Appendix 3 of Online Supplement S1. Case 3 for

Table 1. Cases and Resulting Constraints for the Minimization Problem Feasibility Formulation of the Robust Z-Test

Case	Case constraints	Quadratic constraint
1	$\gamma\geq 0$, $\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t-y_j^c)a_{i,j}\geq 0$	$\left(1+\gamma^2\frac{1}{n}\right)(\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t-y_j^c)a_{i,j})^2-\gamma^2\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}[(y_i^t-y_j^c)a_{i,j}]^2\leq 0$
2	$\gamma\leq 0$, $\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t-y_j^c)a_{i,j}\leq 0$	$\left(1+\gamma^2\frac{1}{n}\right)(\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t-y_j^c)a_{i,j})^2-\gamma^2\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}[(y_i^t-y_j^c)a_{i,j}]^2\geq 0$
3	$\gamma\geq 0$, $\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t-y_j^c)a_{i,j}\leq 0$	No quadratic constraint
4	$\gamma\leq 0$, $\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t-y_j^c)a_{i,j}\geq 0$	No quadratic constraint (redundant, only true when both inequalities are zero)

Published in INFORMS Journal on Computing on August 10, 2022 as DOI: 10.1287/ijoc.2022.1226.
This article has not been copyedited or formatted. The final version may differ from this version.

Table 2. Quadratic Integer Programs (QIPs) for the Robust Z-Test Cases

Case	Case Constraints	QIP
	Minimization problems	
1	$\gamma \geq 0,\ \sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t - y_j^c)a_{i,j} \geq 0$	$\max_{a_{i,j}\in\mathcal{M}} \sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}[(y_i^t - y_j^c)a_{i,j}]^2 - [\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t - y_j^c)a_{i,j}]^2$
2	$\gamma \leq 0,\ \sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t - y_j^c)a_{i,j} \leq 0$	$\max_{a_{i,j}\in\mathcal{M}} [\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t - y_j^c)a_{i,j}]^2 - \sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}[(y_i^t - y_j^c)a_{i,j}]^2$
	Maximization problems	
1	$\gamma \geq 0,\ \sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}[(y_i^t - y_j^c)a_{i,j}] \geq 0$	$\max_{a_{i,j}\in\mathcal{M}} [\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t - y_j^c)a_{i,j}]^2 - \sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}[(y_i^t - y_j^c)a_{i,j}]^2$
2	$\gamma \leq 0,\ \sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}[(y_i^t - y_j^c)a_{i,j}] \leq 0$	$\max_{a_{i,j}\in\mathcal{M}} \sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}[(y_i^t - y_j^c)a_{i,j}]^2 - [\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t - y_j^c)a_{i,j}]^2$

Note. All the QIPs are subject to Constraints (4)–(8) and the case constraints.

both problems results in a linear feasibility problem:

$$\left(\frac{n\gamma^2}{n+\gamma^2}\right)\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}[(y_i^t - y_j^c)a_{i,j}]^2 = \left[\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t - y_j^c)a_{i,j}\right]^2. \tag{29}$$

Even though we converted the general nonlinear integer optimization problems to QIPs, it is still computationally expensive to solve practical-sized problems. Therefore, in the following, we develop greedy algorithms to solve the QIPs in Table 2. We propose three algorithms for solving both the minimization and maximization problems. First, in Algorithm 1, we propose a greedy scheme for efficiently solving Cases 1 and 2 of the minimization problem. Second, in Algorithm 2, we extend a variant of this setup to the maximization problem for Cases 1 and 2. Case 3 of both the minimization and maximization problems are discussed in Algorithm 3. Finally, in Algorithm 4, we provide a combined framework for solving the Z-test problem using Algorithms 1–3. Before we delve into the details of the algorithms, first let us define the following constants for Algorithm 1:

$$\text{Case 1}: \quad \alpha_1 = 1,\ \alpha_2 = 0, \quad \text{Case 2}: \quad \alpha_1 = 0,\ \alpha_2 = -1. \tag{30}$$

Moreover, for all $i \in \mathcal{T}, j \in \mathcal{C}$, we set $\Delta_{ij} = [y_i^t - y_j^c] \odot \mathbf{D}_{ij}$ and construct the list Υ as follows:

$$\Upsilon[r,(i_r,j_r)] = (\Upsilon)_r = \Delta_{i_r j_r};$$
$$\Upsilon[r,(i_r,j_r)] \leq \Upsilon[r+1,(i_{r+1},j_{r+1})],\ \Delta_{i_r j_r} \neq 0 \ \ \forall r \in \|\mathbf{D}\|_0. \tag{31}$$

This list will be used as input list Υ in Algorithms 1 and 2.

Algorithm 1 ($\gamma = \mathcal{G}_1(\Upsilon, n, \alpha_1, \alpha_2)$)

Initialize $k \leftarrow 0$, $\epsilon_k \leftarrow \mathbf{0}$, $l \leftarrow |\Upsilon|$;
while $k \leq \alpha_1\lceil\frac{n}{2}\rceil + |\alpha_2|n$ **or** $\epsilon_k > 0$, **do**
  If $\alpha_2\Upsilon[1,(i_1,j_1)] + \alpha_1\Upsilon[l,(i_l,j_l)] < 0$
    Stop; No feasible solution.
  Else
    If $|\Upsilon[1,(i_1,j_1)]| - \alpha_2\Upsilon[1,(i_1,j_1)] - \alpha_1\Upsilon[l,(i_l,j_l)] \geq 0$
      Assign $a_{i_l,j_l} = 1$.
    Else
      Assign $a_{i_r,j_r} = 1$ such that $|\Upsilon[1,(i_1,j_1)] + \Upsilon[r,(i_r,j_r)]|$ is minimized.
    Denote the assigned entry as $a_{i_{\bar{r}}j_{\bar{r}}}$ and remove the entries of the list that contains indices $i_{\bar{r}}$ or $j_{\bar{r}}$.
    $\epsilon_k \leftarrow \epsilon_k + \Upsilon[\bar{r},(i_{\bar{r}},j_{\bar{r}})]$;
  If $\alpha_1 = 1$
    Assign $a_{i_p,j_p} = 1$ such that $\Upsilon[p,(i_p,j_p)] + \Upsilon[\bar{r},(i_{\bar{r}},j_{\bar{r}})] \geq 0$ and is minimized. Remove the entries that contains the indices i_p or j_p from the list.
    $l \leftarrow l - 2$;
    $k \leftarrow k + 1$;
  Else
    $l \leftarrow l - 1$;
    $k \leftarrow k + 1$;
end while
return $\gamma = \alpha_2\gamma_{\min} + \alpha_1\gamma_{\max}$; $\gamma_{\min}$ and $\gamma_{\max}$ are respectively the minimum and maximum roots of Equation (29).

In a similar fashion, the following constants will be used in Algorithm 2:

$$\text{Case 1}: \quad \beta_1 = -1,\ \beta_2 = 0, \quad \text{Case 2}: \quad \beta_1 = 0,\ \beta_2 = 1. \tag{32}$$

Algorithm 2 ($\gamma = \mathcal{G}_2(\Upsilon, n, \beta_1, \beta_2)$)

Initialize $k \leftarrow 0$, $\epsilon_k \leftarrow \mathbf{0}$, $l \leftarrow |\Upsilon|$;
while $k \leq \beta_2\lceil\frac{n}{2}\rceil + |\beta_1|n$ **or** $\epsilon_k > 0$, **do**
  If $\beta_1\Upsilon[1,(i_1,j_1)] + \beta_2\Upsilon[l,(i_l,j_l)] < 0$
    Stop; No feasible solution.
  Else
    If $(1+\beta_1)|\Upsilon[1,(i_1,j_1)]| - \beta_2\Upsilon[l,(i_l,j_l)] \leq 0$
      Assign $a_{i_l,j_l} = 1$.
    Else
      Assign $a_{i_r,j_r} = 1$ such that $\Upsilon[l,(i_l,j_l)] - \Upsilon[r,(i_r,j_r)]$ is maximized.
    Denote the assigned entry as $a_{i_{\bar{r}}j_{\bar{r}}}$ and remove the entries of the list that contains indices $i_{\bar{r}}$ or $j_{\bar{r}}$.

Published in INFORMS Journal on Computing on August 10, 2022 as DOI: 10.1287/ijoc.2022.1226.
This article has not been copyedited or formatted. The final version may differ from this version.

$\epsilon_k \leftarrow \epsilon_k + \Upsilon[\bar{r},(i_{\bar{r}},j_{\bar{r}})]$;
If $\beta_2 = 1$
Assign $a_{i_p,j_p} = 1$ such that $\Upsilon[p,(i_p,j_p)] - |\Upsilon[\bar{r},(i_{\bar{r}}, j_{\bar{r}})]|$ and is maximized. Remove the entries that contains the indices i_p or j_p from the list.
$l \leftarrow l-2$;
$k \leftarrow k+1$;
Else
$l \leftarrow l-1$;
$k \leftarrow k+1$;
end while
return $\gamma = \beta_2\gamma_{\min} + \beta_1\gamma_{\max}$; $\gamma_{\min}$ and $\gamma_{\max}$ are respectively the minimum and maximum roots of Equation (29).

In Algorithm 1, we consider Cases 1 and 2. For Case 3, we can design an exact method by leveraging the assignment problem structure. Note that, for this case, the nonlinear feasibility problem becomes a linear feasibility problem with constraint $\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}[(y_i^t - y_j^c)\, a_{i,j}] \le 0$ along with necessary assignment constraints. Solving this linear feasibility problem is similar to solving an assignment problem where the assignment cost of assigning the ith treated sample to the jth control sample is $y_i^t - y_j^c$. Using this idea, we propose Algorithm 3 to solve Case 3 of the minimization problem.

Algorithm 3 ($\gamma = \mathcal{G}_3(\Delta, n)$)

Step 1: Put a sufficiently large number $\mathcal{M}$ where assignments are not possible. Convert the negative entries of Δ to nonnegative by adding $\min\Delta_{i,j}$ to all the elements. Assume that the new cost matrix is Δ'.

Step 2: Solve the assignment problem for cost matrix Δ' with the Hungarian-Munkres algorithm to minimize (for the minimization problem) and maximize (for the maximization problem).

Step 3: For a given n, take the first n assignments achieved in Step 2 and calculate the total cost of the assignment using the original cost matrix Δ. Set ϵ^k = total assignment cost of n pairs. If $\epsilon^k \le 0$ (maximization, $\epsilon^k \ge 0$) then, $\gamma^* = 0$; otherwise, no feasible solution is possible.

For Case 3 of the maximization problem, we follow the same scheme, except that, in the maximization problem, instead of minimizing, we maximize the assignment cost.

Algorithm 4 ($\gamma^* = \mathcal{A}(\Delta, n)$)
Sort the list Δ following (31).
For the minimization problem:
Solve $\gamma = \mathcal{G}_1(\Upsilon, n, 0, -1)$
If no feasible solution.
Solve $\gamma = \mathcal{G}_3(\Delta, n)$
If no feasible solution.
Solve $\gamma = \mathcal{G}_1(\Upsilon, n, 1, 0)$
If no feasible solution.
n pairs are not possible.
Else return the current γ as optimal.
Else return the current γ as optimal.
For the maximization problem:
Solve $\gamma^* = \mathcal{G}_2(\Upsilon, n, -1, 0)$
If no feasible solution.
Solve $\gamma = \mathcal{G}_3(\Delta, n)$
If no feasible solution.
Solve $\gamma^* = \mathcal{G}_2(\Upsilon, n, 0, 1)$
If no feasible solution.
n pairs are not possible.
Else return the current γ as optimal.
Else return the current γ as optimal.
Return γ^*.

4.1. Properties of the Greedy Algorithms

As we are making assignments in a greedy way, we may achieve a local optimal solution instead of a global optimal solution. In addition, we may have a smaller number of matched pairs than an exact method, as a greedy approach ignores the combinatorial nature of the problem. In this section, we discuss properties of the algorithms that show that our algorithms can achieve a global optimal solution in some cases, and, for specific matching restrictions, we will have the same number of pairs as an exact method. In addition, we discuss the time complexity of the proposed algorithms.

Proposition 1. *If the set of good matches $\mathcal{M}$ is identified through exact matching, then $\boldsymbol{\Delta}$ can be partitioned into a set of disjoint matrices $\{\Delta^1, \Delta^2, \ldots, \Delta^r, \ldots\}$, and each row in Δ^r is identical.*

Proof. In exact matching, a treated sample t_i can be matched to a control sample c_{j_1} if covariate vector $\mathbf{X}_i^t = \mathbf{X}_{j_1}^c$. In the matrix D (representing the set of good matches $\mathcal{M}$), the entry $d_{i,j_1} := 1$ if $\mathbf{X}_i^t = \mathbf{X}_{j_1}^c$; otherwise, $d_{i,j_1} := 0$. Let's assume that treated sample t_i can be matched to the set of control samples $c' = \{c_{j_1}, c_{j_2}, c_{j_3}, \ldots, c_{j_q}\}$. As the matching is done exactly, we can say that $\mathbf{X}_i^t = \mathbf{X}_{j_1}^c = \mathbf{X}_{j_2}^c = \mathbf{X}_{j_3}^c = \cdots = \mathbf{X}_{j_q}^c$. So, $d_{i,j_l} := 1$, $\forall l \in \{1,2,\ldots,q\}$ and the rest of the values in the ith row of the matrix D are zero. Now, assume that treated unit t_h is a good match to a control sample $c_k \in c'$; then, by the definition of exact match, t_h is a good match for all the control samples in c'. Hence, the vector d_{h,j_l} will be identical to d_{i,j_l}. Now, by reorganizing the rows of D, we can partition it into disjoint matrices, where rows within each of the matrices are identical. As $\boldsymbol{\Delta} = S \odot D$, it can be partitioned into disjoint matrices $\{\Delta^1, \Delta^2, \ldots, \Delta^r, \ldots\}$. □

From Proposition 1, we see that the $\boldsymbol{\Delta}$ can be partitioned into disjoint matrices and rows within a

Published in INFORMS Journal on Computing on August 10, 2022 as DOI: 10.1287/ijoc.2022.1226.
This article has not been copyedited or formatted. The final version may differ from this version.

partitioned matrix are identical. So, by making greedy assignments, we will not lose other possible assignments at any point in the future.

Proposition 2. *If $(y_i^t - y_j^c)a_{i,j}$ is considered as the assignment cost of assigning treated unit i to control unit j, where $a_{i,j} \in \{0,1\}$ is an assignment variable, then the total assignment cost $\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t - y_j^c)a_{i,j}$ is independent of the order of assignment in Δ^r.*

Proof. From Proposition 1, we know that, in Δ^r, each row has nonzero elements in the same columns and the respective partition of D has identical rows. Now assume that a possible assignment $\mathcal{A}_1 = \{a_{1,1} = 1, a_{2,2} = 1, \dots, a_{m,m} = 1, \dots\}$. The total cost of assignment of $\mathcal{A}_1$ is the following:

$$\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t - y_j^c)a_{i,j}$$

$$= (y_1^t - y_1^c) + (y_2^t - y_2^c) + \cdots + (y_m^t - y_m^c) + \cdots \quad (33)$$

$$= (y_1^t + y_2^t + \cdots + y_m^t + \cdots) - (y_1^c + y_2^c + \cdots + y_m^c + \cdots). \quad (34)$$

Now, from expression (34), we see that the total cost is independent of the information pair $a_{i,j}$; that is, we see which treated unit is paired with which control unit. Therefore, in each partition of $\mathbf{\Delta}$, the total cost of the assignment is independent of the order of pair assignment. □

Proposition 3. *The greedy scheme proposed in Algorithm 1 provides an optimal solution of Case 2 of the minimization problem when n pairs are matched. There are at least n possible pairs with $\Delta_{ij} \le 0$, and $\mathcal{M}$ is identified with exact matching.*

Proof. In Case 2 of the minimization problem, we are trying to maximize $[\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t - y_j^c)a_{i,j}]^2 - \sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}[(y_i^t - y_j^c)a_{i,j}]^2$ with Constraints (24) and (4)–(8). This quantity is increasing in $|\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t - y_j^c)a_{i,j}|$ when the difference between the outcomes $(y_i^t - y_j^c)a_{i,j}$ of all pairs are of the same sign. From Proposition 2, we can see that in each $\Delta^r \in \mathbf{\Delta}$ the cost $\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}}(y_i^t - y_j^c)a_{i,j}$ is independent of the order of assignment. Therefore, the greedy assignment will produce an optimal solution for each $\Delta^r \in \mathbf{\Delta}$. On the other hand, from Proposition 1, we see that each $\Delta^r \in \mathbf{\Delta}$ is disjoint when a good set of match $\mathcal{M}$ is identified with exact matching. Hence, the optimal solution of each disjoint set of possible assignments will produce an optimal solution of the complete assignment problem. □

Case 1 of the maximization problem follows the same scheme as the Case 2 of the minimization problem. Hence, the result in Proposition 3 is also valid for Case 1 of the maximization problem. It is important to note that Proposition 3 assumes that the matched pairs are constructed with exact matching. In reality, finding exact matches is highly unlikely when covariates are continuous. However, exact matching is possible for categorical covariates like those in the case study presented in Section 5.3. In addition, the continuous covariates can be transformed into categorical values by applying any suitable discretization technique as a preprocessing step.

Proposition 4. *Denote $d = \max\{\ln\|D\|_0, n\}$. Then the proposed greedy algorithms have running time complexity of $\mathcal{O}(d\|D\|_0)$ and have a storage cost of $\mathcal{O}(\|D\|_0)$.*

Proof. Note that if we run sorting algorithms such as *HeapSort* and *MergeSort* on a list of n entries, then the best worst-case running time complexity that can be achieved is $\mathcal{O}(n\ln n)$. In our proposed schemes, we run a sorting algorithm on the respective $\mathbf{\Delta}$ matrix, and then we input the sorted data on the main algorithms (Algorithms 1 and 2). As both algorithms run in at most n loops (i.e., $\alpha_1\lceil\frac{n}{2}\rceil + |\alpha_2|n, \beta_2\lceil\frac{n}{2}\rceil + |\beta_1|n \le n$), we can calculate the running time complexity as follows:

$$T_1(n) = n\left[1 + 1 + \|D\|_0 + 2 + \|D\|_0\right] = \mathcal{O}(n\|D\|_0).$$

Then, considering the time complexity of the sorting scheme, we can find the total time complexity of the proposed scheme as follows:

$$\begin{aligned} T(n) :=\ & \text{Sorting run time} + \text{Running time of} \\ & \text{Algorithm 1 (or Algorithm 2)} \\ =\ & \mathcal{O}(\|D\|_0 \ln\|D\|_0) + \mathcal{O}(n\|D\|_0) \\ =\ & \mathcal{O}(\max\{n, \ln\|D\|_0\}\|D\|_0) = \mathcal{O}(d\|D\|_0). \end{aligned}$$

Here, we used the fact that we run the sorting scheme on the nonzero elements of the matrix $\mathbf{\Delta}$, which has a total of $\|D\|_0$ entries. It's easy to check that both schemes have a total storage cost of $\mathcal{O}(\|D\|_0)$, as, throughout the scheme, we only need to store at most $\|D\|_0$ entries. □

Proposition 5. *The proposed heuristic for Case 3 runs in strongly polynomial time. Moreover, it will provide us an exact solution.*

Proof. Since we are using the *Hungarian-Munkres* algorithm for solving the main problem, we can calculate the time complexity of the proposed heuristic as follows:

$$T(n) = \mathcal{O}(n) + \mathcal{O}(n^3) = \mathcal{O}(n^3).$$

Here, we used the complexity result of the *Hungarian-Munkres* algorithm (Edmonds and Karp 1972) and the initial time complexity of $\mathcal{O}(n)$. Furthermore, as the *Hungarian-Munkres* scheme provides an exact solution, the proposed heuristic will also provide an exact solution. □

Published in INFORMS Journal on Computing on August 10, 2022 as DOI: 10.1287/ijoc.2022.1226.
This article has not been copyedited or formatted. The final version may differ from this version.

5. Numerical Experiments

In this section, we apply the proposed greedy algorithms to test causal hypotheses from three real-world data sets of varying size. We also compare the solution of the proposed algorithms to the quadratic integer programming (QIP) models in Table 2 solved with Gurobi 9.0.2 (Gurobi 2020) and the ILP-based heuristic proposed by Morucci et al. (2018). It is important to note that we implemented a simpler but computationally less efficient version of the proposed algorithm, as we use unsorted $\mathbf{\Delta}$ and find the maximum or minimum $\Delta_{i,j}$ at each iteration. After calculating the maximum and minimum $Z(\mathbf{a})$ using the proposed greedy algorithms and alternative methods, we convert them into p-values using the relations in Equations (35) and (36), as p-values are commonly used to make inference from the statistical test:

$$p\text{-value}_{\max} = \arg\max_{\mathbf{a}\in\mathcal{M}}[1-\phi(Z(\mathbf{a}))] = 1-\phi\left(\arg\min_{\mathbf{a}\in\mathcal{M}} Z(\mathbf{a})\right), \tag{35}$$

$$p\text{-value}_{\min} = \arg\min_{\mathbf{a}\in\mathcal{M}}[1-\phi(Z(\mathbf{a}))] = 1-\phi\left(\arg\max_{\mathbf{a}\in\mathcal{M}} Z(\mathbf{a})\right). \tag{36}$$

Here, $\phi(Z(\mathbf{a}))$ represents the area under the standard normal distribution curve to the right of $Z(\mathbf{a})$. For all the experiments, we consider the level of significance $\alpha = 0.05$ as a rule of thumb to make robust inference. Even so, our experiments show that the proposed algorithm is scalable and can handle very-large-scale problems, whereas state-of-the-art commercial solvers either provide a worse solution or cannot solve problems of moderate to large sizes. Although the experiments show interesting causal insights, we use them to demonstrate the effectiveness of our algorithms. All the experiments are performed in a Dell Precision 7510 workstation with an Intel Core i7-6820HQ CPU running at 2.70 gigahertz (GHz) and 32 gigabytes (GB) of RAM.

5.1. Effect of Fly Ash on Strength of Concrete

Fly ash, a by-product of thermal power plants, is a common element in producing concrete (Bilodeau and Malhotra 2000). In this experiment, we hypothesize that fly ash has zero effect on concrete's compressive strength. We use a concrete compressive strength data set (Yeh 1998) to test the causal hypothesis using the robust Z-test. The data set has 1,030 instances and nine attributes. The control group includes 529 samples with no fly ash component, whereas the treatment group has 501 samples with at least 24.5 kg/m^3 of fly ash. We perform the matching operation on seven pretreatment covariates. A treated unit i is a good match with control unit j (i.e., $d_{i,j} = 1$) if their differences in cement, blast furnace slag, and water are less than or equal to 30, the difference in superplasticizer is less than or equal to 20, fine and coarse aggregate is less than or equal to 50, and age is less than or equal to 5. The outcome is concrete's compressive strength. The matching process resulted in a group of 68 treated samples that can be matched with 60 control samples, where many treated samples have multiple pair assignment options. The D matrix has 146 nonzero entries, which indicates that we need to solve a nonlinear optimization problem with 146 binary variables.

Table 3 shows the maximum and minimum Z-values achieved for different numbers of pairs (n) with the greedy algorithms (GA) and solving the QIPs in Table 2 with Gurobi 9.0.2. For the QIPs, we used for the stopping criteria a 500-second time limit or a 2% optimality gap. As we can see, the QIPs provide better solutions compared with the proposed greedy algorithm. However, the greedy approach takes a fraction of a second, whereas the QIP takes more than 500 seconds. For the minimization problem with the number of pairs

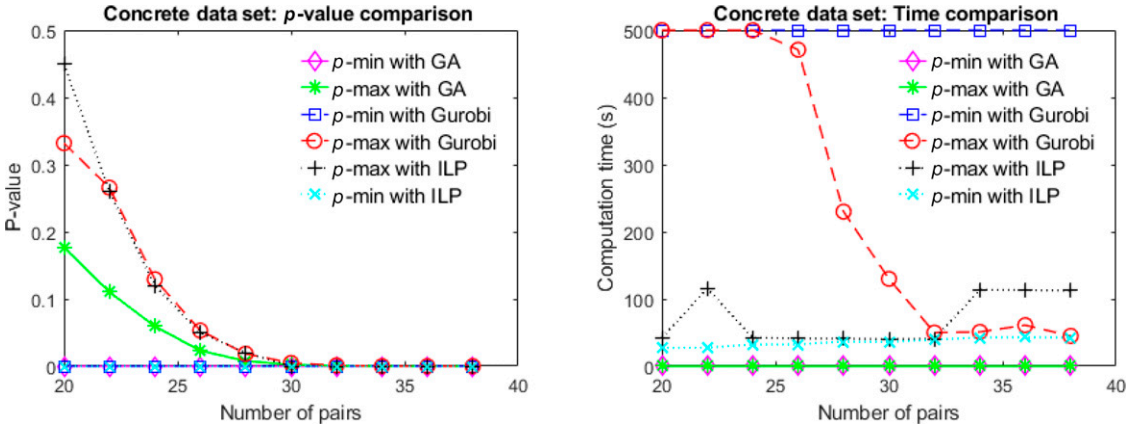
Table 3. Comparison of Greedy Algorithm (GA) and QIP Solved with Gurobi for the Concrete Compressive Strength Data Set

	Maximum Z				Minimum Z			
n	Z with GA	Z with QIP	Time GA	Time QIP	Z with GA	Z with QIP	Time GA	Time QIP
20	12.257	13.343	0.019	500	0.927	0.4346	0.028	500
22	12.446	13.555	0.016	500	1.2246	0.626	0.019	500
24	12.584	13.78	0.019	500	1.5594	1.1292	0.022	500
26	12.645	15.272	0.021	500	1.993	1.6151	0.023	471
28	12.564	15.195	0.021	500	2.424	2.0858	0.021	230
30	12.129	15.232	0.029	500	2.8573	2.5886	0.030	130
32	11.624	14.79	0.019	500	3.3077	3.0315	0.023	50
34	11.171	14.187	0.016	500	3.7772	3.525	0.029	51
36	9.5952	13.567	0.019	500	4.2498	4.0181	0.037	61
38	7.8861	12.948	0.024	500	4.7101	4.4831	0.027	45

Note. The QIP is solved with stopping criteria of a 500-second time limit or 2% optimality gap.

Published in INFORMS Journal on Computing on August 10, 2022 as DOI: 10.1287/ijoc.2022.1226.
This article has not been copyedited or formatted. The final version may differ from this version.

Figure 1. (Color online) Comparison Between Greedy Algorithm (GA), ILP-Based Heuristic (ILP) Proposed by Morucci et al. (2018), and Gurobi Solving the QIP on p-Values and Computation Time for the Concrete Data Set



greater than 24, the QIP achieves a 2% optimality gap in a fairly short duration, but, after that, it takes days to reach optimality. The resulting p-values are presented in Figure 1. In Figure 1, we also include the result using the ILP-based heuristic proposed by Morucci et al. (2018). For the number of pairs greater than 26, all three algorithms achieve p-values less than 0.05, and, after 38 pairs, both the greedy algorithm and the ILP-based heuristic did not find any pairs. As both the maximum and minimum p-values coincide in Figure 1, we achieve an absolute robust test according to Definition 3. Even though the ILP-based heuristic and QIP perform better in terms of solution quality, for a larger number of pairs, the greedy algorithm achieves similar quality solutions. All three algorithms reject the hypothesis of the zero treatment effect, which supports the traditional knowledge of fly ash's positive effect on concrete strength (Yeh 1998). This result shows that the proposed greedy approach (GA) can achieve the same conclusion as the ILP-based heuristics and QIPs, but in a significantly smaller amount of time.

5.2. Effect of Misty Weather on Number of Bike Rentals

In this experiment, we consider a slightly larger-sized problem compared with the concrete compressive strength data set. We evaluate the effect of misty weather on the number of bike rentals. Our naive hypothesis is that there is no effect of mist on the usage of rental bikes. To test this hypothesis, we use a bike-sharing data set (Fanaee-T and Gama 2014) available in the UCI Machine Learning Repository. The data set contains the daily count of bikes rented for 731 days, weather, and seasonal information of corresponding days between 2011 and 2012 in the capital bike-share system in Washington, DC. We consider 247 days as the treatment group, which had mist with a different combination of clouds; 463 days are considered as the control group, which consisted of days with clear skies, few clouds, or partial clouds (without any mist). The seasonal information such as season, year, and workday were matched exactly. The weather variables such as temperature, wind speed, and humidity were matched if the differences were less than or equal to 2, 6, and 6, respectively. If treated sample i and control sample j follow the above criteria, then $d_{i,j} = 1$, and 0 otherwise. This matching process produced a nonlinear optimization problem with 326 binary variables, more than double in size compared with the previous experiment. The robust test statistics achieved by the proposed algorithms and solving the QIPs of Table 2 with Gurobi, as well as the computation times, are provided in Table 4 and Figure 2. We followed the same QIP stopping criteria as in the previous experiment.

From Table 4, we can see that the greedy approach outperforms the QIP for all numbers of pairs. In fact, the QIP could not find an initial integer solution within the time limit for n greater than 82. For n between 50 and 82, the QIP solution improves marginally after 500 seconds. On the other hand, the greedy algorithm finds better solutions with a significantly lower amount of time for a number of matched pairs up to 88.

We also compare the results from the greedy algorithm with the ILP-based heuristic of Morucci et al. (2018) in Figure 2. For a fair comparison, we use the same matching algorithm and recommended heuristic settings as described in Morucci et al. (2018). Figure 2 shows that the greedy algorithm finds 88 pairs in the good set of matches, and, after 87 matched pairs, both maximum and minimum p-values are less than $\alpha = 0.05$. Therefore, we have an α-robust test that fails to reject our hypothesis on the effect of mist on rental bike usage. The ILP-based heuristic also achieves the α-robust test; however, it finds 94 matched pairs in the data set. As the ILP-based heuristic solves an integer

Published in INFORMS Journal on Computing on August 10, 2022 as DOI: 10.1287/ijoc.2022.1226.
This article has not been copyedited or formatted. The final version may differ from this version.

Table 4. Comparison of Greedy Algorithm (GA) and QIP Solved with Gurobi for the Bike-Sharing Data Set

	Maximum Z				Minimum Z			
n	Z with GA	Z with QIP	Time GA	Time QIP	Z with GA	Z with QIP	Time GA	Time QIP
50	6.9736	6.7914	0.034	500	−11.5334	−10.1657	0.035	500
55	6.0925	5.8509	0.032	500	−11.2922	−9.6567	0.034	500
60	5.1916	4.8345	0.034	500	−10.9022	−9.1222	0.028	500
65	4.2864	3.7268	0.036	500	−10.5792	−8.4047	0.036	500
70	3.4461	2.6946	0.039	500	−10.181	−7.5596	0.036	500
75	2.5065	1.0067	0.034	500	−9.6883	−6.2907	0.041	500
80	1.4238	0.2036	0.043	500	−9.1318	−5.0902	0.036	500
81	1.216	−0.2893	0.039	500	−9.0138	−4.7659	0.036	500
82	0.9255	−0.5338	0.036	500	−8.9001	−4.4127	0.039	500
83	0.6468	—	0.049	500	−8.7803	—	0.050	500
84	0.3713	—	0.039	500	−8.6469	—	0.047	500
85	−1.2046	—	0.045	500	−8.4748	—	0.036	500
86	−1.3929	—	0.037	500	−8.2821	—	0.036	500
87	−1.6767	—	0.053	500	−8.0448	—	0.046	500
88	−1.9539	—	0.043	500	−7.7887	—	0.037	500

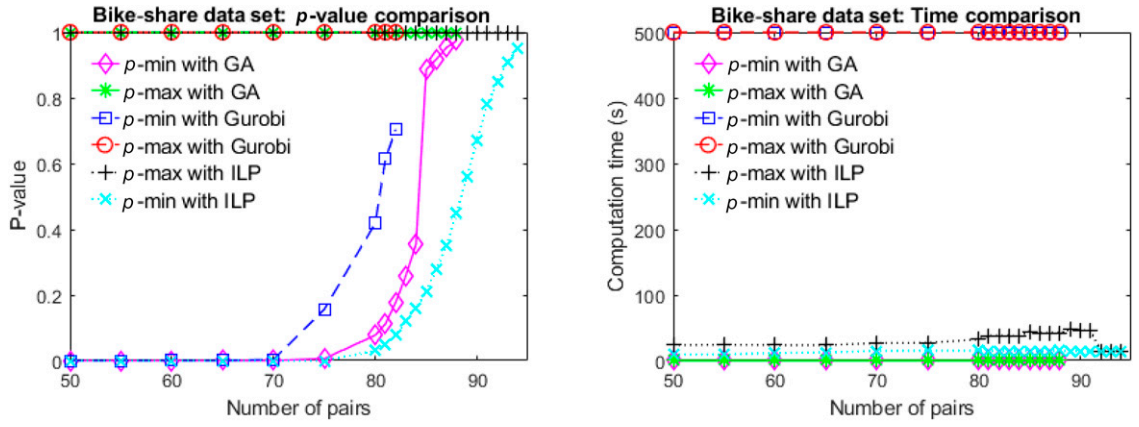
Notes. The QIP is solved with stopping criteria of a 500-second time limit or a 2% optimality gap. "—" implies no integer solution found within the time limit.

linear program iteratively by bounding the sample standard deviation, it finds the maximum number of pairs. In contrast, the proposed algorithm assigns pairs greedily, so it may select a smaller number of pairs. Nonetheless, as we show in Proposition 3, the greedy algorithm can achieve an optimal solution and the maximum number of pairs if the matching is performed using exact matching methods. In terms of computation time, the greedy algorithm takes a fraction of a second, whereas the ILP-based heuristic takes more than nine seconds to solve an instance of the problem.

5.3. Effect of Product's Location on Price

To show the scalability of the proposed greedy algorithm, we consider a very-large-scale clickstream data set of online shopping. We used the clickstream data from Lapczynski and Bialowas (2013), which contain information on clicks from online stores that are selling clothes for pregnant women. In this paper, we refer to the data set as the *e-shop data set*. We hypothesize that if a store web page is split horizontally into two panels, top and bottom, then the top part is as valuable as the bottom part and high-price products are not always placed on the top panels. The data set consists of 165,474 instances of clicks, and each instance contains information on the product clicked on, time of the click, and origin of the IP address. The treatment group includes more than 35,000 click instances on the top-left panel products, and the control group includes more than 27,000 clicks on products displayed on the bottom-left panel. We implemented exact matching on

Figure 2. (Color online) Comparison Between Greedy Algorithm (GA), ILP-Based Heuristic (ILP) Proposed by Morucci et al. (2018), and Gurobi Solving the QIP on p-Values and Computation time for the Bike-Sharing Data Set



Published in INFORMS Journal on Computing on August 10, 2022 as DOI: 10.1287/ijoc.2022.1226.
This article has not been copyedited or formatted. The final version may differ from this version.

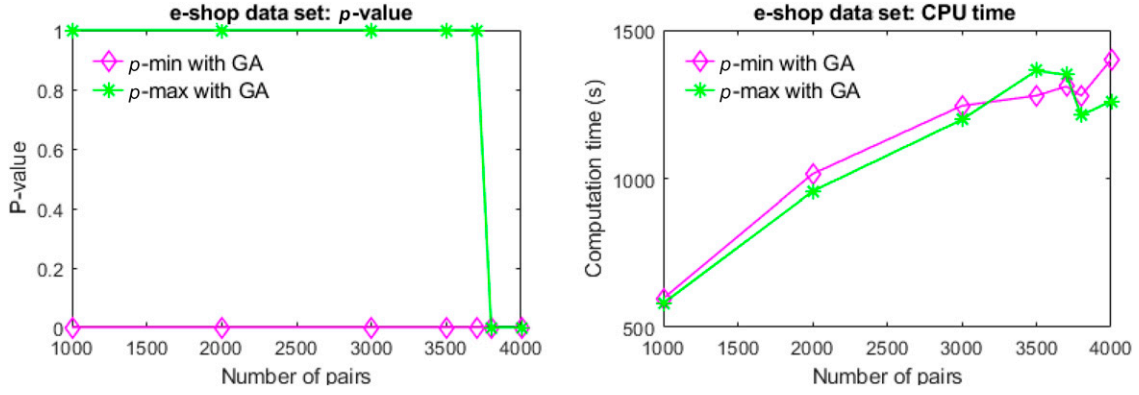
the pretreatment covariates: month, order, country of IP address, product category, color, model photography, and page number within the website. For treated and control samples i and j, $d_{i,j} = 1$ if two samples have exactly the same value on all the covariates. Product price is the outcome. Compared with the concrete strength and bike-sharing examples, the e-shop data set has a significantly large number of samples and produces a nonlinear optimization problem with more than 350,000 binary variables. The decision problem of this scale is almost impossible to solve with state-of-the-art commercial solvers. However, the proposed greedy algorithm can solve the problem in a reasonable amount of time considering the problem size.

Figure 3 presents the maximum and minimum p-values achieved with the greedy algorithms for the robust Z-test and the time required for a different number of pairs. After 3,800 pairs, both the maximum and minimum p-values coincide at zero. So, we reject the hypothesis and conclude that the product location and price have a causal relation: high-price products are placed on the top panel. From CPU time consumption in Figure 3, we can see that the greedy algorithm takes a significantly higher time at around 1,300 seconds to solve the Z-test problems for e-shop data compared with the concrete strength and bike-sharing data sets. However, considering a nonlinear optimization problem with over 350,000 binary variables, the time consumption can be considered reasonable. On the other hand, after running several hours, the ILP-based heuristic and QIP (solved with Gurobi) ran out of memory and could not find a solution.

As the ILP-based heuristic performs better than the greedy heuristics on the smaller data sets, we conducted three sets of experiments to compare their performance on e-shop data set with increased memory and more processors. We used a more powerful computing workstation with an Intel Xeon CPU E5-2670 v3 running at 2.30 GHz (two processors) with 64 GB of memory for these experiments. We conducted the same experiment on the complete e-shop data set. Unfortunately, the increased memory is not enough for the ILP-based heuristic, as it ran out of memory and could not find a solution for the complete e-shop data set. Therefore, to compare the performance of the ILP-based heuristic with the proposed greedy method when memory is not an issue, we created two data sets by taking (i) the first 25% and (ii) the first 10% of treated and control samples from the e-shop data set. We were able to conduct experiments on these two reduced data sets without running out of memory. However, the ILP-based heuristic was not able to solve a single instance (i.e., for a specific number of pairs n) of the robust causal inference problem within four days when 25% samples are used. On the other hand, for the smaller data set, one with the first 10% of treated and control samples, the ILP-based heuristic was able to solve the problem within a reasonable amount of time. We present the p-value and computation time comparison between these methods in Figure 4. As shown in Figure 4, the ILP-based heuristic performs marginally better for the larger number of pairs. However, the greedy heuristics can solve the problems in a fraction of seconds, whereas the ILP-based heuristic takes more than 500 seconds with a 1% MIP gap. The p-value comparison shows that both methods provide the same inference decision that a robust conclusion could not be made possible.

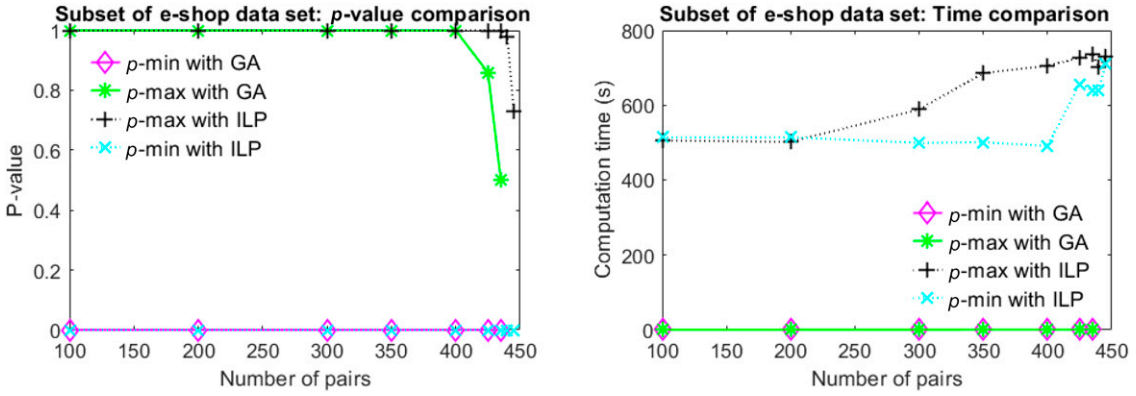
Considering the results from the three data sets, it is evident that both the ILP-based heuristic and solving the QIP with a commercial solver provide solutions of marginally better quality but nonetheless find the same inferential conclusion. For very small problems (i.e., problems similar to the concrete compressive strength data sets), the ILP-based heuristic can be a reasonable choice; however, in today's big data world, such small problems are highly unlikely in practice. In contrast, the greedy algorithm proposed in this paper

Figure 3. (Color online) Comparison Between Greedy Algorithm (GA) and Gurobi Solving the QIP on p-Values and Computation Time for the e-Shop Data Set



Published in INFORMS Journal on Computing on August 10, 2022 as DOI: 10.1287/ijoc.2022.1226.
This article has not been copyedited or formatted. The final version may differ from this version.

Figure 4. (Color online) Comparison Between Greedy Algorithm (GA) and ILP-Based Heuristic on *p*-Values and Computation Time for the 10% Samples of the e-Shop Data Set



provides same conclusion on the robust inference in a significantly smaller amount of time and is highly scalable for practical-sized problems.

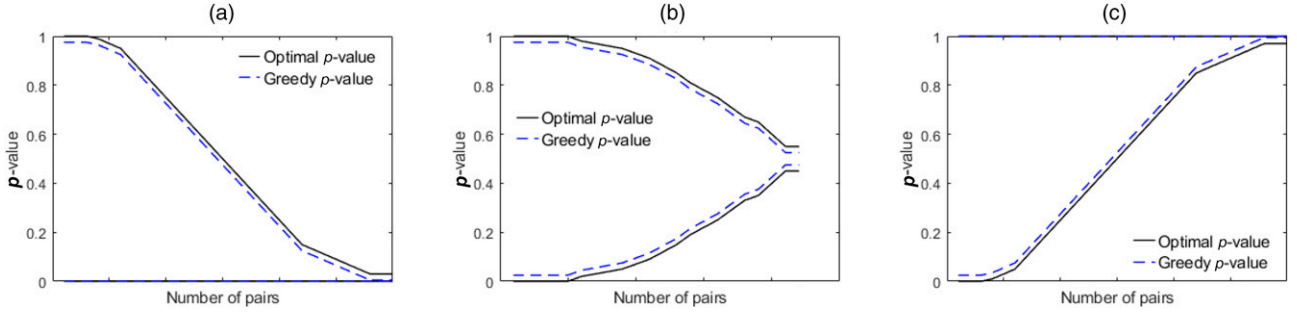
5.4. Inferential Implication

In general, the greedy algorithms proposed in this paper make local optimal choices. In this section, we investigate the inferential implication of such approximate solution choices by identifying how much deviation from the global optimal solution is allowable to preserve the actual inference of the robust hypothesis test.

In Figure 5, we present three possible scenarios of *p*-value for the robust Z-test. The *p*-values can be close to 0 or 1 (Figure 5, (a) and (c), respectively) or somewhere in the middle (Figure 5(b)). Let's assume that $Z_{\max}^{greedy}$ is the solution of the greedy algorithm and that $Z_{\max}^{opt}$ is the global optimal solution of the maximization problem in any scenario. Now, as the greedy algorithm will produce a suboptimal solution, we can say that $Z_{\max}^{greedy} = Z_{\max}^{opt} - gap$, where *gap* represents the optimality gap between two methods. Therefore, based on Equations (35) and (36), *p*-values with greedy algorithm and optimal solution will have the following relation: $P_{\min}^{greedy} \geq P_{\min}^{opt}$. Now, for a level of significance α, the *p*-value with greedy algorithm ($P_{\min}^{greedy}$) can deviate by the amount α without changing the conclusion on robustness. As the *p*-value is calculated from the standard normal distribution curve, we can establish a connection between α and $gap = Z_{\max}^{opt} - Z_{\max}^{greedy}$ using Equations (35) and (36)—namely, how much deviation on the $Z(\mathbf{a})$ we can allow without altering the inference on robustness. We can derive a similar relation for $P_{\max}$ from the minimization problem following the same argument.

Using $\alpha = 0.05$ as the rule of thumb and the probabilities from the standard normal distribution table, we can calculate the maximum allowable gap. For all the scenarios, if the optimal solutions of both the maximization and minimization problems are very large (i.e., $Z \geq 4$ in Figure 5(a)) for the maximum number of pairs (n), like the solutions of the e-shop and concrete compressive strength data sets, then it would require more than a 50% optimality gap between the greedy and global optimal solutions to reverse the inference on robustness. This also applies to the situation when both problems have very small Z-values (i.e., $Z \leq -4$ in Figure 5(c)) for the maximum value of n in the data

Figure 5. (Color online) Different Scenarios Showing the Gap Between *p*-Values Obtained by an Exact Method and the Greedy Algorithm



Published in INFORMS Journal on Computing on August 10, 2022 as DOI: 10.1287/ijoc.2022.1226.
This article has not been copyedited or formatted. The final version may differ from this version.

set, as the area under the normal distribution curve is very small for $Z \geq |4|$. When $Z \leq |4|$, an approximate allowable *gap* is 18%. In the case Concrete Compressive Strength study, we know the optimal solution for seven instances. For those seven instances, the average gap between the greedy and global optimal solution is around 11% which is lower than the allowable gap. For the cases in Figure 5, (a) and (c), we will have the same allowable gap due to the symmetry of the standard normal distribution curve. When both the maximum and minimum Z-values deviate from optimality, as shown in Figure 5(b), we can split the allowable gap in half for each problem. Despite the fact that the observed optimally gaps of the proposed greedy algorithms are well below the maximum allowable gap, our analysis shows that the proposed greedy algorithms will find an optimal solution (with zero gap) for restricted cases (i.e., when match pairs are constructed with exact matching). If all the variables in the data set are categorical, then we can apply an exact matching algorithm to find the good set of matches. In addition, we can take the same advantage with continuous data by converting them into categories by applying domain knowledge or any appropriate discretization algorithm.

5.5. Practical Guideline

In the numerical experiments, we show the test statistics for a wide variety of sample sizes (n). Our purpose here is to show the level of uncertainty in the inference; however, in practice, the experimenter's goal is to find the robust inference. Hence, we do not need to consider such a wide range of n. Instead, we can pick a suitable n and increase (decrease) it until the problem becomes infeasible (or feasible) due to Constraint (18): $\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{C}} a_{ij} = n$. In the good set of matches, multiple treated units can be a good match to a single control unit (or the other way around). As we are considering a one-to-one pair assignment, we may not be able to assign pairs for all the treated or control samples. Therefore, if we keep increasing n, then, after a certain number of pairs, the optimization problem will be infeasible. Conversely, if we select a large n and decrease it, for a certain number of pairs, the problem will be feasible. Unfortunately, identifying that specific value of n is not possible without solving a combinatorial optimization problem. In this case, an efficient approach can be to start with the highest and lowest values of n possible ($n_{\max}, n_{\min}$) and apply a binary search strategy until the problem becomes infeasible.

We recommend starting the experiment with $n_{\max}$, as the smallest number of treated or control samples in the good set of matches $D^{|\mathcal{T}|\times|\mathcal{C}|}$: $n_{\max} = \min(|\mathcal{T}|, |\mathcal{C}|)$, where $|\mathcal{T}|$ is the number of treated samples that are good matches for $|\mathcal{C}|$, the number of control samples. The experimenter can set $n_{\min}$ based on the number possible good matches in the data set. If there are many nonzero entries available in $D^{|\mathcal{T}|\times|\mathcal{C}|}$, then an experimenter can choose a larger value of $n_{\min}$ that can save significant computation time. For instance, in the experiment with the concrete compressive strength data set, we can start with $n_{\max} = 60$, as the matching algorithms provided a group of 68 treated samples that can be matched with 60 control samples. If we apply a binary search strategy considering $n_{\min} = 20\%$ of $n_{\max}$, then we can find a robust test with a single run of the proposed algorithm. Using the exact strategy for the Bikeshare data set, we can find a robust solution by checking for only two different values of n.

6. Conclusion

In this study, we investigate the robust causal hypothesis test, the robust Z-test, from observational data with continuous outcomes and develop a unique computational framework that includes a novel reformulation technique and efficient algorithms. The robust Z-test produces nonlinear integer optimization problems that are difficult to solve for very small data sets, where, in today's big data world, causal hypothesis test problems are becoming larger and larger. We reformulated the nonlinear optimization models of the robust Z-test into feasibility problems. By leveraging the structure of the reformulation, we developed greedy algorithmic schemes that are very efficient and scalable. The feasibility reformulation also allows us to pose the robust test problems as quadratic integer programming problems, and, for smaller data sets, we can use any commercial MIP solvers to achieve exact solutions. Moreover, the proposed unique reformulation scheme can be used to model general nonlinear and quadratic optimization problems (e.g., the quadratic assignment problem) as feasibility problems. Apart from the scalability, we show that the greedy approaches achieve the global optimal solution in many cases. The effectiveness of the proposed algorithms is demonstrated with three real-world case studies of varying sizes and by comparing the result with equivalent QIPs solved with the exact method and the ILP-based heuristic proposed by Morucci et al. (2018). Numerical experiments on the case studies reveal that the proposed algorithms achieve the same inference as the exact method for the small test case; however, it takes significantly less computational time. On the other hand, for moderate to large instances, the proposed algorithms significantly outperform both methods. For moderately size problems, our algorithm produces a better solution in a fraction of a second, whereas the exact method struggles to find any integer

Published in INFORMS Journal on Computing on August 10, 2022 as DOI: 10.1287/ijoc.2022.1226.
This article has not been copyedited or formatted. The final version may differ from this version.

solution in hundreds of seconds. With the availability of observational data and increasing use of causal inference in decision making, our algorithms can be very effective in harnessing the power of big data in the decision-making process. A major limitation of the greedy algorithms is that they provide local optimal solutions; hence, there is a chance of altering the robustness inference. Nonetheless, it would take a significantly large optimally gap to change the conclusion on robustness, and alternative methods cannot solve large-scale problems.

As a future extension, we plan to use the feasibility formulation of the Z-test to develop several potential algorithms. First, we plan to develop an iterative projection-based algorithm to solve the feasibility problems with quadratic constraints and binary variables. Then, we intend to incorporate algorithmic acceleration schemes to further improve the efficiency of the iterative algorithm and ensure scalability.

References

Achterberg T, Wunderling R (2013) Mixed Integer Programming: Analyzing 12 Years of Progress. Jünger M, Reinelt G, eds. *Facets of Combinatorial Optimization* (Springer, Berlin), 449–481.

Alvarez AM, Louveaux Q, Wehenkel L (2014) A supervised machine learning approach to variable branching in branch-and-bound. Preprint, submitted May 15, https://hdl.handle.net/2268/167559.

Anthony M, Boros E, Crama Y, Gruber A (2017) Quadratic reformulations of nonlinear binary optimization problems. *Math. Programming* 162(1-2):115–144.

Archetti C, Guerriero F, Macrina G (2020) The online vehicle routing problem with occasional drivers. *Comput. Oper. Res.* 127:105144.

Basu A, Loera JAD, Junod M (2014) On Chubanov's method for linear programming. *INFORMS J. Comput.* 26(2):336–350.

Bertsimas D, Dunn J (2019) *Machine Learning Under a Modern Optimization Lens* (Dynamic Ideas, Charlestown, MA).

Bertsimas D, King A, Mazumder R (2016) Best subset selection via a modern optimization lens. *Ann. Statist.* 44(2):813–852.

Bilodeau A, Malhotra VM (2000) High-volume fly ash system: Concrete solution for sustainable development. *Materials J.* 97(1):41–48.

Bixby RE (2012) A brief history of linear and mixed-integer programming computation. *Doc. Math.* Extra vol.:107–121.

Boros E, Hammer PL, Tavares G (2007) Local search heuristics for quadratic unconstrained binary optimization (QUBO). *J. Heuristics* 13(2):99–132.

Cafieri S, Omheni R (2017) Mixed-integer nonlinear programming for aircraft conflict avoidance by sequentially applying velocity and heading angle changes. *Eur. J. Oper. Res.* 260(1):283–290.

Chubanov S (2012) A strongly polynomial algorithm for linear systems having a binary solution. *Math. Programming* 134(2): 533–570.

Chubanov S (2015) A polynomial projection algorithm for linear feasibility problems. *Math. Programming* 153(2):687–713.

Coker B, Rudin C, King G (2021) A theory of statistical inference for ensuring the robustness of scientific results. *Management Sci.* 67(10):6174–6197.

De Bernardi CA, Miglierina E, Molho E (2019) Stability of a convex feasibility problem. *J. Global Optim.* 75(4):1061–1077.

Dehejia RH, Wahba S (1999) Causal effects in nonexperimental studies: Reevaluating the evaluation of training programs. *J. Amer. Statist. Assoc.* 94(448):1053–1062.

Diamond A, Sekhon JS (2013) Genetic matching for estimating causal effects: A general multivariate matching method for achieving balance in observational studies. *Rev. Econom. Statist.* 95(3):932–945.

Edmonds J, Karp RM (1972) Theoretical improvements in algorithmic efficiency for network flow problems. *J. ACM* 19(2): 248–264.

Escalante R, Raydan M (2011) *Alternating Projection Methods* (SIAM, Philadelphia).

Etheve M, Alès Z, Bissuel C, Juan O, Kedad-Sidhoum S (2020) Reinforcement learning for variable selection in a branch and bound algorithm. Hebrard E, Musliu N, eds. Proc. *Internat. Conf. Integration Constraint Programming Artificial Intelligence Oper. Res.* (Springer, Cham, Switzerland), 176–185.

Fanaee-T H, Gama J (2014) Event labeling combining ensemble detectors and background knowledge. *Progress Artificial Intelligence* 2(2):113–127.

Fischetti M, Lodi A (2010) *Heuristics in Mixed Integer Programming* (Wiley, New York).

Fischetti M, Monaci M (2012) Branching on nonchimerical fractionalities. *Oper. Res. Lett.* 40(3):159–164.

Fogarty CB (2020) Studentized sensitivity analysis for the sample average treatment effect in paired observational studies. *J. Amer. Statist. Assoc.* 115(531):1518–1530.

Gangl M (2010) Causal inference in sociological research. *Annual Rev. Sociol.* 36:21–47.

Glover F, Alidaee B, Rego C, Kochenberger G (2002) One-pass heuristics for large-scale unconstrained binary quadratic problems. *Eur. J. Oper. Res.* 137(2):272–287.

Gopalakrishnan H, Kosanovic D (2015) Operational planning of combined heat and power plants through genetic algorithms for mixed 0-1 nonlinear programming. *Comput. Oper. Res.* 56: 51–67.

Gurobi (2020) Gurobi optimizer reference manual, https://www.gurobi.com/documentation/9.5/refman/index.html.

He H, Daume H III, Eisner JM (2014) Learning to search in branch and bound algorithms. *Adv. Neural Inform. Processing Systems* 27:3293–3301.

Hill JL (2011) Bayesian nonparametric modeling for causal inference. *J. Comput. Graphical Statist.* 20(1):217–240.

Holland PW (1986) Statistics and causal inference. *J. Amer. Statist. Assoc.* 81(396):945–960.

Howard SR, Pimentel SD (2021) The uniform general signed rank test and its design sensitivity. *Biometrika* 108(2):381–396.

Iacus SM, King G, Porro G (2011) Multivariate matching methods that are monotonic imbalance bounding. J. Amer. Statist. Assoc. 106(493):345–361.

Islam MS, Morshed MS, Young GJ, Noor-E-Alam M (2019) Robust policy evaluation from large-scale observational studies. *PLoS One.* 14(10):e0223360.

Lapczynski M, Bialowas S (2013) Discovering patterns of users' behaviour in an e-shop-comparison of consumer buying behaviours in Poland and other European countries. *Studia Ekonomiczne* 151:144–153.

Li C, Duan X, Lu L, Wang Q, Shen S (2019) Iterative algorithm for solving a class of convex feasibility problem. *J. Comput. Appl. Math.* 352:352–367.

Low RKY, Faff R, Aas K (2016) Enhancing mean–variance portfolio selection by modeling distributional asymmetries. *J. Econom. Bus.* 85:49–72.

Lucidi S, Rinaldi F (2010) Exact penalty functions for nonlinear integer programming problems. *J. Optim. Theory Appl.* 145(3): 479–488.

Martin A, Achterberg T, Koch T (2005) Branching rules revisited. *Oper. Res. Lett.* 33(1):342–354.

McIlvennan CK, Eapen ZJ, Allen LA (2015) Hospital readmissions reduction program. *Circulation* 131(20):1796–1803.

McNemar Q (1947) Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika* 12(2):153–157.

Morgan SL, Winship C (2015) *Counterfactuals and Causal Inference* (Cambridge University Press, Cambridge, UK).

Morshed MS, Islam MS, Noor-E-Alam M (2021) Sampling Kaczmarz-Motzkin method for linear feasibility problems: Generalization and acceleration. *Math. Programming* 194:1–61.

Morucci M, Noor-E-Alam M, Rudin C (2018) A robust approach to quantifying uncertainty in matching problems of causal inference. Preprint, submitted December 5, https://doi.org/10.48550/arXiv.1812.02227.

Murray W, Ng KM (2010) An algorithm for nonlinear optimization problems with binary variables. *Comput. Optim. Appl.* 47(2):257–288.

Necoara I, Richtárik P, Patrascu A (2019) Randomized projection methods for convex feasibility: Conditioning and convergence rates. *SIAM J. Optim.* 29(4):2814–2852.

Nikolaev AG, Jacobson SH, Cho WKT, Sauppe JJ, Sewell EC (2013) Balance optimization subset selection (boss): An alternative approach for causal inference with observational data. *Oper. Res.* 61(2):398–412.

Pitsoulis L, Pardalos PM (2009) Quadratic assignment problem. Floudas CA, Pardalos PM, eds. *Encyclopedia of Optimization* (Springer, New York), 3119–3149.

Rosenbaum PR (1989) Optimal matching for observational studies. *J. Amer. Statist. Assoc.* 84(408):1024–1032.

Rosenbaum PR (2002) *Observational Studies* (Springer, New York).

Rosenbaum PR, Rubin DB (1983) The central role of the propensity score in observational studies for causal effects. *Biometrika* 70(1): 41–55.

Rosenbaum PR, Rubin DB (1985) Constructing a control group using multivariate matched sampling methods that incorporate the propensity score. *Amer. Statistician* 39(1):33–38.

Rubin DB (1979) Using multivariate matched sampling and regression adjustment to control bias in observational studies. *J. Amer. Statist. Assoc.* 74(366):318–328.

Sauppe JJ, Jacobson SH (2017) The role of covariate balance in observational studies. Naval Res. Logist. 64(4):323–344.

Sauppe JJ, Jacobson SH, Sewell EC (2014) Complexity and approximation results for the balance optimization subset selection model for causal inference in observational studies. *INFORMS J. Comput.* 26(3):547–566.

Stuart EA (2010) Matching methods for causal inference: A review and a look forward. *Statist. Sci.* 25(1):1–21.

Yeh IC (1998) Modeling of strength of high-performance concrete using artificial neural networks. *Cement Concrete Res.* 28(12): 1797–1808.

Zhao X, Ng KF, Li C, Yao JC (2018) Linear regularity and linear convergence of projection-based methods for solving convex feasibility problems. *Appl. Math. Optim.* 78(3):613–641.

Zubizarreta JR (2012) Using mixed integer programming for matching in an observational study of kidney failure after surgery. *J. Amer. Statist. Assoc.* 107(500):1360–1371.

Zubizarreta JR (2015) Stable weights that balance covariates for estimation with incomplete outcome data. *J. Amer. Statist. Assoc.* 110(511):910–922.

Zubizarreta JR, Paredes RD, Rosenbaum PR (2014) Matching for balance, pairing for heterogeneity in an observational study of the effectiveness of for-profit and not-for-profit high schools in Chile. *Ann. Appl. Statist.* 8(1):204–231.