Optimization and Engineering (2021) 22:1419–1457
https://doi.org/10.1007/s11081-020-09536-5

RESEARCH ARTICLE

Inversion of convection–diffusion equation with discrete sources

Meenarli Sharma[1] · Mirko Hahn[2] · Sven Leyffer[3] · Lars Ruthotto[4] · Bart van Bloemen Waanders[5]

Received: 15 January 2020 / Revised: 18 June 2020 / Accepted: 7 July 2020 / Published online: 25 July 2020

Abstract

We present a convection–diffusion inverse problem that aims to identify an unknown number of sources and their locations. We model the sources using a binary function, and we show that the inverse problem can be formulated as a large-scale mixed-integer nonlinear optimization problem. We show empirically that current state-of-the-art mixed-integer solvers cannot solve this problem and that applying simple rounding heuristics to solutions of the relaxed problem can fail to identify the correct number and location of the sources. We develop two new rounding heuristics that exploit the value and a physical interpretation of the continuous relaxation solution, and we apply a steepest-descent improvement heuristic to obtain satisfactory solutions to both two- and three-dimensional inverse problems. We also provide the code used in our numerical experiments in open-source format.

Keywords Mixed-integer optimization · PDE-constrained optimization · Mixed-integer nonlinear programming · Source inversion

✉ Sven Leyffer
leyffer@anl.gov

Meenarli Sharma
meenarli@iitb.ac.in

Mirko Hahn
mirhahn@ovgu.de

Lars Ruthotto
lruthotto@emory.edu

Bart van Bloemen Waanders
bartv@sandia.gov

[1] Indian Institute of Technology Bombay, Powai, Maharashtra 400076, India

[2] Otto-von-Guericke Universität, Universitätsplatz 2, Magdeburg, Germany

[3] Argonne National Laboratory, 9700 South Cass Ave., Lemont, IL 50439, USA

[4] Emory University, 400 Dowman Drive, Atlanta, GA 30322, USA

[5] Sandia National Laboratories, P.O. Box 5800, Albuquerque, NM 87185, USA

Mathematics Subject Classification 65M22 · 90C10 · 90C30

1 Introduction

Many applications in science and engineering require the solution of an optimization problem where decision variables are both integral and continuous. For instance, the design of nuclear plants depends on selecting different types of core (fuel rod) configurations while controlling flow rates to maximize the heat extraction process Committee (2010). Remediating contaminated sites and maximizing oil recovery both involve flow through porous media to determine the number of wellbores, in addition to calculating optimal flow rates (Ozdogan 2004; Fipki and Celi 2008) and operational schedule (Bangerth et al. 2005, 2006; Bellout et al. 2012). Related applications also arise in optimally scheduling shale-gas recovery (Sharma 2013). Next-generation solar cells face complicated geometric and discrete design decisions to achieve effective electromagnetic performance (Reinke et al. 2011). In disaster-recovery scenarios, such as oil spills (You and Leyffer 2010, 2011), wildfires (Donovan and Rideout 2003; Fügenschuh et al. 2009), and hurricanes (Legg et al. 2013), resources need to be scheduled for mitigation purposes while predicting material properties and boundary conditions to calibrate the underlying dynamics for accurate forecasts. Many other science and engineering examples have similar decision-making characteristics, including wind farm design (Zhang et al. 2013) and the design, control, and operation of gas networks (De Wolf and Smeers 2000; Martin et al. 2006; Ehrhardt and Steinbach 2005; Steinbach 2007; Zavala 2014; Gugat et al. 2018; Hahn et al. 2017). The common theme of these applications is a need to address integral and continuous optimization variables in the context of large-scale multiphysics applications.

In this paper we consider an inverse problem in which the optimization variables are discrete and constrained by partial differential equations (PDEs), specifically in our case to describe convection–diffusion. We are interested in determining the number and location of a set of sources by reconciling the difference between measurements and numerical prediction of the concentration. Our work is motivated by applications in groundwater flow, where we want to find the location of pollutants in the subsurface; see, for example (Ozdogan 2004; Fipki and Celi 2008), for more detailed background.

The combination of discrete optimization variables and PDE constraints presents difficult problems, providing a range of new mathematical challenges; see, for example, Leyffer et al. (2013). These challenges arise out of the combination of the combinatorial complexity of integer variables and the computational difficulties of the discretized PDE. Little is known about this class of problems or solution approaches, and one motivation of this paper is to experiment with state-of-the-art mixed-integer solvers for solving this class of problems. Our solution strategy leverages concepts from topology optimization (Sigmund and Maute 2013b; Borzì and Schulz 2009). We chose the steady-state convection–diffusion equation as our model problem because it allows us to solve the resulting mixed-integer PDE-constrained optimization (MIPDECO) problems in a reasonable amount of time. We stress, however,

that our results and approaches generalize to time-dependent convection–diffusion processes.

We advance the state of the art in MIPDECO in a number of ways. First, we develop new rounding schemes that take the physics of the problem into account by preserving the mass of the source when we move from a relaxation to a rounded solution. Second, we apply a simplified version of the trust-region method (Hahn et al. 2020), and show that it already provides competitive integer solutions. Third, we improve the trust-region approach by developing a new problem-specific neighborhood that takes the topology of our problem into account, and we use a specialized knapsack solve for the resulting trust-region subproblem. Using the modified trust-region method we show that we can solve 3D instances of MIPDECO efficiently and in a reasonable amount of time, and we provide our Julia (Bezanson et al. 2012) code under a permissible open-source license. We also provide AMPL (Fourer et al. 1993) models of 2D instances to promote experimentation with existing mixed-integer programming (MIP) solvers.

In the next section, we provide background on MIP and PDECO that is relevant to our developments, and we discuss the challenges of MIPDECO in more detail. In Sect. 3, we describe the variational description of our model problem in terms of topology optimization, introduce our finite-element discretization, and discuss suitable regularization terms. In Sect. 4, we introduce problem-specific rounding schemes and a new trust-region approach. We also comment on how the resulting trust-region subproblem can be solved efficiently by formulating it as a knapsack problem. In Sect. 5 we give implementation details and describe our experimental setup. In Sect. 6 we present the results of our numerical experiments. We derive values for the regularization parameter and then show empirically that current state-of-the-art MINLP solvers cannot handle even two-dimensional instances of our MIPDECO on a realistic mesh. We also show that standard rounding schemes do not provide competitive solutions and may fail to identify sources. We demonstrate that our trust-region scheme can produce good solutions with moderate effort. In Sect. 7 we summarize our conclusions and briefly discuss avenues for future research.

2 Background

Mixed-integer PDE-constrained optimization brings together complicated elements from two algorithmic areas to solve relevant application problems. Because considerable development history is associated with each area, we provide only a general overview to highlight the most relevant features needed to introduce MIPDECO.

2.1 Mixed-integer programming

Nonlinear MIPs are a challenging class of problems in their own right: they are in general NP-hard (Kannan and Monma 1978) and in the worst case undecidable (Jeroslow 1973). Most nonlinear MIP methods use a tree search to resolve the integrality restrictions. We distinguish three basic classes of methods: branch-and-bound

or single-tree methods, multitree methods such as outer approximation, and hybrid techniques. Branch-and-bound (Dakin 1965; Gupta and Ravindran 1985) searches a tree where each node corresponds to a nonlinear subproblem. Branching corresponds to adding integer bounds on fractional integer variables that separate the fractional solution from the integer feasible set, creating two new nonlinear subproblems. Branch-and-bound methods can be improved by adding cutting planes (Stubbs and Mehrotra 1999; Akrotirianakis et al. 2001; Frangioni and Gentile 2006; Günlük and Linderoth 2008; Drewes 2009; Drewes and Ulbrich 2012; Çezik and Iyengar 2005) to tighten the continuous relaxations, resulting in a smaller tree that needs to be searched. Outer approximation (Duran and Grossmann 1986), Benders decomposition (Geoffrion 1972), and the extended cutting plane method (Still and Westerlund 2006) are multitree techniques. These methods define a lower-bounding linear MIP master problem that can be solved efficiently by using commercial solvers. The solution of the MIP master problem typically violates the nonlinear constraints, and new linearizations obtained from the solution of a nonlinear subproblem are added to the MIP master problem, resulting in an alternating sequence of linear MIP and nonlinear optimization subproblem. Hybrid methods (Quesada and Grossmann 1992; Abhishek et al. 2010; Bonami et al. 2008) combine nonlinear branch-and-bound with methods such as outer approximation and form the basis of the most efficient nonlinear MIP solvers (Abhishek et al. 2010; Bonami et al. 2008). The LP/NLP-based branch and bound method starts by solving a linear relaxation obtained by outer-approximating nonlinear constraints at the solution of the continuous relaxation of the problem. Whenever a new integer assignment is found, the linear tree-search is interrupted, and a nonlinear problem is solved obtained by fixing all integer variables to this assignment. The master problem is then updated by adding outer approximations from the solution of the nonlinear problem. More details can be found in the monographs (Floudas 2000; Tawarmalani and Sahinidis 2002), the collection (Lee and Leyffer 2011), and the survey papers (Burer and Letchford 2012; Grossmann and Kravanja 1997; Grossmann 2002; Belotti et al. 2013a; Bussieck and Pruessner 2003).

Adding PDE-constraints leads to a range of computational and conceptual challenges for MINLP. First, the computational expense of solving PDE-constrained optimization problems is much higher than the computational expense of solving standard NLP. Second, the number of optimization variables is typically very large, and if the integer variables are functions defined over the computational domain, then the number of integer variables typically also grows as we refine the discretization, resulting in huge combinatorial search spaces. Third, the solutions of the relaxations of the PDE-constrained optimization problem are typically only locally optimal and do not provide valid lower bounds for nonlinear PDE-constrained optimization problems.

2.2 PDE-constrained optimization

PDE-constrained optimization (PDECO) refers to the optimization of systems governed by partial differential equations. In most cases the goal is to optimize

an objective function with respect to a quantity that is defined on subregions or everywhere in the computational domain. The inversion for initial conditions and the reconstruction of material properties are examples of typical optimization problems.

The PDE-constrained optimization problem is an infinite-dimensional optimization problem, and two approaches exist for obtaining a finite-dimensional approximation: the optimize-then-discretize approach and the discretize-then-optimize approach. In the former, one finds the necessary optimality conditions of the PDECO in function spaces and then discretizes this system of equations. In the latter, one first discretizes the PDE and then uses nonlinear optimization techniques to solve the large-scale optimization problem. Both approaches lead to an optimization problem with a large number of variables and a large system of equations (the discretized PDEs) that describe the underlying physics. The large-scale nature of these problems dictates the use of efficient sensitivities (adjoints), Newton-based methods to handle the nonlinearity of the optimization formulation, the coordination of globalization, and the use of parallel matrix-vector operators to address the computational requirements (Nocedal and Wright 2000; Biegler et al. 2001, 2007). The combination of these technologies poses formidable challenges to achieve efficient and accurate solutions. Considerable research and development have been conducted; the interested reader is referred to Vogel (1999), Ascher and Haber (2001), Haber and Ascher (2001), Vogel (2002), Laird et al. (2005), Hintermuller and Vicente (2005), Hazra and Schulz (2006), Borzi (2007), Hinze et al. (2009). Advances have been made to accelerate the convergence of these algorithms, with recent examples in special preconditioners, reduced-space methods, full-space algorithms, and multigrid approaches (Biros and Ghattas 2005a, b; Akcelik et al. 2005; Bartlett et al. 2005; Heinkenschloss and Ridzal 2008; Borzì and Schulz 2009; Simon 2008).

A full-space solution algorithm forms the Lagrangian function and takes variations with respect to the state variables, the adjoint variables, and the optimization variables. This approach results in the first-order optimality conditions, which form a nonlinear system of equations. This system is typically solved by using Newton's method, requiring the solution of large structured systems of equations.

Alternatively, a reduced-space method eliminates the state variables by using the discretized PDE, resulting in an optimization problem in the optimization or control variables only, where the effect of the states in represented implicitly. The gradient of the objective function can be computed via the chain rule; and the solution process consists of an iterative process in which forward, adjoint, and gradient equations are successively solved. Even though there are advantages to using the full-space methods, in particular when the solution of the forward solve is slow to converge, we use the reduced-space method here because it has advantages in the mixed-integer case, resulting in an easy-to-solve subproblem.

PDE-constrained optimization problems with integer optimization variables have been solved with PDE-constrained optimization methods. Topological optimization is an example where the discrete optimization variables are approximated with continuous variables with the unfortunate consequence of errors (nonintegral values, or gray areas) at the boundaries of the topological solution. Although excellent practical results are obtained for a host of problems, establishing rigorous optimality

proofs for this approach remains a challenge (Bendsøe and Sigmund 2004; Sigmund and Maute 2013a).

2.3 MIPDECO

Practical approaches to MIPDECO must tackle the challenges posed by the number of integer variables in the discretized problem and the computational complexity of solving PDECO problems. We briefly discuss how recent approaches to MIPDECOs tackle these challenges.

The two classical approaches for solving PDECO problems are optimize-then-discretize and discretize-then-optimize. We do not believe that it is possible to apply the optimize-then-discretize to obtain first-order conditions for MIPDECOs, because such a generalization would also imply a set of first-order conditions for the (global) optimality of integer solutions for MINLPs, which seems unlikely. Hence, we consider only the discretize-then-optimize approach as a practical way to solve MIPDECOs.

If the integer controls are functions over the computational domain, then the discretization of the PDE and the controls results in MINLPs with a large number of integer variables, which has implications for standard MINLP solvers. Current state-of-the-art solution methods for MINLP employ a branch-and-bound tree search (Belotti et al. 2013b; Bonami et al. 2008) at some stage of the solution process and the large number of integer variables arising in discretized MIPDECOs means that this tree can become huge, even for coarse discretization levels. In some cases, the tree search can be customized for solving discretized MIPDECO by using problem specific branching rules; see, for example, (Hahn et al. 2017), which introduces a new branching rule and backtracking strategy that work well for time-dependent control problems.

Another solution approach for discretized MIPDECO is penalty-based methods, which avoid the branch-and-bound tree altogether by penalizing the violation of integrality. The resulting nonlinear optimization problem is solved iteratively for an increasing penalty parameter, until all integer variables are integral. Unfortunately, the penalty is in general nonconvex, and stationary points of the nonlinear optimization problem correspond only to feasible points of the discretized MIPDECO without any optimality guarantees. See (Costa et al. 2016; Lucidi and Rinaldi 2013; Garmatter et al. 2019) for penalty-based methods in the context of MINLP and MIPDECO.

There are other notable solution approaches for MIPDECOs that avoid a tree search, including decomposition methods that solve a relaxed form of the original problem and approximate the effect of the relaxed optimal control with that of an integer-valued one. For ODE- and DAE-constrained problems, heuristics such as sum-up rounding (SUR) (Sager 2006, 2009) and next-forced rounding (NFR) (Jung 2013) can produce arbitrarily good approximations of the optimal relaxed behavior given sufficiently high grid resolutions (Gerdts and Sager 2012; Sager et al. 2012). These heuristics are special cases of the combinatorial integral approximation (CIA) approach (Sager et al. 2011), which formulates the approximation problem as an

MILP. Efficient problem-specific tree-based solvers have also been developed to solve the CIA problem directly (Bürger et al. 2020; Jung et al. 2015).

Early extensions of CIA heuristics to PDE-constrained problems (Hante and Sager 2013; Hante 2017) limited themselves to rounding purely time-distributed controls where the arrow of time can be exploited as part of rounding heuristics such as SUR and NFR. More recent results, however, have shown that SUR can be applied to elliptic PDEs with spatially distributed controls by imposing an order through space-filling Hilbert curves (Manns and Kirches 2018, 2019a, b), though other orders have also been used successfully (Yu and Anitescu 2019). Rounding methods for topology optimization problems are also explored in (Garmatter et al. 2019).

These methods are collectively based on the recognition that spatially distributed integer-valued controls are not truly discrete, but form a continuum. This is explored in Hahn et al. (2020), where a trust-region steepest-descent method for binary optimal control is developed that is closely related to our approach. The authors show convergence to first-order stationarity in a topological sense, provided that the mesh is refined. This is a remarkable result because it replaces the combinatorial challenge of MIPDECO by a set-based approach. This method avoids the combinatorial complexity of the tree search and instead solves a sequence of knapsack problems that can be interpreted as a local improvement strategy. A similar model to ours has been studied in Guo et al. (2019), where pollution sources are identified and a genetic algorithm heuristic is employed to resolve the integrality restrictions.

3 Mathematical formulation of the model

We formulate the constrained source inversion problem as a mixed-integer PDE-constrained optimization problem with binary inversion parameters. We discretize the PDEs with finite elements and present the resulting finite-dimensional formulation. We also present an alternative finite-difference discretization that provides self-contained AMPL models to run state-of-the-art MINLP solvers.

3.1 Variational formulation of the source inversion problem

The goal of the inverse problem is to estimate the source w from measurements $\mathbf{b}$ assuming that the properties of the PDE are known and assuming a sparse set of sensors. The problem can be written as follows:

$$\begin{cases} \underset{u,w}{\text{minimize}} & \mathcal{J}(u,w) = \dfrac{1}{2\sigma}\displaystyle\sum_{i=1}^{m} \left\| \langle p(r_i), u\rangle - \mathbf{b}_i \right\|^2 + \alpha \mathcal{R}(w) & \\ \text{subject to} & -c\Delta u + v^\top \nabla u = w, & \text{in } \Omega \\ & \dfrac{\partial u}{\partial n} = 0, & \text{on } \Gamma_N \\ & u = g, & \text{on } \Gamma_D, \end{cases} \tag{1}$$

where $c > 0$ is the diffusion coefficient, $v : \Omega \to \mathbb{R}^d$ is the velocity vector, and $w : \Omega \to \{0, 1\}$ represents the source terms. The boundary of the domain, Γ, is

partitioned into Γ_N and Γ_D for Neumann and Dirichlet conditions, respectively, $n : \Gamma \to \mathbb{R}^d$ denotes the outward normal vector; and $g : \Gamma_D \to \mathbb{R}$ defines the Dirichlet condition. Let $\Omega \subset \mathbb{R}^d$ denote the computational domain, where in this work $d = 2, 3$. Discrete measurements $\mathbf{b} \in \mathbb{R}^m$ are given as

$$\mathbf{b}_i = \langle p(r_i), u \rangle + \boldsymbol{\epsilon}_i, \quad i = 1, 2, \ldots, m, \tag{2}$$

where u is the concentration of the pollutant, $p(r_i) : \Omega \to \mathbb{R}$ are the receiver functions at the locations r_i for $i = 1, \ldots, m$, $\langle \cdot, \cdot \rangle$ denotes the $\mathcal{L}_2$ inner product, and $\boldsymbol{\epsilon}_1, \boldsymbol{\epsilon}_2, \ldots, \boldsymbol{\epsilon}_m$ represent measurement noise. To model point measurements of the PDEs, we consider the receiver function $p(r)$ to be a Dirac δ-function centered at r. In our numerical demonstrations, we generate the sensor data synthetically and assume that the measurement noise is independent and identically distributed (iid) Gaussian noise with constant, known standard deviation $\sigma > 0$.

$\mathcal{R}$ is a regularization functional that promotes the existence and regularity of solutions. This is important because the inverse problem is underdetermined and ill-conditioned as a result of data sparsity and noise. The term $\mathcal{R}$ can also be used to penalize undesirable features. The regularization parameter $\alpha > 0$ balances between fitting the data (for small values of α) and ensuring regularity of the solution (for large values of α). Choosing an "optimal" α, is both crucial and nontrivial. No general rule exists for picking α; however, strategies using generalized cross validation (Golub et al. 1979; Haber and Oldenburg 2000), discrepancy principle (Engl et al. 1996), and L-curve (Hansen 1998) are commonly used. In practice, any of these methods will require us to approximately solve (1) for a set of regularization parameters. Because of the binary constraints, the source w will be continuous only in trivial cases. In general, we expect piecewise constant solutions with finite edge measure. This guides our choice of the regularization function and motivates us to use the total variation (TV) semi-norm. With this choice, the solution of the relaxed problem will have bounded variation (Chan and Shen 2005; Rudin et al. 1992; Vogel 2002). In our numerical experiments, we formally write the total variation regularizer as

$$\mathcal{R}(w) = \int_\Omega \|\nabla w(x)\|_2 dx, \tag{3}$$

which is well-defined for all $w : \Omega \to [0, 1]$ with bounded variation; for non-differentiable functions (e.g., binary-valued functions) $\mathcal{R}(w)$ can be computed using the co-area formula (Scherzer et al. 2013, Thm 9.75). This regularizer is *isotropic*, which means that its value does not depend on the orientation the source w. In other words, its value is invariant to rotations of the coordinate system. This is an advantage over the also commonly used *anisotropic* version of TV (obtained by replacing the Euclidean norm with the ℓ_1-norm in (3)), which is sensitive to rotations of the domain. Problem (1) is not a quadratic program, because the TV regularization term involves a square-root. It is easy to show that (3) is second-order-cone representable. However, we do not exploit this fact in our results.

3.2 Finite-dimensional approximations of the source inversion problem

In the following, we briefly outline a finite-element discretization of the PDE and boundary condition and comment on the structure these discretizations imply for the finite-dimensional MINLPs. To make the mathematics as well as the implementation more accessible, we include additional details in "Appendix 1". For ease of presentation, we assume a rectangular domain $\Omega = [0, 1]^d$.

The finite-element approach partitions the domain Ω into a mesh Ω_N containing N^d congruent elements (pixels or voxels; see an additional explanation in "Appendix 1") in $d = 2$ and $d = 3$, respectively. We note however that a strength of the finite-element method is that it extends easily to a wide class of non-rectangular domains. We then approximate the concentration u in a finite-dimensional subspace consisting of globally continuous and piecewise bi/trilinear functions on Ω_N. Similarly, we approximate the sources w in the space of all piecewise constant functions. This approach allows us to replace u and w by finite-dimensional vectors, $\mathbf{u} \in \mathbb{R}^{(N+1)^d}$ and $\mathbf{w} \in \{0, 1\}^{N^d}$, respectively, where $\mathbf{u}_i$ corresponds to the value of u at node i of the finite-element mesh and $\mathbf{w}_i$ is the value of w inside element i. Even though the basis functions are nonlinear, the expansion of u and w is *linear* in the coefficients of the basis functions, and results in a set of linear equality constraints, leading to the finite-dimensional PDE constraint

$$\mathbf{Su} = \mathbf{Mw}, \tag{4}$$

where $\mathbf{S}$ is the stiffness matrix and $\mathbf{M}$ is the mass matrix, whose entries are obtained by integrating the weak form of the PDE constraint in (1); see (23) for details. Because of the compact support of the Ansatz functions for u and w, both matrices are sparse. The discrete objective function is obtained after discretizing the receiver functions and gradient operators on the finite-element mesh. To discretize the inner products in (2), we use a midpoint rule. The resulting finite-dimensional convex MINLP is

$$\underset{\mathbf{u},\mathbf{w}}{\text{minimize}} \quad \frac{1}{2\sigma}\|\mathbf{Pu} - \mathbf{b}\|^2 + \alpha R(\mathbf{w}) \tag{5a}$$

$$\text{subject to} \quad \begin{aligned} &\mathbf{Su} = \mathbf{Mw} \\ &\mathbf{w} \in \{0, 1\}^{N^d}, \end{aligned} \tag{5b}$$

which is a finite-dimensional MINLP with a convex objective and linear constraints. Here, the matrix $\mathbf{P}$ is the interpolation of the states to the point-measurement locations. In particular, the ith row of the matrix $\mathbf{P} \in \mathbb{R}^{m\times(N+1)^d}$ contains the discretization of the receiver function $p(r_i)$ on the mesh. In our experiments below, we assume point measurements at the locations $r_1, \ldots, r_m$ and use a bi/trilinear spline interpolation to obtain the PDE solutions at those points. Hence, each row of $\mathbf{P}$ contains only four/eight nonzero elements that correspond to the interpolation weights. The discretized regularizer, $R(\mathbf{w})$, is derived in (27).

We note that the MINLP (5) contains special structure that is not typically present in standard MINLPs, and we exploit this structure in the solution of the

problem. Because the stiffness matrix, $\mathbf{S}$, is nonsingular by design, we can solve (4) uniquely for $\mathbf{u}$, given any choice of the discretized controls, $\mathbf{w}$. Formally, we obtain $\mathbf{u} = \mathbf{S}^{-1}\mathbf{M}\mathbf{w}$ and eliminate $\mathbf{u}$, resulting in a pure integer nonlinear program over the controls, $\mathbf{w}$, only. This corresponds to a reduced-space approach. The feasibility of this approach hinges on an efficient way to solve the PDE (operate with $\mathbf{S}^{-1}$). For some problems, one may be able to obtain a factorization of the stiffness matrix and reuse it to evaluate the reduced-space objective and gradients. For large-scale problems the reduced-space approach may still be feasible if an effective iterative method is available. This approach leads to the discretized MIPDECO

$$\underset{\mathbf{w}}{\text{minimize}} \quad J(\mathbf{w}) = \frac{1}{2\sigma}\left\|\mathbf{P}\mathbf{S}^{-1}\mathbf{M}\mathbf{w} - \mathbf{b}\right\|^2 + \alpha R(\mathbf{w}) \tag{6a}$$

$$\text{subject to} \quad \mathbf{w} \in \{0,1\}^{N^d}. \tag{6b}$$

We note that the regularization term in problem (6) can be reformulated so that (6) becomes a mixed-integer second-order cone problem. The relaxation of (6) is

$$\underset{\mathbf{w}}{\text{minimize}} \quad J(\mathbf{w}) = \frac{1}{2\sigma}\left\|\mathbf{P}\mathbf{S}^{-1}\mathbf{M}\mathbf{w} - \mathbf{b}\right\|^2 + \alpha R(\mathbf{w}) \tag{7a}$$

$$\text{subject to} \quad \mathbf{w} \in [0,1]^{N^d}, \tag{7b}$$

which we solve using a projected Gauss-Newton algorithm.

Remark 1 (Structure of the MINLP (6))

1. The reduced-space approach presented here can be generalized to nonlinear PDEs. In this case, however, the solution operator of the PDE ($\mathbf{S}^{-1}$ in the linear case) is no longer a linear operator. This observation implies that we can no longer reuse the factors of $\mathbf{S}$ and instead must "reinvert" the solution for every iterate in a quasi-Newton process; see (Biros and Ghattas 2005a, b; Akçelik et al. 2006) on how the reduced-space methods are applied to nonlinear PDEs. The iterative process consists of a linearization of the optimality conditions, a gradient calculation with adjoint-based sensitivities, which consists of a linear system solve with $\mathbf{S}^T$ and a linearized objective function as a right hand side.
2. The reduced-space objective function in (6) will typically have a dense Hessian matrix, even if $\mathbf{S}$ is sparse, and this can cause computational difficulties for MINLP solvers as we increase the mesh size.
3. In general, it is difficult to obtain closed-form expressions for the coefficients of the matrices $\mathbf{S}$ and $\mathbf{M}$, making it cumbersome to state these equations in a self-contained model.

The last point motivates us to present an alternative finite-difference discretization that provides closed-form expressions for the coefficients of the discretized PDE to facilitate the reproducibility of our experiments; see "Appendix 2". The

finite-difference discretization again results in a MINLP with linear constraints and a convex objective function. As before, we can eliminate the state variables using the discretized PDE and boundary conditions.

In the remainder of this paper, we present our integrated rounding and trust-region heuristic and our numerical results. We note that the rounding and trust-region schemes are agnostic to the discretization scheme.

4 An integrated rounding and trust-region heuristic

We show in our numerical results that standard MINLP methods cannot solve the discretized MIPDECOs from the preceding section within a reasonable amount of time, even for coarse discretization levels, because the search tree becomes too large and the subproblems at every node take too much time to solve. Hence, one must consider heuristic techniques. We present a new 'two-phase heuristic' for MIPDECO. In the first phase, we deploy a problem-specific rounding scheme whose solution is passed as an initial guess to the next phase. The second phase is an 'improvement heuristic' that is motivated by trust-region methods for nonlinear optimization; see, for example, (Conn et al. 2000), as well as local-branching heuristics for MINLP (Fischetti and Lodi 2002; Nannicini et al. 2008). We start our approach by first solving the continuous relaxation, which is then rounded using different heuristics. In Sect. 6.3, we numerically illustrate that starting from various rounded initial points, we arrive at different final solutions due to the different trust-region subproblems obtained at these initial solutions. Other heuristics that have been proposed for MINLPs are large neighborhood search (Danna et al. 2005) and feasibility pump (Fischetti et al. 2005; Bonami et al. 2009). However, we do not believe that the latter is practical for MIPDECOs because it would require factorizations and rank-one updates of the basis matrices involving the discretized PDE, which may be prohibitive or even impossible for small mesh sizes.

Our heuristic is agnostic to the discretization of the PDE or to the solution of the continuous relaxation. Hence, we assume in the remainder that the control variables, $\mathbf{w}_i$, either represent the values of the control in element i from the finite-element discretization of Sect. 3.2 or represent a lexicographical ordering of the cell-centered controls, $\mathbf{W}_{kl}$, in the finite-difference discretization (see "Appendix 2").

4.1 Rounding schemes for MIPDECO

The improvement heuristic used in our proposed method requires a starting solution. To obtain such an integer feasible solution we present several rounding based schemes in this section. Our results show that the naïve/standard rounding (using a cut-off of 0.5) could fail to identify some sources. On the other hand, the existing NLP-based rounding heuristics from the literature (Sigmund and Maute 2013b) does not produce competitive solutions in the sense that our trust-region method can improve these solutions objective value by around 24%. Hence, we propose two new rounding schemes to obtain an integer feasible solution, starting from the optimal

solution of the continuous relaxation, which can be used as a starting solution for our heuristics. The first scheme takes the objective function into account while rounding, and the second aims to preserve the mass of the sources; that is, it tries to keep $\int_\Omega w\,dx$ invariant. For both proposed heuristics, we let $\widetilde{\mathbf{w}} \in [0,1]^{N_d}$ be the solution of the relaxation (7) or (29).

First we briefly discuss existing NLP-based heuristics followed by the proposed rounding schemes: objective-gap-reduction and mass-preserving.

Penalization-Based NLP Heuristics We implemented a penalty-based rounding scheme from the literature (Sigmund and Maute 2013b) in which we relax the integrality restrictions and instead solve a sequence of penalized NLPs for an increasing value of penalty parameter to drive the integrality gap to zero. In particular, we add a penalty term to the objective of (7) and (29), respectively, resulting in the following (nonconvex) penalized formulation of (7) (the approach for (29) is similar):

$$\begin{aligned} \underset{\mathbf{w}}{\text{minimize}} \quad & \frac{1}{2\sigma}\left\|\mathbf{P}\mathbf{S}^{-1}\mathbf{M}\mathbf{w}-\mathbf{b}\right\|^2 + \alpha R(\mathbf{w}) + \beta \sum_{i=1}^{N^d} \left(\mathbf{w}_i\left(1-\mathbf{w}_i\right)\right)^q \\ \text{subject to} \quad & \mathbf{w} \in [0,1]^{N^d}, \end{aligned} \tag{8}$$

where q is a positive integer and β is a penalty parameter that we increase until the integrality gap is sufficiently small. In our implementation, we use $q = 1$. We solve the continuous relaxation with $\beta = 0$, set $\beta = 10^{-6}$, and increase β by a factor 2 until the integrality gap, $\max_i \left\{\min\{\mathbf{w}_i, 1-\mathbf{w}_i\}\right\}$, is sufficiently small ($\leq \epsilon := 10^{-4}$), and then round the final $\mathbf{w}$ to its nearest integer. We summarize this approach in Algorithm 1.

Let $\mathbf{w}^{(0)}$ be a solution of (8) with $\beta = 0$;
Choose integrality gap $\epsilon > 0$, iteration limit, $K_{\max}$, set $k := 0$, and $\beta_0 := \beta_{\min} > 0$;
while $k < K_{\max}$ *and* $\max_i \left\{\min\{\mathbf{w}_i^{(k)}, 1-\mathbf{w}_i^{(k)}\}\right\} > \epsilon$ **do**
  Let $\mathbf{w}^{(k+1)}$ solve the penalized relaxation (8) with $\beta = \beta_k$;
  Set $\beta_{k+1} := 2 \cdot \beta_k$ and $k := k+1$;
return *Rounded* $\mathbf{w}^{(k)} \in \{0,1\}$;

Algorithm 1: Penalization-Based NLP Heuristic for FEM Discretization.

Objective-gap-reduction rounding Given $\widetilde{\mathbf{w}} \in [0,1]^{N_d}$, the first scheme selects a cut-off value t such that the resulting rounded solution defined as

$$\overline{\mathbf{w}}_i(t) = \begin{cases} 1, & \text{if } \widetilde{\mathbf{w}}_i \geq t \\ 0, & \text{otherwise,} \end{cases} \quad \forall i = 1, \ldots, N_d \tag{9}$$

is as close as possible to the relaxation solution in terms of its objective value. Mathematically, the desired cut-off value, t, is the minimizer of the following optimization problem:

$$\underset{0\leq t\leq 1}{\text{minimize}}\ J(\mathbf{w}(t)) - J(\widetilde{\mathbf{w}}) \quad \Leftrightarrow \quad \underset{0\leq t\leq 1}{\text{minimize}}\ J(\mathbf{w}(t)). \tag{10}$$

Consequently, we call this scheme *objective gap-reduction rounding*. The optimization problem in (10) can be written as a convex MINLP and the upper bound on t can be tightened to $\max_{i=1,\ldots,N_d} \widetilde{\mathbf{w}}_i$. Because this problem is hard to solve, we propose a simple iterative algorithm to obtain an acceptable cut-off value for the rounding (see Algorithm 2), because we are interested only in the approximate solution of (10). The process starts by iteratively increasing t by a constant step $T \in (0, 1)$ from a small initial value until t exceeds $t_{\max} = \max_{i=1,\ldots,N_d} \widetilde{\mathbf{w}}_i$ and outputs the best cut-off value t^*. When $t^* = 0.5$, this scheme is the same as naïve rounding.

Let $T \in (0, 1)$;
Set $k := 0$, $t_k := \min_{i=1,\ldots,N_d} \widetilde{\mathbf{w}}_i$, $t^* := 0$, $J^* := \infty$, and $t_{\max} := \max_{i=1,\ldots,N_d} \widetilde{\mathbf{w}}_i$;
while $t_k \leqslant t_{\max}$ **do**
  Form $\mathbf{w}(t_k)$;
  Solve the discretized PDE (6a) with $\mathbf{w} = \mathbf{w}(t_k)$ and evaluate $J(\mathbf{w}(t_k))$;
  if $J(\mathbf{w}(t_k)) < J^*$ **then**
    Set $J^* := J(\mathbf{w}(c_k))$ and $t^* := t_k$;
  Set $k := k + 1$ and $t_k := t_{k-1} + T$;
Output: The best cut-off value t^* and the rounded solution, $\overline{\mathbf{w}} = \mathbf{w}(t^*)$;

Algorithm 2: Objective-gap-reduction rounding.

Mass-Preserving Rounding. The scheme rounds the relaxation solution while preserving the mass of the sources in the relaxation solution as much as possible. The mass of the sources in the relaxation solution is given by $\widetilde{S} = \sum_{i=1}^{N_d} \widetilde{\mathbf{w}}_i$. Let $\bar{S} = \lceil \widetilde{S} \rceil$ be the nearest integer to $\widetilde{S}$. To compute the rounded solution $\overline{\mathbf{w}}$, we first arrange the components of $\mathbf{w}$ in decreasing order of $\widetilde{\mathbf{w}}_i$ values:

$$1 \geq \widetilde{\mathbf{w}}_{i_1} \geq \widetilde{\mathbf{w}}_{i_2} \geq \ldots \geq \widetilde{\mathbf{w}}_{i_{N_d}} \geq 0.$$

Next, the largest $\bar{S}$ entries are set to 1, and the remaining entries are set to zero. The resulting rounded solution is

$$\overline{\mathbf{w}}_{i_k} = \begin{cases} 1, & k = 1, \ldots, \bar{S}, \\ 0, & k = \bar{S} + 1, \ldots, N_d. \end{cases} \tag{11}$$

It follows that the difference in the mass of the sources between the rounded and the relaxed solutions is less than 1. Unlike the first rounding scheme, this rounding scheme does not require the solution of any additional PDEs once the relaxed problem (7) has been solved.

4.2 Trust-region-based improvement heuristic

Our trust-region-based heuristic for solving (6) starts from a binary vector, $\mathbf{w}^{(0)} \in \{0, 1\}^{N^d}$, and iterates on the binary variables, $\mathbf{w}$. Here $\mathbf{w}^{(0)} = \overline{\mathbf{w}}$, an integer feasible solution from a rounding scheme in the first phase. At iteration k, we assume that we have solved the discretized PDE in (6) with fixed binary vector $\mathbf{w}^{(k)} \in \{0, 1\}^{N^d}$ and have evaluated the objective function and its adjoint; see, for

example, (Biegler et al. 2003; Akçelik et al. 2006), with respect to the (relaxation of the) binary variables,

$$J^{(k)} := J(\mathbf{u}^{(k)}, \mathbf{w}^{(k)}) \quad \text{and} \quad J'^{(k)} := \nabla_{\mathbf{w}} J(\mathbf{u}^{(k)}, \mathbf{w}^{(k)}).$$

We then define the trust-region subproblem that aims to find an improved point, $\widehat{\mathbf{w}}$:

$$\begin{array}{ll} \underset{\mathbf{w}}{\text{minimize}} & J^{(k)} + J'^{(k)^T}(\mathbf{w} - \mathbf{w}^{(k)}) \\ \text{subject to} & \left\|\mathbf{w} - \mathbf{w}^{(k)}\right\|_1 \le \Delta_k, \quad \mathbf{w} \in \{0,1\}^{N^d}, \end{array} \tag{12}$$

where $\Delta_k > 0$ is the trust-region radius and $\Delta_k \in \mathbb{Z}$ is the maximum number of components of $\mathbf{w}$ that can flip from 0 to 1 or 1 to 0 during an iteration. It is well known that we can rewrite the ℓ_1 trust-region constraint of (12) equivalently as a knapsack constraint:

$$\begin{aligned} &\sum_{i:\mathbf{w}_i^{(k)}=0} \mathbf{w}_i + \sum_{i:\mathbf{w}_i^{(k)}=1} (1 - \mathbf{w}_i) \le \Delta_k \\ &\Leftrightarrow \sum_{i:\mathbf{w}_i^{(k)}=0} \mathbf{w}_i - \sum_{i:\mathbf{w}_i^{(k)}=1} \mathbf{w}_i \le \Delta_k - \left|\left\{ i : \mathbf{w}_i^{(k)} = 1 \right\}\right|, \end{aligned}$$

because $\mathbf{w}^{(k)} \in \{0,1\}^{N^d}$. This reformulation is the motivation for using the ℓ_1, rather than the ℓ_2 trust-region, because it results in an easier to solve trust-region subproblem.

We also introduce an alternative trust-region subproblem that, in addition to the ℓ_1 trust-region constraint of (12), restricts the changes in $\mathbf{w}$ to components that are close to current source locations. In particular, we let $\theta > 0$ be a bound on the topological distance from the current solution, and we define the center of $C(\mathbf{w}_i)$ as the coordinates of the center of the element or cell, i, corresponding to $\mathbf{w}_i$. We define the topological θ-neighborhood of the current iterate $\mathbf{w}^{(k)}$ as

$$\mathcal{N}_\theta(\mathbf{w}^{(k)}) := \left\{ i \ : \ \exists j \text{ with } \left\| C(\mathbf{w}_i) - C(\mathbf{w}_j^{(k)}) \right\|_2 \le \theta \text{ and } \mathbf{w}_j^{(k)} = 1 \right\}, \tag{13}$$

which defines an index set of the finite elements whose centroid is within a distance θ to the centroids of the current estimate of the source, $\mathbf{w}_j^{(k)} = 1$. Our alternative trust-region subproblem is then defined as the following problem in which the binary variables that lie outside the neighborhood, $\mathcal{N}_\theta(\mathbf{w}^{(k)})$, are fixed at their current values, 0:

$$\begin{array}{ll} \underset{\mathbf{w}}{\text{minimize}} & J^{(k)} + J'^{(k)^T}(\mathbf{w} - \mathbf{w}^{(k)}) \\ \text{subject to} & \left\|\mathbf{w} - \mathbf{w}^{(k)}\right\|_1 \le \Delta_k, \\ & \mathbf{w}_i = 0, \forall i \notin \mathcal{N}_\theta(\mathbf{w}^{(k)}), \ \mathbf{w} \in \{0,1\}^{N^d}. \end{array} \tag{14}$$

Unlike (12), this problem takes the topology of the current iterate into account when defining the trust-region subproblem, because values of $\mathbf{w}_i$ that are far from the

current sources are fixed at zero. Given either of these trust-region subproblems, we now state our improvement heuristic in Algorithm 3.

Set the initial trust-region radius, $\Delta_0 \in \mathbb{Z}_+$, and let $\mathbf{w}^{(0)} \in \{0,1\}^{N^d}$;
Choose constant $0 < \gamma < 1$, and set $k := 0$;
while $\Delta_k \geqslant 1$ **do**
  Solve subproblem (12) or (14) for $\hat{\mathbf{w}} := \operatorname{argmin}$(12) or (14);
  Solve the PDE for $\hat{\mathbf{u}} := \mathbf{u}(\hat{\mathbf{w}})$ and evaluate the objective $J(\hat{\mathbf{w}})$;
  Compute the ratio of actual over predicted reduction:

$$\rho_k := \frac{J(\mathbf{w}^{(k)}) - J(\hat{\mathbf{w}})}{-J'^{(k)^T}\left(\hat{\mathbf{w}} - \mathbf{w}^{(k)}\right)}$$

  if $\rho_k > \gamma$ **then**
    Accept the new point: $\mathbf{w}^{(k+1)} = \hat{\mathbf{w}}$, solve the adjoint PDE to get $J'^{(k+1)}$;
    if $\|\mathbf{w}^{(k+1)} - \mathbf{w}^{(k)}\|_1 = \Delta_k$ **then**
      Increase the trust-region radius $\Delta_{k+1} := 2\Delta_k$;
  else if $\gamma \geqslant \rho_k > 0$ **then**
    Accept the new point $\mathbf{w}^{(k+1)} := \hat{\mathbf{w}}$, solve the adjoint PDE to get $J'^{(k+1)}$;
    Leave the trust-region unchanged $\Delta_{k+1} = \Delta_k$;
  else
    Reject the new point, set $\mathbf{w}^{(k+1)} := \mathbf{w}^{(k)}$, and set $J'^{(k+1)} := J'^{(k)}$;
    Reduce the trust region $\Delta_{k+1} = \lfloor \Delta_k/2 \rfloor$;
  Set $k := k + 1$;

Algorithm 3: Trust-Region-Based Improvement Heuristic.

In Algorithm 3, we increase the trust-region radius if we observe good agreement (as measured by ρ_k) between the objective function in (6) and its linear approximation in (12). If the two do not agree, then we reduce the trust-region radius in the hope of getting better agreement in a smaller region. We use the ℓ_1-norm trust-region, because it corresponds to the Hamming distance for binary vectors, and the trust-region radius can be interpreted as limiting the number of binary variables that can change from the current iterate $\mathbf{w}^{(k)}$. We also ensure that the trust-region radius is always an integer, and the algorithm stops, once the radius becomes zero. The trust-region radius becomes zero after a finite number of iterations, because the objective is bounded below by zero, there exists a finite number of integer assignments, $\mathbf{w}$, and the trust-region reduction uses the floor operator.

4.3 Solving the trust-region subproblems

The trust-region subproblems (12) and (14) are linear binary optimization problems with a single constraint. One can easily see that as long as the trust-region radius is non-negative (i.e., as long as $\Delta_k \geq 0$), the problem is feasible. Here we show that this binary optimization problem can be solved to optimality efficiently by observing that it can be reduced to a special knapsack problem for which efficient solution methods exist.

We start by writing the trust-region subproblem as a generic binary linear program,

$$\underset{\mathbf{w}}{\text{minimize}} \quad \mathbf{g}^T\mathbf{w} \quad \text{subject to} \quad \mathbf{a}^T\mathbf{w} \leq b,\ \mathbf{w} \in \{0,1\}^p, \tag{15}$$

where $\mathbf{g} \in \mathbb{R}^p$ is the gradient, $\mathbf{a} \in \{-1,1\}^p$, and $b \geq 0$ is a positive integer.

To extend the knapsack solution approach to our problem, we distinguish the following cases:

1. $\mathbf{a}_i = 1$ and $\mathbf{g}_i > 0$ implies that $\mathbf{w}_i = 0$ at a solution of (15) (because increasing $\mathbf{w}_i$ deteriorates both the objective and the constraint satisfaction).
2. $\mathbf{a}_i = -1$ and $\mathbf{g}_i < 0$ implies that $\mathbf{w}_i = 1$ at a solution of (15) (because decreasing $\mathbf{w}_i$ deteriorates both the objective and the constraint satisfaction).
3. $\mathbf{a}_i = -1$ and $\mathbf{g}_i \geq 0$: We replace the variable $\mathbf{w}_i$ by its "inverse", $\check{\mathbf{w}}_i := 1 - \mathbf{w}_i$. This change of variable reverses the signs of $\mathbf{g}_i$ and $\mathbf{a}_i$, which can be handled by the knapsack approach. We also need to update the right-hand side of the constraint as $\check{b} := b - \mathbf{a}_i = b + 1 > b$.

We can now remove the variables $\mathbf{w}_i$ that correspond to the first two cases and consider a reduced knapsack problem in standard form with $\check{m} \leq p$ binary variables in the transformed data $\check{\mathbf{g}}$, $\check{\mathbf{a}}$, $\check{b}$:

$$\underset{\check{\mathbf{w}}}{\text{minimize}} \quad \check{\mathbf{g}}^T \check{\mathbf{w}} \quad \text{subject to} \quad \check{\mathbf{a}}^T \check{\mathbf{w}} \leq \check{b}, \ \check{\mathbf{w}} \in \{0,1\}^{\check{m}}, \tag{16}$$

where $\check{\mathbf{a}} = (1, \ldots, 1)^T$, and $\check{b} \geq 0$. We now sort the indices in increasing order of coefficients:

$$\check{\mathbf{g}}_{i_1} \leq \check{\mathbf{g}}_{i_2} \leq \check{\mathbf{g}}_{i_k} < 0 \leq \check{\mathbf{g}}_{i_{k+1}} \leq \cdots \leq \check{\mathbf{g}}_{i_{\check{m}}},$$

where ties are broken arbitrarily, and we set $i_k = 0$ if $\check{\mathbf{g}}_i \geq 0$ for all indices i. The solution of the reduced knapsack problem (16) is obtained by setting $\check{\mathbf{w}}_{i_l} = 1$ for all $l = 1, \ldots, \min(\check{b}, i_k)$ and $\check{\mathbf{w}}_{i_l} = 0$ for all $l > \min(\check{b}, i_k)$; see, for example, (Balas 1975; Horowitz and Sahni 1974; Martello and Toth 1990, 1988; Pisinger and Toth 1998; Pisinger 1995; Martello et al. 1999).

5 Implementation and experimental setup

In this section, we describe our implementation and the generation of the test instances, and we briefly comment on the calibration of the regularization parameter. We also review a popular NLP-based rounding heuristic that we use in our comparisons.

5.1 Implementation details

We implemented prototype versions and test instances of the proposed algorithms in Julia (Bezanson et al. 2012), which will enable future algorithmic developments thanks to Julia's rapid prototyping capabilities, and AMPL, which facilitates testing different MINLP solvers and relaxation ideas for our problem. To enable the reproducibility of our results, we provide a Julia module containing the source inversion problem and an implementation of the trust-region method freely at www.github.com/JuliaInv/ConvDiffMIPDECO.

The module provides methods to compute the forward problem and matrix-vector products with the adjoint. It also contains several interactive examples that can be modified and extended. The module depends on and extends jInv (Ruthotto et al. 2017), a toolbox for PDE-parameter estimation problem. Our module uses the existing methods in jInv for numerical optimization, PDE solvers, regularization, and visualization in our experiments. All models are solved on a system with two 64-bit Intel(R) Xeon(R) E5-2670 v2, 2.50 GHz CPUs having 10 cores each and sharing 128 GB of RAM. Using the module JuMP (Dunning et al. 2017), our module can also be used to interface with a variety of integer-programming solvers.

In addition we created AMPL code that discretizes the PDE constraint and formulates the MIPDECO, using the finite-difference discretization described below in "Appendix 2". The models and run scripts are freely available at https://github.com/JuliaInv/ConvDiffMIPDECO/tree/master/examples/ampl

We provide the AMPL model as well as scripts that run the penalization-based NLP heuristic described in Sect. 4.1 and scripts that allow the user to output images for further processing in MATLAB. We do not directly compare the Julia runs with the AMPL runs in terms of CPU time because this performance measure is strongly dependent on how well the solvers can exploit the structure of the PDE constraint.

5.2 Generation of test problems and regularization parameter

Two-dimensional instance Using the domain $\Omega = [0, 2] \times [0, 1]$, we construct a 2D source model by evaluating MATLAB peaks function at the cell centers of a grid with 550×256 equally sized cells. Rounding the function with a threshold of 2 results in two sources, one of which we shift right along the x-axis. The true model can be seen in the upper-left subplot of Fig. 1.

To generate the measurements, we solve the PDE using the finite-element method (FEM) discretization on this mesh with a velocity of $v = (1, 0)^\top$ and a diffusion of $c = 0.01$ and then evaluate the PDE solution at 200 random receiver locations sampled from a uniform distribution on Ω. We visualize the PDE solution and the receiver locations (marked by red dots) in the upper-right subplot of Fig. 1.

Three-dimensional instance The data for the 3D instance is generated along the same lines. Here, we choose the domain $\Omega = [0, 2] \times [0, 1] \times [0, 1]$, a mesh size of $128 \times 64 \times 64$, and construct a 3D source model by adding three scaled and shifted norm balls. We visualize the true model in the lower-left subplot of Fig. 1.

To generate the measurements, we solve the PDE using the FEM on this mesh with a velocity of $v = (1, 0, 0)^\top$ and a viscosity of $\sigma = 0.01$, and we then evaluate the PDE solution at 200 randomly spaced boreholes whose first two components are sampled from a uniform distribution on $[0, 2] \times [0, 1]$. In the third dimension, we place one receiver at each mesh cell, which yields overall 12,800 measurements. We visualize the PDE solution using an isocontour plot and the receiver locations (marked by red lines) in the lower-right subplot of Fig. 1. In Table 1 we show the problem sizes of the instances that we solve in our experiments.

Because our inverse problem is underdetermined (we have fewer measurements than unknown optimization variables, $\mathbf{w}$), we must add a regularization term, $\mathcal{R}(\mathbf{w})$,

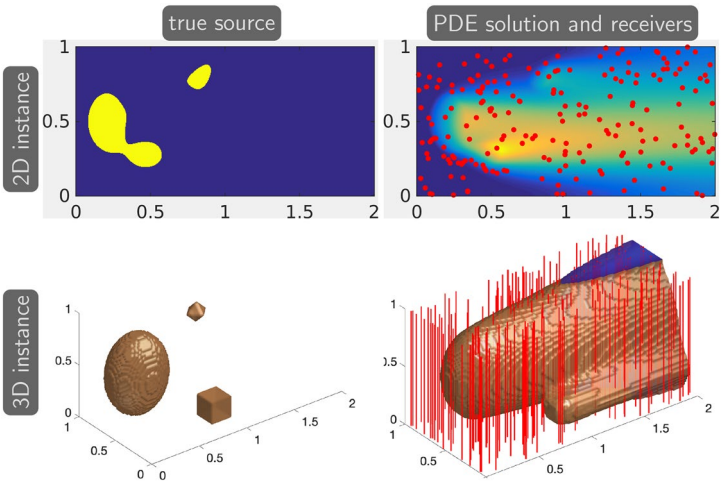

Fig. 1 Visualization of the ground-truth sources (left column) and the generated test data (right column) for the 2D (top row) and 3D (bottom row) instance. The data is obtained by sampling the PDE solution associated with the source model at the randomly chosen receiver locations (indicated by red dots and lines, respectively)

Table 1 Problem size for 2D and 3D MIPDECO instances: For each method and mesh size, we show the number of discrete state, control variables, and constraints. 2D and 3D instances have 200 and 12, 800 number of measurements, respectively

Method	Mesh size	# States	# Binary control	# Constraints
FDM	16×8	180	128	180
	32×16	612	512	612
	64×32	2244	2048	2244
	128×64	8580	8192	8580
	256×128	33540	32768	33540
FEM	256×128	33153	32768	33153
	$96 \times 48 \times 48$	232897	221184	232897

in (3). This regularization term requires us to choose the regularization parameter, α, in (5a). To find an effective regularization parameter, we use the continuous relaxation (7) and follow the L-curve approach that we describe in more detail in "Appendix 3". Using this process, we select the regularization parameters $\alpha = 8.531 \times 10^{-3}$ for the 2D instance and $\alpha = 5.298 \times 10^{-3}$ for the 3D instance, respectively.

6 Numerical results and discussion

In this section, we illustrate the performance of the different approaches to the discrete source inversion problem using numerical experiments in two and three dimensions with known ground truth. We show empirically that state-of-the-art MINLP solvers cannot solve even small-scale two-dimensional instances of this problem.

Next, we consider naïve rounding (also referred to as standard rounding) and two proposed rounding heuristics applied to the relaxed problem, and we show that they also fail to solve the problem. The latter rounding schemes yield better solution than does naïve rounding. We then show that our trust-region heuristic improves on rounding heuristics to produce good-quality solutions in a reasonable amount of time in both two- and three-dimensional cases.

6.1 Performance of MINLP solvers on 2D instances

In this section, we explore the effectiveness of state-of-the-art MINLP solvers for tackling the discretized MIPDECO (6) for the 2D instance. We use six state-of-the-art MINLP solvers: Scip Achterberg 2009, Bonami and Lee (2007) using its hybrid (Bonmin-Hyb), branch-and-bound (Bonmin-BnB), and outer-approximation (Bonmin-OA) algorithms, and Minotaur Mahajan et al. (2017) with both its branch-and-bound (Minotaur-Bnb) and LP/NLP based branch-and-bound (Minotaur-QG) algorithms. We use the self-contained finite-difference discretized MIPDECO model (presented in (29)) for this comparison, because it allows us to easily explore the effect of increasing the discretization and enables others to easily reproduce our results. Otherwise, we use the same problem setup as described in Sect. 5.

We solve a number of instances of the 2D test problem for mesh sizes between 16×8 to 256×128. Table 1 reports the sizes of these instances. In this experiment, we use the regularization parameter, $\alpha = 8.531 \times 10^{-3}$, obtained following the L-curve approach. We limit the CPU time for the MINLP solvers to 10 hours. We report the number of nodes processed, runtime, lower and upper bounds, and percentage gap which is a measure of the optimality gap, and is defined as $100 \times \frac{UB-LB}{|UB|}$, where UB and LB are the upper and lower bounds, respectively, from these solvers. If any of these bounds is unknown, we set the percentage gap to infinity. We note that Scip reports that the optimality gap is infinite if the lower and upper bounds have opposite signs.

We use the intersection-over-union (IoU) score (also known as Jaccard index) to quantify the overlap between the true source and the reconstruction source; see (Csurka et al. 2013). Let M_{true} and M_{recon} denote the sets for which the true source, w, and the reconstructed source, w_{recon}, are indicator functions, respectively. The IoU score is then defined as the volume of the intersection divided by the volume of the union of these sets:

$$\text{IoU} = \frac{|M_{\text{true}} \cap M_{\text{recon}}|}{|M_{\text{true}} \cup M_{\text{recon}}|} \in [0, 1].$$

Higher values of the IoU score indicate a better overlap. Since the inversion is performed on coarser meshes, we use a next-neighbor interpolation to refine the reconstructed sources.

In Table 2, we summarize the performance of the MINLP solvers. We observe that only the two branch-and-bound solvers, Bonmin-Bnb and Minotaur-Bnb, are able to solve the smallest 16×8 instance; Bonmin-BnB also solved 32×16 but took around 454 minutes. All other runs time out after 10 CPU-hours, and in many cases

Table 2 Performance of state-of-the-art MINLP solvers for instances of the 2D test problem with mesh sizes ranging from 16×8 to 256×128

	Solver	# nodes	Runtime (s)	Bound		Gap (%)
				Lower	Upper	
16×8	Scip	225,691	TIME-OUT	− 0.5609	0.1733	∞
	Bonmin-Bnb	266	10.71	0.0530	0.0530	0
	Bonmin-Hyb	2,087,613	TIME-OUT	0.0413	0.0612	32.40
	Minotaur-Bnb	1043	163.87	0.0530	0.0530	0
	Minotaur-QG	560,436	TIME-OUT	0.0416	0.0530	21.51
32×16	Scip	32,868	TIME-OUT	− 1.6063	–	∞
	Bonmin-Bnb	242,126	27222.15	0.0393	0.0393	0
	Bonmin-Hyb	1,606,956	TIME-OUT	0.0347	0.0634	45.24
	Bonmin-OA	2,087,613	TIME-OUT	0.0413	0.0612	32.40
	Minotaur-Bnb	58,115	TIME-OUT	0.0364	0.0575	36.71
	Minotaur-QG	265,309	TIME-OUT	0.0347	0.0470	26.08
64×32	Scip	183	TIME-OUT	-2.0189	1.7104	∞
	Bonmin-Bnb	13,569	TIME-OUT	0.0338	0.0369	8.38
	Bonmin-Hyb	293,128	TIME-OUT	0.0329	0.0859	61.64
	Bonmin-OA	2,087,613	TIME-OUT	0.0413	0.0612	32.40
	Minotaur-Bnb	956	TIME-OUT	0.0329	0.1570	79.03
	Minotaur-QG	322,969	TIME-OUT	0.0329	0.0449	26.61
128×64	Scip	1	TIME-OUT	–	–	∞
	Bonmin-Bnb	1	TIME-OUT	0.0323	0.0481	32.79
	Bonmin-Hyb	17,316	TIME-OUT	0.0293	0.1696	82.69
	Bonmin-OA	2,087,613	TIME-OUT	0.0413	0.0612	32.40
	Minotaur-Bnb	8	TIME-OUT	0.0322	–	∞
	Minotaur-QG	77,295	TIME-OUT	0.0322	0.0696	53.74
256×128	Scip	811	TIME-OUT	–	–	∞
	Bonmin-Bnb	1	TIME-OUT	0.0355	–	∞
	Bonmin-Hyb	17,316	TIME-OUT	0.0119	0.1725	93.1
	Bonmin-OA	2,087,613	TIME-OUT	0.0413	0.0612	32.40
	Minotaur-Bnb	1	TIME-OUT	–	–	∞
	Minotaur-QG	1	TIME-OUT	–	–	∞

The rows represent different solvers and are grouped by mesh sizes. The columns show (left to right) the mesh size, the name of the solver, the number of nodes processed, the run-time in seconds (where TIME-OUT indicates that we reached the time limit of 10 CPU-hours), the lower and upper bounds, and the percentage gap remaining

the solvers fail to even produce a feasible source, $\mathbf{W}$, or at least one of the bounds (lower or upper), indicated by ∞ in the last column. Bonmin-OA finds only the trivial feasible point, $\mathbf{W} = 0$, on all the instances, indicating that there are no sources. Hence, we excluded the Bonmin-OA results.

Figure 2 shows the best solution, $\mathbf{W}$, obtained by the MINLP solvers. The results for the 16×8 case show that the upper bound by Scip and Bonmin-Hyb are far from

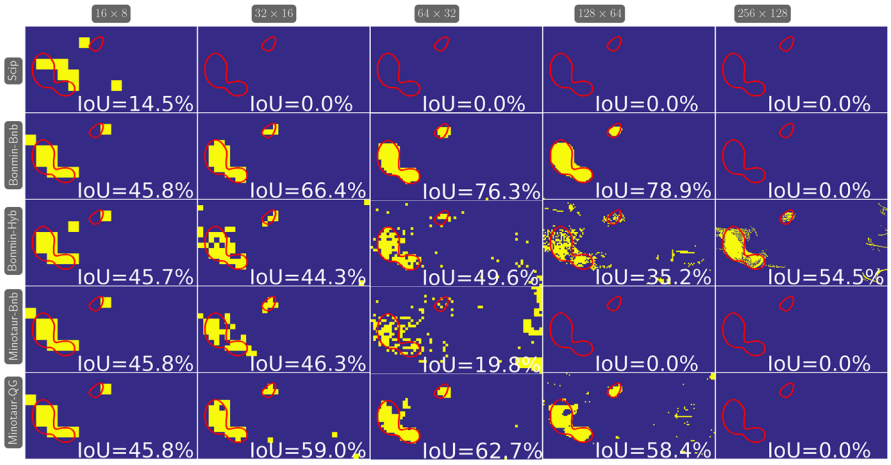

Fig. 2 Solutions (**W**) from MINLP solvers SCIP, Bonmin-Bnb, Bonmin-Hyb, Minotaur-Bnb, and Minotaur-QG (row-wise) on mesh sizes of 16×8, 32×16, 64×32, 128×64, 256×128 (column-wise). We indicate the IoU value in the lower-right corner of each image

optimal; Minotaur-QG found the upper bound (which in this case is also optimal) but could not improve its lower bound and thus finished with a positive optimality gap. As we increase the size of the problem, the MINLP solvers tend to obtain poor reconstructions with speckled areas that would make a source identification difficult. The worst performance is at the finest discretization level, where only Bonmin-Hyb returns some speckled sources and all other solvers fail to identify the sources.

One reason for this poor performance is that all MINLP methods solve a large number of relaxations of the original problem, such as linear programs, nonlinear programs, and mixed-integer linear programs, depending on the specific method. Moreover, the problem size increases as we refine the computational mesh, making these problems larger and computationally harder to solve. None of the off-the-shelf solvers exploit the special structure that is inherent in the discretized PDEs and, for example, do not take advantage of the fact that the stiffness matrix needs to be factorized only once.

Another factor that prevents the MINLP solvers from solving our problem is the presolve techniques that SCIP and Minotaur employ, such as bound tightening, and the derivation of implications, before starting (and intermittently during) the tree-search (Achterberg 2005; Mahajan et al. 2011). SCIP, for example, reformulates the original problem by decomposing the nonlinear objective function into a set of quadratic and nonlinear constraints whose number increases with the size of the instance. For mesh size 128×64, the SCIP preprocessing step took 425.95 seconds and resulted in 8,002 added quadratic constraints, making relaxations harder to solve, especially in view of the fact that our problem can be solved as a bound-constrained NLP by eliminating the PDE states and constraint.

We note that the heuristics used in SCIP and cutting planes used in Bonmin-OA to find better feasible solutions seem to be ineffective for our problems. For example, the best solutions reported by Bonmin-OA for all instances is $W = 0$, which

indicates that there are no sources, which - even though feasible - is not a meaningful solution.

We also observe that the optimality gap is high for most solvers, because the lower and upper bounds are quite weak, leading to slow convergence. For mesh-size 256×128, Bonmin (in all algorithms) could not improve its starting lower bound even after the 10 CPU-hours, and Minotaur failed to report even a single lower bound because of a restoration failure in IPOPT that assumed local infeasibility. Initially, we allowed only two CPU-hours. Raising this limit to ten hours did not change the quality of the bounds, indicating that these problems are unlikely to be solved within a reasonable time with existing MINLP solvers.

6.2 Results for rounding approaches

Here, we compare the effectiveness of the different rounding schemes discussed in Sect. 4.1, on the finest mesh (256×128). The naïve, mass-preserving, and gap-reduction rounding schemes start from the continuous relaxation solution of (5) given by (7). In contrast, the NLP-based rounding heuristic solves a sequence of NLPs, taking 11 and 7 iterations for the 2D and 3D case, respectively. In Table 3 we report the objective value, the solution time, and the number of PDEs of the solutions obtained from these rounding schemes. For the first two rounding schemes, the number of PDE solves is due to solving the relaxation (7). While the computational costs of the naïve and mass-preserving scheme are negligible, the gap reduction rounding requires repeated evaluation of the objective function and thus the PDE solves. Note that the factorizations of the discretized PDEs were computed during the solution of the relaxed problem and reused during the rounding. The additional costs are 16 and 13 PDE solves in the 2D and 3D cases, respectively. The majority of the time required by these three rounding schemes is from solving the relaxation. All the objective values reported henceforth are the objective value as in (7).

In Figs. 6 and 7 the first column shows the solution $\mathbf{w}$ from these rounding schemes. The NLP-based rounding resulted in a better solution than the rest but took around 33 (22) times more time for 2D (3D) case. We applied the different schemes at every iteration of the NLP-based rounding heuristic. In the 2D case, we obtain better solutions with proposed rounding schemes (mass-preserving and gap-reduction) than with a simple rounding scheme, and in some iterations, our rounded

Table 3 Comparison of objective value, solving time, and number of the PDE solves of different rounding schemes

Rounding	2D			3D		
	Obj	Time (s)	# PDE solves	Obj	Time (s)	# PDE solves
Naïve	0.1098	24.29	462	0.0246	587.27	565
Mass-pres	0.0780	24.25	462	0.0262	587.24	565
Gap-red	0.1027	24.76	478	0.0246	600.36	578
NLP-based	0.0530	794.10	2750	0.0204	12920.39	1518

solutions are better than any solution of the NLP-based rounding heuristic. In the 3D case, the naïve and gap-reduction roundings resulted in the same solution. Figure 3 reports iteration statistics of the NLP-based rounding heuristic. The top row corresponds to the 2D case and bottom row to the 3D case. In a row, the leftmost figure shows the number of PDE solves and solution time at each iteration; the middle figure reports the optimal objective value (Obj val) and objective value of the integer solution obtained by employing different rounding schemes (naïve, mass-preserving, and gap-reduction) to the solution of NLP-based heuristic at each iteration; the rightmost figure shows the IoU values associated with different integer feasible solution reported in the middle figure. These results encourage us to believe that NLP-based heuristic with higher integrality tolerance can also give a good-quality integer solution when used in conjunction with the proposed simple rounding schemes. We note that at iteration 10 of the 2D instance the objective value from the NLP-based heuristic is more than the rounded solution given by the gap-reduction rounding. This means that the former is not a valid lower bound, because problem (8) is nonconvex, and we solve it only with a (local) projected Gauss-Newton method.

6.3 An integrated rounding heuristic and trust-region approach

We now apply our new trust-region approach to the intermediate solutions from the NLP-based rounding approach of Sect. 6.1 for both the 2D and 3D instances. Note that the first iteration in the NLP-based heuristic corresponds to the relaxation of (5). For each iteration we consider three rounding schemes (naïve, mass-preserving, and objective gap-reduction) and two versions of the trust-region approach (full

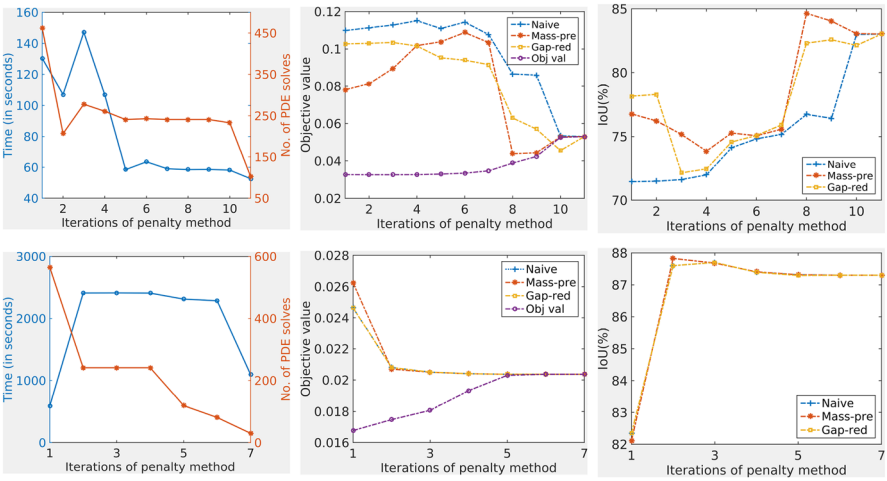

Fig. 3 Detailed summary of solution time, # of PDE solves, objective value of the original problem, and objective and IoU values of integer feasible solutions obtained by the naïve, mass-preserving, and gap-reduction roundings of the solution at each iteration in the NLP-based heuristic. Top row is for 2D instance and bottom row is for 3D instance

region and neighborhood of 1 pixel), resulting in six combinations of our algorithm (at each iteration). We refer to $\mathcal{N}_\theta$ in (13) with $\theta = \sqrt{\left(\frac{2}{256}\right)^2 + \left(\frac{1}{128}\right)^2}$ and $\theta = \sqrt{\left(\frac{2}{96}\right)^2 + \left(\frac{1}{48}\right)^2 + +\left(\frac{1}{48}\right)^2}$ as neighborhood of 1 pixel in our 2D and 3D cases, respectively. For the 2D instance, Fig. 4 shows the objective and IoU values from each of these combinations at each iteration; left and right columns are for the trust-region approach on full-space and neighborhood of one pixel, respectively. From our computational results we see that for the 2D instance, for all the iterations (other than iteration 7) the mass-preserving has performed better than the other rounding schemes. Also, the mass-preserving and objective gap-reduction roundings give better solutions in terms of objective than does naïve rounding for the trust-region approach in all the iterations other than the last two. In the last two iterations the integrality gap in small and all three roundings result is the same initial and thus final solutions. Similar results for the 3D case are reported in Fig. 5.

Figure 6 shows the source reconstructions for the first iterate of the penalty method applied to the 2D instance for each of the six combinations of our algorithm (first three rows), and the last iterate considering only mass-preserving

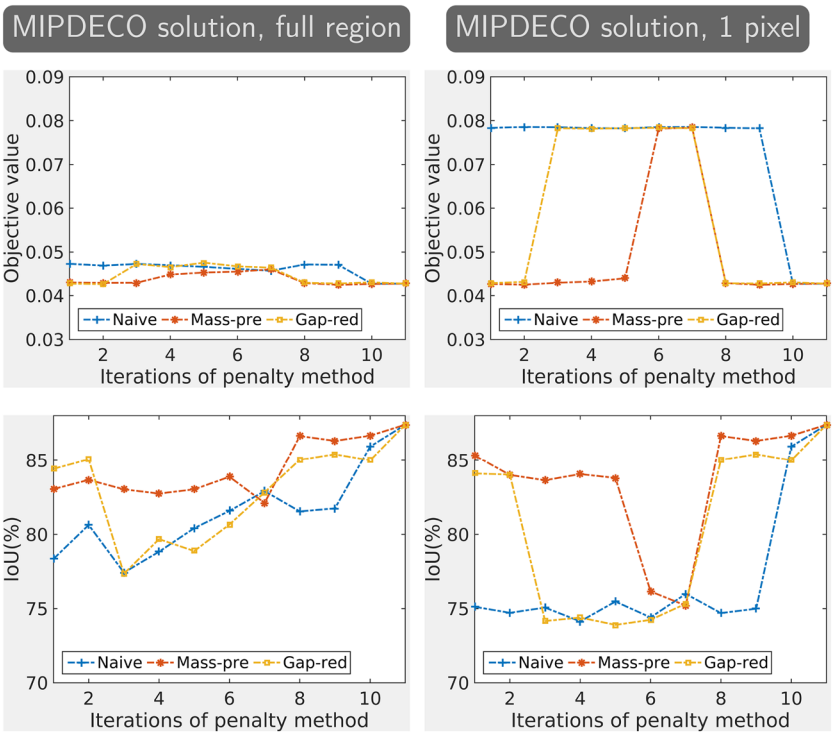

Fig. 4 Objective and IoU values of trust-region approach on the solutions obtained by the naïve, mass-preserving, and gap-reduction roundings of the solution at each iteration in the penalization-based NLP heuristic for 2D instance

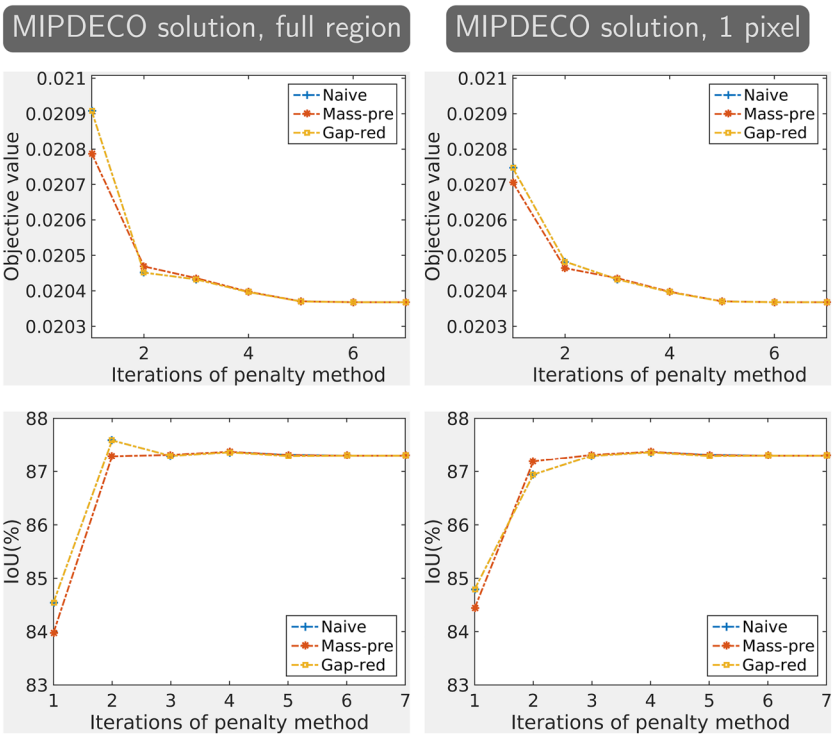

Fig. 5 Objective and IoU values of trust-region approach on the solutions obtained by the naïve, mass-preserving, and gap-reduction roundings of the solution at each iteration in the penalization-based NLP heuristic for 3D instance

rounding (since the integrality gap is small, all the three rounding schemes give same initial solution) in the last row. Each row depicts the initial guess, the reconstruction computed by the full-space trust-region method, and the reconstruction from the neighborhood trust-region method for a given rounding scheme. The superimposed red lines depict the shape of the true source. In each case, the overlap is improved by the trust-region method. Similar results for the 3D case are reported in Fig. 7.

In the 2D case, we observe that for the first few iterations in the NLP-based heuristics naïve rounding identified the larger of the two sources and completely failed to identify the second smaller source. However, the other two roundings, mass-preserving and gap-reduction, identified both sources. When only one of the sources is recovered by using a rounding schemes the neighborhood trust-region approach cannot identify the second source, because it can change $\mathbf{w}_i$ only near the initial guess. On the other hand, the full-space trust-region algorithm discovers the second smaller source as well, independent of the starting guess.

In 2D (3D) instances, the added runtime of the trust-region approaches is around 7 seconds (between 58 and 129 seconds) when the initial guess is obtained

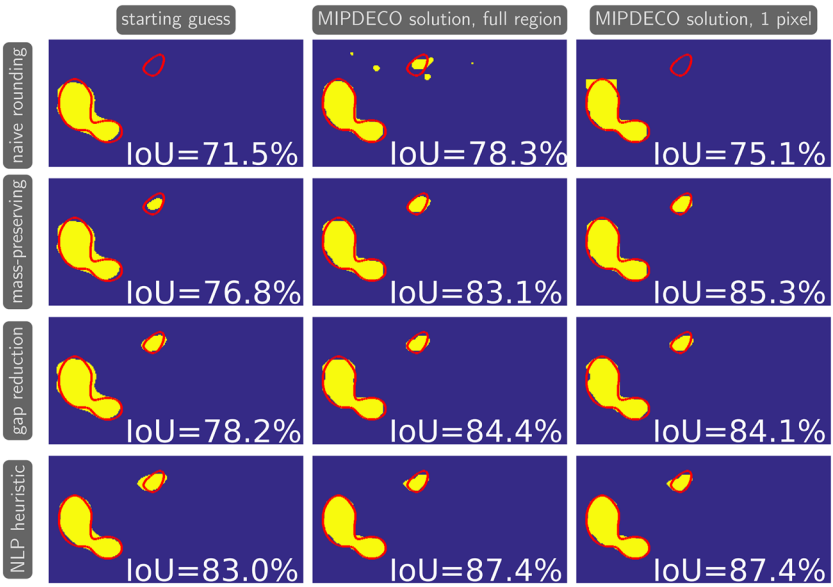

Fig. 6 2D results of the full-space and neighborhood variant of the trust-region approach for different rounding heuristics (row-wise). The left column shows the starting guess, obtained by rounding the solution of the relaxed problem at the α value selected from the L-curve. The middle and right columns depict the solutions obtained using the trust region methods. The superimposed red line indicates the location of the true source. The overlap is quantified using the intersection-over-union score (IoU) and printed in the lower-right corner of each image

from the first iteration of the NLP-based heuristic and less than 3 (between 10 and 74 seconds) seconds when the initial guess is from any other iteration (the largest number of PDE solves for the trust-region scheme was 102 (82); combining this with the PDE solves required for solving the relaxed problem, the total number of PDE solves was 564 (647)). We note that the number of PDE solves is significantly lower than the total number of binary variables, indicating that our solution approach is efficient for solving these large-scale MINLPs.

In Table 4 we report the final objective value obtained from our algorithms. In the first three rows, the initial guess is obtained from rounding the relaxation solution. For the NLP heuristic we used as a starting guess the final iteration solution with the naïve rounding, because the other two rounding schemes also result

Table 4 Final objective function value of rounding heuristics and trust-region approach

Rounding	2D		3D	
	Full region	1 pixel	Full region	1 pixel
Naïve	0.0473	0.0784	0.0209	0.0207
Mass-preserving	0.0431	0.0427	0.0208	0.0207
Gap-reduction	0.0428	0.0429	0.0209	0.0207
NLP heuristic	0.0428	0.0428	0.0204	0.0204

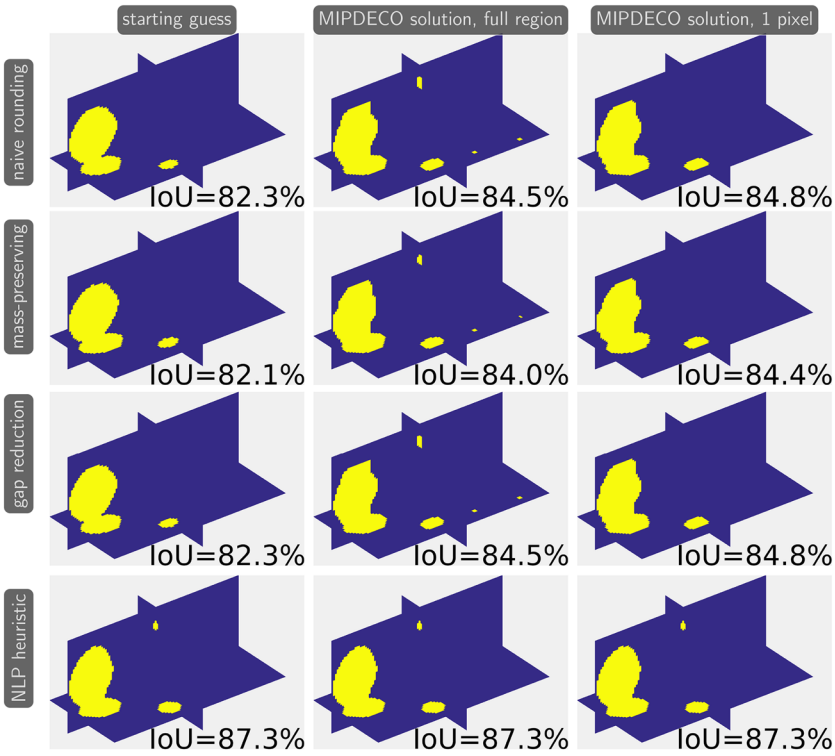

Fig. 7 3D results of the full-space and neighborhood variant of the trust-region approach for different rounding heuristics (row-wise). The left column shows the starting guess, obtained by rounding the solution of the relaxed problem at the α value selected from the L-curve. The middle and right columns depict the solutions obtained using the trust region methods. The overlap is quantified using the IoU and printed in the lower-right corner of each image

in the same integer feasible solution (due to the very small integrality gap). Using the proposed trust-region heuristic, We obtain a percentage improvement of 132%, 82.67%, 139.39%, and 23.83% in the objective values of naïve, mass-preserving, gap-reduction, and NLP heuristic rounding solutions, respectively, for 2D case; and of 18.84%, 26.57%, and 18.84% in the objective values of naïve, mass-preserving, and gap-reduction solutions, respectively, for the 3D case.

Figure 8 shows the progress in objective value and the change in the trust-region radius for the MIPDECO instances. Here we use the first iteration of the penalty method as an initial guess. We observe that the trust-region algorithm terminates in a modest number of iterations (typically in the range [25, 51]), which implies that we solved at most twice the PDEs to obtain function and adjoint information (we do not need to solve the adjoint equation on iterations on which we reject the step). These results are encouraging, given that the bulk of the computational effort is the initial factorization of the stiffness matrix, which we do once during the solution of the relaxed discretized MIPDECO and after which we can reuse the factors for fast

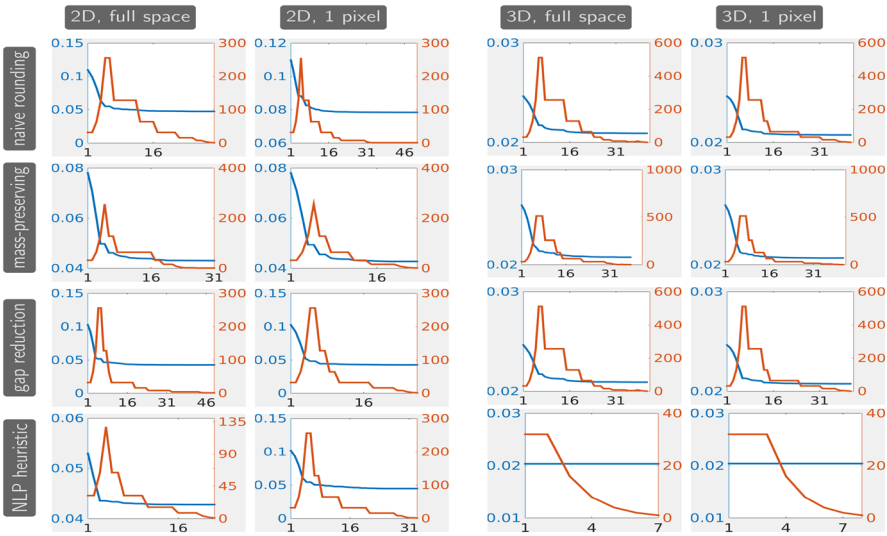

Fig. 8 Convergence histories of the MIPDECO heuristics. Each plot shows the values of the objective function (blue line, left y-axis) and the trust-region radius (red line, right y-axis) at each iteration (x-axis). The rows correspond to the instances obtained for the naïve, the mass-preserving, and the gap-reduction rounding, respectively. The columns represent the 2D and 3D results of the full-space and reduced space methods

PDE solves. The reduction in the function value that we obtain is also encouraging, showing that we can significantly improve the objective value in our trust-region iterations.

7 Conclusions and future work

In this paper, we apply several solution approaches to a discrete source inversion problem for the convection–diffusion equation. We discretize the given mixed-integer PDE-constrained optimization (MIPDECO) problem using finite elements and obtain a large-scale convex MINLP. Using numerical examples, we demonstrate that the discretization of this problem can be solved neither by rounding solutions of the relaxed problem nor by state-of-the-art MINLP solvers. We propose a new heuristic for MIPDECO that combines a problem-specific rounding scheme with an improvement heuristic. The method is motivated by trust-region methods for nonlinear optimization and is related to the neighborhood search and local-branching heuristics for MINLP.

We show that our proposed heuristic can solve both 2D and 3D problem instances with more than 65,000 binary variables. In particular, our full-space trust-region approach can add sources even if the initial guess misses an existing source. The algorithm solves at most two PDEs per iteration, and our Julia implementation reuses factorizations of the stiffness matrix for computational

efficiency. In most cases, the trust-region approach converges in a modest number of iterations (often around 30).

There are several ways to further improve the efficiency of MINLP solvers, which use IPOPT solve the continuous relaxations and the nodes in the branch-and-bound tree. Because IPOPT is a general-purpose framework for solving optimization problems, it does not take advantage of the structure of the discretized PDE. In particular, IPOPT refactors the stiffness matrix on every iteration, although in principle one could rewrite the linear algebra inside IPOPT to take advantage of these factors. On the other hand, the PDECO solver jInv is geared toward PDE-constrained problems and includes a number of choices that reduce the runtime for the specific instance. Since in the problem at hand the PDE-operator does not depend on the optimization variable, our jInv uses a direct method to factorize the stiffness matrix before solving the relaxed problem. While computing the factorization in 3D takes a significant amount of time, subsequent evaluations of the objective function, gradients, and matrix-vector products with the Hessians can be computed quickly. We include open-source implementations in Julia and AMPL that allow others to reproduce and improve the results shown here.

An interesting question raised by one of the referees is whether our trust-region algorithm could be generalized to solve very large-scale MINLPs with a convex objective and linear constraints. The challenge in our view is to ensure that the trust-region subproblem remains tractable. In our case, we can eliminate the continuous (state) variables, $\mathbf{u}$, using the linear system (4), and solve a simple knapsack problem. In general, MINLPs, have additional structure, and it may not be possible to simply eliminate the continuous variables and constraints. However, we believe that our approach could still be used in certain large-scale MINLPs.

Acknowledgements This work was initiated as a part of the SAMSI Program on Optimization 20162017. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. Part of this material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, under Contract DE-AC02-06CH11357. This work was also supported by the U.S. Department of Energy through grant DE-FG02-05ER25694. Ruthotto's work was partially supported by the US National Science Foundation award DMS 1522599. Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia LLC, a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA-0003525. SAND2019-10626 J. Hahn's work was supported by the DFG (German Research Foundation) under SPP 1962 and by the German Federal Ministry of Education and Research (BMBF) under P2Chem grant 05M18NMA. We are grateful to Prof. Martin Siebenborn who helped us with the FEM derivation and deriving the weak form of the PDE. The submitted manuscript has been created by UChicago Argonne, LLC, Operator of Argonne National Laboratory (Argonne). The U.S. Government retains for itself, and others acting on its behalf, a paid-up nonexclusive, irrevocable worldwide license in said article to reproduce, prepare derivative works, distribute copies to the public, and perform publicly and display publicly, by or on behalf of the Government. The Department of Energy will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan. http://energy.gov/downloads/doe-public-access-plan

Appendix 1: Finite-element discretization of the source inversion problem

Here, we briefly review how we discretize the variational problem (1) on a computational mesh. We employ the first-discretize-then-optimize approach, which is a common strategy in PDE-constrained optimization; see; for example, (Gunzburger 2003, Sec. 2.9).

We begin by discretizing the PDE-constraint in (1) following the usual procedure of finite-element methods. To obtain a weak form of the constraint, we multiply both sides of the PDE constraint in (1) with a test function $\phi \in H^1(\Omega, \mathbb{R})$, the Sobolev space of all functions whose first derivative is square integrable over Ω, with $\phi(x) = 0$ for $x \in \Gamma_D$, $\frac{\partial \phi(x)}{\partial n} = 0$ for $x \in \Gamma_N$, and then apply Green's identity and integration by parts:

$$\int_\Omega w\phi dx = \int_\Omega (-c\Delta u + v^\top \nabla u)\phi dx \tag{17}$$

$$= \int_\Omega c(\nabla u)^\top \nabla \phi + (v^\top \nabla u)\phi dx + \int_{\partial\Omega} \phi\Big(-c\frac{\partial u}{\partial n}\Big) ds \tag{18}$$

$$= \int_\Omega c(\nabla u)^\top \nabla \phi + (v^\top \nabla u)\phi dx, \tag{19}$$

where the last identity is obtained by using the Neumann boundary conditions on Γ_N and the fact that $\phi(x) = 0$ on Γ_D. The weak problem then is to find a $u \in H^1(\Omega, \mathbb{R})$ that satisfies (19) for all test functions $\phi \in H^1(\Omega, \mathbb{R})$.

Next, we discretize the state u and the control w in the weak form of the PDE constraint (19) using compactly supported Ansatz functions on a computational mesh. For ease of presentation we assume that our computational domain $\Omega = [0, 1]^d$ is divided into N^d quadrilateral finite elements of edge length $L = 1/N$. We assume a lexicographical ordering of the elements $\Omega_1, \ldots, \Omega_{N^d}$ in the mesh, which in our case consists of pixels and voxels for $d = 2, 3$, respectively. We note that extensions to more general domains and anisotropic or unstructured meshes are straightforward. For example, the implementation used in our numerical experiments uses rectangular meshes whose elements have different edge lengths along the coordinate direction. We use the standard bi-/trilinear Ansatz functions $\phi_1, \phi_2, \ldots, \phi_{(N+1)^d}$ for $d = 2$ and $d = 3$, respectively, as test functions and to discretize the state variable u:

$$u(x) = \sum_{i=1}^{(N+1)^d} \mathbf{u}_i \phi_i(x). \tag{20}$$

Because the Ansatz functions form a Lagrange basis (i.e., for the jth node of the mesh we have $\phi_i(x_j) = \delta_{ij}$), the elements in $\mathbf{u}$ correspond to the value of u at the nodes of the mesh. We represent the control variable w as a linear combination of piecewise constant Ansatz functions $\psi_1, \psi_2, \ldots, \psi_{N^d}$:

$$w(x) = \sum_{i=1}^{N^d} \mathbf{w}_i \psi_i(x). \tag{21}$$

Similar to above, the Ansatz functions form a Lagrange basis (i.e., $\psi_i(x) = 1$ if $x \in \Omega_i$ and $\psi(x) = 0$ else); hence the entries in $\mathbf{w}$ correspond to the values of w in the pixels/voxels of our mesh.

The finite-dimensional approximations of u and w in (20) and (21) allow us to write the weak form of the PDE-constraint (19) in terms of the coefficient vectors $\mathbf{u}$ and $\mathbf{w}$ as

$$\mathbf{S}\mathbf{u} = \mathbf{M}\mathbf{w} - \mathbf{r}, \tag{22}$$

where $\mathbf{S} \in \mathbb{R}^{(N+1)^d \times (N+1)^d}$ is the (nonsingular) stiffness and $\mathbf{M} \in \mathbb{R}^{(N+1)^d \times N^d}$ is the (full-rank) mass matrix, respectively. We compute the entries of both matrices, $\mathbf{M}$ and $\mathbf{S}$, approximately by applying a 5th-order Gaussian quadrature rule to (19):

$$\mathbf{S}_{ij} \approx \int_\Omega c(\nabla \phi_j)^\top \nabla \phi_i - \phi_j v^\top \nabla \phi_i dx \tag{23}$$

$$\mathbf{M}_{ij} \approx \int_\Omega \psi_j \phi_i dx. \tag{24}$$

$$\mathbf{r}_j \approx \int_{\Gamma_D} \phi_j g(v^\top n) ds. \tag{25}$$

Because of the compact support of the basis functions, the integrands vanish on all but a few elements, which leads to both matrices being sparse.

We now derive our discretization of the total variation regularizer (3). In this work, we use a first-order finite-difference discretization of the regularizer. This choice is motivated by its simplicity and computational efficiency. We refer to the recent work in Herrmann et al. (2018) for a more extensive discussion and finite-element discretizations of the total variation. Because we assume a quadrilateral mesh, the discrete gradient operators for $d = 2$ and $d = 3$ are

$$\mathbf{G} = \begin{pmatrix} \mathbf{I}_N \otimes \mathbf{D} \\ \mathbf{D} \otimes \mathbf{I}_N \end{pmatrix} \quad \text{and} \quad \mathbf{G} = \begin{pmatrix} \mathbf{I}_N \otimes \mathbf{I}_N \otimes \mathbf{D} \\ \mathbf{I}_N \otimes \mathbf{D} \otimes \mathbf{I}_N \\ \mathbf{D} \otimes \mathbf{I}_N \otimes \mathbf{I}_N \end{pmatrix}. \tag{26}$$

Here, $\mathbf{I}_N \in \mathbb{R}^{N \times N}$ is the identity matrix, $\otimes$ denotes the Kronecker product, and the one-dimensional difference operator is

$$\mathbf{D} = \frac{1}{L}\begin{pmatrix} -1 & 0 & 0 & \cdots & \cdots & 0 \\ -1 & 1 & 0 & \cdots & \cdots & 0 \\ 0 & -1 & 1 & 0 & \cdots & 0 \\ \vdots & & \ddots & \ddots & & \vdots \\ \vdots & & & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & 0 & -1 & 1 \\ 0 & \cdots & \cdots & 0 & 0 & 1 \end{pmatrix} \in \mathbb{R}^{N+1\times N}.$$

We note that this discretization involves a grid change (from the N cell-centers to the $N+1$ nodes). To accommodate for the grid change, the relaxed form includes the average matrix $\mathbf{A}$

$$\mathbf{A} = \big(\mathbf{I}_N \otimes \mathbf{B} \,|\, \mathbf{B} \otimes \mathbf{I}_N\big)$$

for $d = 2$ and

$$\mathbf{A} = \big(\mathbf{I}_N \otimes \mathbf{I}_N \otimes \mathbf{B} \,|\, \mathbf{I}_N \otimes \mathbf{B} \otimes \mathbf{I}_N \,|\, \mathbf{B} \otimes \mathbf{I}_N \otimes \mathbf{I}_N\big),$$

for $d = 3$, respectively, where we use the one-dimensional average matrix

$$\mathbf{B} = \frac{1}{2}\begin{pmatrix} 2 & 0 & & \cdots & \cdots & 0 \\ 1 & 1 & 0 & \cdots & \cdots & 0 \\ 0 & 1 & 1 & 0 & \cdots & 0 \\ \vdots & & \ddots & \ddots & & \vdots \\ \vdots & & & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & 0 & 1 & 1 \\ 0 & \cdots & \cdots & & 0 & 1 \end{pmatrix} \in \mathbb{R}^{N\times N+1}.$$

This leads to the discrete regularization function

$$R_{\text{iso}}(\mathbf{w}) = L^d \mathbf{e}^\top \sqrt{\mathbf{A}(\mathbf{G}\mathbf{w})^2 + \kappa}, \tag{27}$$

where $\mathbf{e}$ is a vector of all ones, the factor L^d represents the volume of the cells, and $\kappa > 0$ is a conditioning parameter (in our experiments, we use $\kappa = 10^{-3}$). Note that the square and the square root are applied component-wise.

Appendix 2: Finite-difference discretization of the source inversion problem

We discretize both the source, w, and the state, u, in the cell-centered points of our $N_x \times N_y$ computational mesh:

$$\mathbf{W}_{kl} \simeq w\big(kL_x - L_x/2, lL_y - L_y/2\big), \quad \mathbf{U}_{ij} \simeq u\big(iL_x - L_x/2, jL_y - L_y/2\big),$$

where $L_x = 2/N_x$ and $L_y = 1/N_y$ are the discretization steps in the x and y direction, respectively, and $k = 1, \ldots, N_x, l = 1, \ldots, N_y,\ i = 0, \ldots, N_x + 1, j = 0, \ldots, N_y$. Here,

the variables $\mathbf{U}_{i,0}, \mathbf{U}_{N_x+1,j}, \mathbf{U}_{i,N_y+1}$ approximate the PDE solution at ghost points placed along Γ_N, and the variables $\mathbf{U}_{0,j}$ are the ghost points near Γ_D.

Let us further define $\mathbf{V} \in \mathbb{R}^m$ to obtain the bilinear interpolation from the cell-centered points of the mesh closest to the receiver locations $r^1, r^2, \ldots, r^m$. Mathematically, for each receiver location $r^k = (r_x^k, r_y^k), k = 1, \ldots, m$, we define variable V_k as

$$V_k = \frac{1}{L_x L_y} \begin{pmatrix} iL_x + \frac{L_x}{2} - r_x^k \\ r_x^k - iL_x + \frac{L_x}{2} \end{pmatrix}^T \begin{pmatrix} \mathbf{U}_{i,j} & \mathbf{U}_{i,j+1} \\ \mathbf{U}_{i+1,j} & \mathbf{U}_{i+1,j+1} \end{pmatrix} \begin{pmatrix} jL_y + \frac{L_y}{2} - r_y^k \\ r_y^k - jL_y + \frac{L_y}{2} \end{pmatrix}, \tag{28}$$

where, $i = \lfloor \frac{r_x^k}{L_x} + 0.5 \rfloor$ and $j = \lfloor \frac{r_y^k}{L_y} + 0.5 \rfloor$, leading to the following mixed-integer quadratic program:

$$\begin{aligned} &\min_{\mathbf{U},\mathbf{W}} \frac{1}{2\sigma} \Big(\sum_{k=1}^{m} \Big(\mathbf{V}_k - b_k \Big)^2 \Big) \\ &\quad + \alpha L_x L_y \sum_{k=2}^{N_x} \sum_{l=2}^{N_y} \sqrt{ \left(\frac{\mathbf{W}_{kl} - \mathbf{W}_{(k-1)l}}{L_x} \right)^2 + \left(\frac{\mathbf{W}_{kl} - \mathbf{W}_{k(l-1)}}{L_y} \right)^2 + \kappa }, \\ &\text{s.t. } c \frac{4\mathbf{U}_{ij} - \mathbf{U}_{(i-1)j} - \mathbf{U}_{(i+1)j} - \mathbf{U}_{i(j-1)} - \mathbf{U}_{i(j+1)}}{{L_x}^2 {L_y}^2} \\ &\quad + \frac{\mathbf{U}_{ij} - \mathbf{U}_{(i-1)j}}{L_x} = \mathbf{W}_{ij}, \ i = 1, \ldots, N_x, \quad j = 1, \ldots, N_y \\ &\quad \mathbf{U}_{N_x+1,j} = \mathbf{U}_{N_x,j}, \ \mathbf{U}_{i,0} = \mathbf{U}_{i,1}, \ \mathbf{U}_{i,N_y+1} = \mathbf{U}_{i,N_y}, \mathbf{U}_{0,j} = -\mathbf{U}_{1,j}, \\ &\quad i = 0, \ldots, N_x, \quad j = 0, \ldots, N_y \\ &\quad \mathbf{W} \in \{0,1\}^{N_x \times N_y}, \mathbf{U} \in \mathbb{R}^{(N_x+2)\times(N_y+2)}. \end{aligned} \tag{29}$$

Here, the fifth row explicitly encodes the Neumann and Dirichlet boundary conditions. As before, we can again eliminate the state variables $\mathbf{U}$ using the discretized PDE and boundary conditions and the state variables $\mathbf{V}$ using (28), resulting in a problem that has similar structure to (6). As before, the objective function is second-order cone representable.

Appendix 3: Selection of regularization parameter for the relaxed problem

To find an effective regularization parameter, we consider the continuous relaxation (7) and follow the L-curve procedure; see (Hansen 1998) for details. In the inversions we use coarser meshes with 256×128 and $96 \times 48 \times 48$ cells for the 2D instance and 3D instance, respectively. We consider the datasets generated in the preceding section and perturb the generated data with 10% iid Gaussian white noise.

To compute the L-curve, we solve 30 instances of the continuous relaxation for different values of α that are logarithmically spaced between 1 and 10^{-6}. To accelerate the computation, we initialize the optimization with the solution from the previous α. For each value of α, we use up to 20 Gauss-Newton iterations and approximately compute the search direction using 5 iterations of projected preconditioned CG that use the Hessian of the regularization function as a preconditioner. For each experiment, we store the reconstructed source, the predicted data, the value of the misfit function, and the values of the regularization function (without the factor α). The L-curve shown in Fig. 9 show the value of the regularizer and the value of the misfit of these optimal solutions. As is common, the axes are scaled logarithmically; and to provide additional insight, we have added visualizations of the reconstructed sources for the largest and smallest value of α (resulting in overly smoothed and very noisy reconstructions, respectively) as well as solutions that provide a good trade-off. Using this process we select the regularization parameters $\alpha = 8.531 \times 10^{-3}$ for the two-dimensional instance and $\alpha = 5.298 \times 10^{-3}$ for the three-dimensional instance, respectively. Computing the L-curves took about 4 and 48 minutes and involved 32,580 and 30,892 PDE solves in 2D and 3D, respectively. The large number of PDE solves underscores the importance of computing a factorization (or a good preconditioner in large-scale problems) apriori.

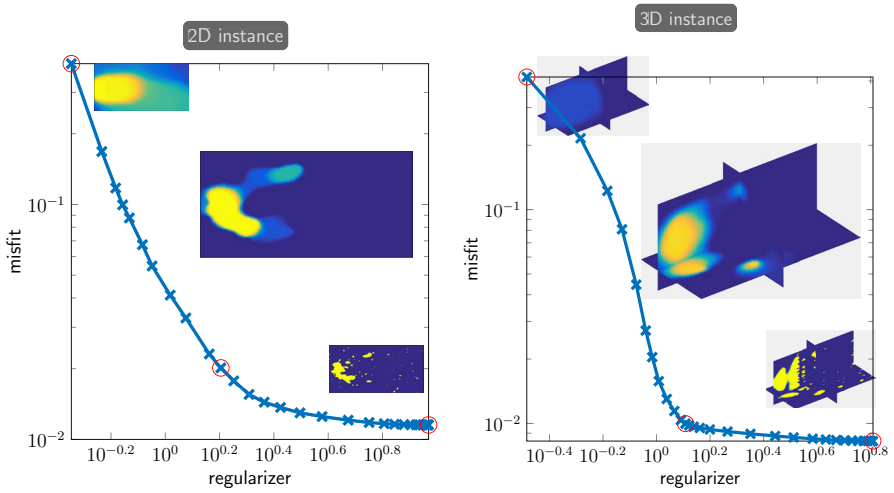

Fig. 9 L-curve plots for the relaxed optimization problem (7) for the two-dimensional instance (left) and three-dimensional instance (right). In both cases, we solve the relaxed problem for 30 α values logarithmically spaced between 1 and 10^{-6}. We plot the value of the regularizer and misfit at the computed solution in a loglog plot. To highlight the impact of α on the smoothness of the reconstructed images, we provide snapshots of the reconstructed source at the extremal values and one value that provides a good trade-off (values are marked with a circle)

References

Abhishek K, Leyffer S, Linderoth JT (2010) FilMINT: an outer-approximation-based solver for nonlinear mixed integer programs. INFORMS J Comput 22:555–567. https://doi.org/10.1287/ijoc.1090.0373

Achterberg T (2005) SCIP—a framework to integrate constraint and mixed integer programming. Technical Report ZIB-Report 04-19, Konrad-Zuse-Zentrum für Informationstechnik Berlin, Takustr. 7, Berlin

Achterberg T (2009) Scip: solving constraint integer programs. Math Program Comput 1(1):1–41

Akcelik V, Biros G, Draganescu A, Ghattas O, Hill J, van Bloemen Waanders B (2005) Dynamic data-driven inversion for terascale simulations: Real-time identification of airborne contaminants. In: Proceedings of SC2005, Seattle, WA

Akçelik V, Biros G, Ghattas O, Hill J, Keyes D, van Bloemen Waanders B (2006) Parallel algorithms for PDE-constrained optimization. In: Parallel processing for scientific computing. SIAM, pp 291–322

Akrotirianakis I, Maros I, Rustem B (2001) An outer approximation based branch-and-cut algorithm for convex 0–1 MINLP problems. Optim Methods Softw 16:21–47

Ascher UM, Haber E (2001) Grid refinement and scaling for distributed parameter estimation problems. Inverse Prob 17:571–590

Balas E (1975) Facets of the knapsack polytope. Math Program 8:146–164

Bangerth W, Klie H, Matossian V, Parashar M, Wheeler MF (2005) An autonomic reservoir framework for the stochastic optimization of well placement. Cluster Comput 8(4):255–269

Bangerth W, Klie H, Wheeler M, Stoffa P, Sen M (2006) On optimization algorithms for the reservoir oil well placement problem. Comput Geosci 10(3):303–319. https://doi.org/10.1007/s10596-006-9025-7

Bartlett R, Heinkenschloss M, Ridzal D, van Bloemen Waanders B (2005) Domain decomposition methods for advection dominated linear-quadratic elliptic optimal control problems. Comput Methods Appl Mech Eng 195(44–47):6428–6447

Bellout MC, Ciaurri DE, Durlofsky LJ, Foss B, Kleppe J (2012) Joint optimization of oil well placement and controls. Comput Geosci 16(4):1061–1079

Belotti P, Kirches C, Leyffer S, Linderoth J, Luedtke J, Mahajan A (2013) Mixed-integer nonlinear optimization. Acta Numerica 22:1–131. https://doi.org/10.1017/S0962492913000032

Belotti P, Kirches C, Leyffer S, Linderoth J, Luedtke J, Mahajan A (2013) Mixed integer nonlinear programming. Acta Numerica 22:1–131

Bendsøe M, Sigmund O (2004) Topological optimization theory. Springer, Berlin

Bezanson J, Karpinski S, Shah VB, Edelman A (2012) Julia: a fast dynamic language for technical computing. arXiv preprint arXiv:1209.5145

Biegler L, Ghattas O, Heinkenschloss M, van Bloemen Waanders B (eds) (2001) Large-scale PDE-constrained optimization, vol 30. Lecture notes in computational science and engineering. Springer, Berlin

Biegler L, Ghattas O, Heinkenschloss M, Keyes D, van Bloemen Waanders B (eds) (2007) Real-time PDE-constrained optimization. SIAM, Philadelphia

Biegler LT, Ghattas O, Heinkenschloss M, van Bloemen Waanders B (2003) Large-scale PDE-constrained optimization: an introduction. Large-scale PDE-constrained optimization. Springer, Berlin, pp 3–13

Biros G, Ghattas O (2005) Parallel Lagrange-Newton-Krylov-Schur Methods for PDE-constrained optimization. Part I: The Krylov-Schur Solver. SIAM J Sci Comput 27(2):687–713

Biros G, Ghattas O (2005) Parallel Lagrange-Newton-Krylov-Schur Methods for PDE-Constrained Optimization, Part II: The Lagrange-Newton Solver, and its Application to Optimal Control of Steady Viscous Flows. SIAM J Sci Comput 27(2):714–739

Bonami P, Biegler L, Conn A, Cornuéjols G, Grossmann I, Laird C, Lee J, Lodi A, Margot F, Sawaya N, Wächter A (2008) An algorithmic framework for convex mixed integer nonlinear programs. Discrete Optim 5(2):186–204

Bonami P, Cornuéjols G, Lodi A, Margot F (2009) A feasibility pump for mixed integer nonlinear programs. Math Program 119:331–352

Bonami P, Lee J (2007) Bonmin user's manual. Numer Math 4:1–32

Borzi A (2007) High-order discretization and multigrid solution of elliptic nonlinear constrained optimal control problems. J Comp Appl Math 200:67–85

Borzì A, Schulz V (2009) Multigrid methods for PDE optimization. SIAM Rev 51(2):361–395. https://doi.org/10.1137/060671590
Burer S, Letchford A (2012) Non-convex mixed-integer nonlinear programming: a survey. Surv Oper Res Manag Sci 17:97–106
Bürger A, Zeile C, Hahn M, Altmann-Dieses A, Sager S, Diehl M (2020) pycombina: an open-source tool for solving combinatorial approximation problems arising in mixed-integer optimal control. In: IFAC World Congress 2020. Accepted
Bussieck MR, Pruessner A (2003) Mixed-integer nonlinear programming. SIAG/OPT Views-and-News 14(1):19–22
Çezik MT, Iyengar G (2005) Cuts for mixed 0–1 conic programming. Math Program 104:179–202
Chan TF, Shen J (2005) Image Processing and Analysis. Society for Industrial and Applied Mathematics. http://bookstore.siam.org/ot94/
Committee T (2010) Advanced fueld pellet materials and fuel rod design for water cooled reactors. Technical report, International Atomic Energy Agency
Conn AR, Gould NI, Toint PL (2000) Trust region methods, vol 1. SIAM, Philadelphia
Costa MFP, Rocha AMA, Francisco RB, Fernandes EM (2016) Firefly penalty-based algorithm for bound constrained mixed-integer nonlinear programming. Optimization 65(5):1085–1104
Csurka G, Larlus D, Perronnin F, Meylan F (2013) What is a good evaluation measure for semantic segmentation? In: BMVC, vol 27. Citeseer
Dakin RJ (1965) A tree search algorithm for mixed programming problems. Comput J 8:250–255
Danna E, Rothberg E, LePape C (2005) Exploring relaxation induced neighborhoods to improve MIP solutions. Math Program 102:71–90
De Wolf D, Smeers Y (2000) The gas transmission problem solved by an extension of the simplex algorithm. Manage Sci 46:1454–1465
Donovan G, Rideout D (2003) An integer programming model to optimize resource allocation for wildfire containment. Forest Sci 61(2):331–335
Drewes S (2009) Mixed integer second order cone programming. Ph.D. thesis, Technische Universität Darmstadt
Drewes S, Ulbrich S (2012) Subgradient based outer approximation for mixed integer second order cone programming. In: Mixed integer nonlinear programming, The IMA volumes in mathematics and its applications, vol 154. Springer, New York, pp 41–59. ISBN 978-1-4614-1926-6
Dunning I, Huchette J, Lubin M (2017) Jump: a modeling language for mathematical optimization. SIAM Rev 59(2):295–320
Duran MA, Grossmann I (1986) An outer-approximation algorithm for a class of mixed-integer nonlinear programs. Math Program 36:307–339
Ehrhardt K, Steinbach MC (2005) Nonlinear optimization in gas networks. Springer, Berlin
Engl HW, Hanke M, Neubauer A (1996) Regularization of inverse problems, mathematics and its applications, vol 375. Kluwer Academic Publishers Group, Dordrecht. https://doi.org/10.1007/978-94-009-1740-8
Fipki S, Celi A (2008) The use of multilateral well designs for improved recovery in heavy oil reservoirs. In: IADV/SPE Conference and Exhibition. SPE, Orlanda, Florida
Fischetti M, Glover F, Lodi A (2005) The feasibility pump. Math Program 104:91–104
Fischetti M, Lodi A (2002) Local branching. Math Program 98:23–47
Floudas CA (2000) Deterministic global optimization: theory, algorithms and applications. Kluwer Academic Publishers, Berlin
Fourer R, Gay DM, Kernighan BW (1993) AMPL: a modeling language for mathematical programming. The Scientific Press, Beijing
Frangioni A, Gentile C (2006) Perspective cuts for a class of convex 0–1 mixed integer programs. Math Program 106:225–236
Fügenschuh A, Geißler B, Martin A, Morsi A (2009) The transport PDE and mixed-integer linear programming. In: Dagstuhl Seminar Proceedings. Schloss Dagstuhl-Leibniz-Zentrum für Informatik
Garmatter D, Porcelli M, Rinaldi F, Stoll M (2019) Improved penalty algorithm for mixed integer PDE constrained optimization (MIPDECO) problems. arXiv preprint arXiv:1907.06462
Geoffrion AM (1972) Generalized Benders decomposition. J Optim Theory Appl 10(4):237–260
Gerdts M, Sager S (2012) Mixed-integer DAE optimal control problems: necessary conditions and bounds. In: Biegler L, Campbell S, Mehrmann V (eds) Control and optimization with differential-algebraic constraints. SIAM, Berlin, pp 189–212

Golub GH, Heath M, Wahba G (1979) Generalized cross-validation as a method for choosing a good ridge parameter. Technometrics 21(2):215–223. https://doi.org/10.1080/00401706.1979.10489751
Grossmann IE (2002) Review of nonlinear mixed-integer and disjunctive programming techniques. Optim Eng 3:227–252
Grossmann IE, Kravanja Z (1997) Mixed-integer nonlinear programming: a survey of algorithms and applications. In: Biegler ACLT, Coleman TF, Santosa F (eds) Large-scale optimization with applications, Part II: optimal design and control. Springer, New York
Gugat M, Leugering G, Martin A, Schmidt M, Sirvent M, Wintergerst D (2018) MIP-based instantaneous control of mixed-integer PDE-constrained gas transport problems. Comput Optim Appl 70(1):267–294
Günlük O, Linderoth J (2008) Perspective relaxation of mixed integer nonlinear programs with indicator variables. In: Lodi A, Panconesi A, Rinaldi G (eds.) IPCO 2008: The thirteenth conference on integer programming and combinatorial optimization, vol 5035, pp. 1–16
Gunzburger MD (2003) Perspectives in flow control and optimization, advances in design and control, vol 5. Society for Industrial and Applied Mathematics (SIAM), Philadelphia
Guo Jy Lu, Wx Yang Qc, Ts Miao (2019) The application of 0–1 mixed integer nonlinear programming optimization model based on a surrogate model to identify the groundwater pollution source. J Contam Hydrol 220:18–25
Gupta OK, Ravindran A (1985) Branch and bound experiments in convex nonlinear integer programming. Manage Sci 31:1533–1546
Haber E, Ascher UM (2001) Preconditioned all-at-once methods for large, sparse parameter estimation problems. Inverse Prob 17:1847–1864
Haber E, Oldenburg D (2000) A GCV based method for nonlinear ill-posed problems. Comput Geosci 4:41–63. https://doi.org/10.1023/A:1011599530422
Hahn M, Leyffer S, Zavala VM (2017) Mixed-Integer PDE-Constrained Optimal Control of Gas Networks. Technical Report Preprint ANL/MCS-P7095-0817, Mathematics and Computer Science Division, Argonne National Laboratory
Hahn M, Sager S, Leyffer S (2020) Binary optimal control by trust-region steepest descent. http://www.optimization-online.org/DB_FILE/2020/01/7589.pdf. Submitted for publication
Hansen PC (1998) Rank-deficient and discrete ill-posed problems. SIAM monographs on mathematical modeling and computation. Society for Industrial and Applied Mathematics (SIAM), Philadelphia. https://doi.org/10.1137/1.9780898719697
Hante FM (2017) Relaxation methods for hyperbolic PDE mixed-integer optimal control problems. Opt Control Appl Methods 38(6):1103–1110
Hante FM, Sager S (2013) Relaxation methods for mixed-integer optimal control of partial differential equations. Comput Optim Appl 55(1):197–225
Hazra SB, Schulz V (2006) Simultaneous pseudo-timestepping for aerodynamic shape optimization problems with state constraints. SIAM J Sci Comput 28:1078–1099
Heinkenschloss M, Ridzal D (2008) Lecture notes in computational science and engineering, chapter. Integration of sequential quadratic programming and domain decomposition methods for nonlinear optimal control problems. Springer, Berlin
Herrmann M, Herzog R, Schmidt S, Vidal-Núñez J, Wachsmuth G (2018) Discrete total variation with finite elements and applications to imaging. arXiv.org
Hintermuller M, Vicente LN (2005) Space mapping for optimal control of partial differential equations. SIAM J Opt 15:1002–1025
Hinze M, Pinnau R, Ulbrich M, Ulbrich S (2009) Optimization with PDE constraints. Springer, Berlin
Horowitz E, Sahni S (1974) Computing partitions with applications to the knapsack problem. J ACM 21:277–292
Jeroslow RG (1973) There cannot be any algorithm for integer programming with quadratic constraints. Oper Res 21(1):221–224
Jung M (2013) Relaxations and approximations for mixed-integer optimal control. Ph.D. thesis, University Heidelberg. http://www.ub.uni-heidelberg.de/archiv/16036
Jung M, Reinelt G, Sager S (2015) The Lagrangian relaxation for the combinatorial integral approximation problem. Optim Methods Softw 30(1):54–80
Kannan R, Monma C (1978) On the computational complexity of integer programming problems. In: Henn R, Korte B, Oettli W (eds.) Optimization and Operations Research. Lecture notes in economics and mathematical systems, vol 157. Springer, Berlin, pp 161–172

Laird CD, Biegler LT, van Bloemen Waanders B, Bartlett RA (2005) Time dependent contaminant source determination for municipal water networks using large scale optimization. ASCE J Water Res Mgt Plan pp. 125–134

Lee J, Leyffer S (eds) (2011) Mixed integer nonlinear programming, IMA volume in mathematics and its applications. Springer, New York

Legg M, Davidson R, Nozick L (2013) Optimization-based regional hurricane mitigation planning. J Infrastruct Syst 19:1–11

Leyffer S, Munson T, Wild S, van Bloemen Waanders B, Ridzal D (2013) Mixed-integer PDE-constrained optimization. Position Paper #15 submitted in response to the ExaMath13 Call for Position Papers. https://collab.mcs.anl.gov/download/attachments/7569466/examath13_submission_15.pdf

Lucidi S, Rinaldi F (2013) An exact penalty global optimization approach for mixed-integer programming problems. Optim Lett 7(2):297–307

Mahajan A, Leyffer S, Linderoth J, Luedtke J, Munson T (2011) MINOTAUR: a toolkit for solving mixed-integer nonlinear optimization. wiki-page. http://wiki.mcs.anl.gov/minotaur

Mahajan A, Leyffer S, Linderoth J, Luedtke J, Munson T (2017) Minotaur: a mixed-integer nonlinear optimization toolkit. Technical reprt, ANL/MCS-P8010-0817, Argonne National Laboratory

Manns P, Kirches C (2018) Multi-dimensional sum-up rounding for elliptic control systems. DFG SPP 1962 Preprint . https://spp1962.wias-berlin.de/preprints/080.pdf. (submitted to SIAM Journal on Numerical Analysis)

Manns P, Kirches C (2019) Improved regularity assumptions for partial outer convexification of mixed-integer PDE-constrained optimization problems. ESAIM: control, optimisation and calculus of variations. http://www.optimization-online.org/DB_HTML/2018/04/6585.html. (accepted)

Manns P, Kirches C (2019) Multi-dimensional sum-up rounding using Hilbert curve iterates. In: Proceedings in applied mathematics and mechanics. (accepted)

Martello S, Pisinger D, Toth P (1999) Dynamic programming and strong bounds for the 0–1 knapsack problem. Manage Sci 45(3):414–424

Martello S, Toth P (1988) A new algorithm for the 0–1 knapsack problem. Manage Sci 34:633–644

Martello S, Toth P (1990) Knapsack problems: algorithms and computer implementations. Wiley, Chichester

Martin A, Möller M, Moritz S (2006) Mixed integer models for the stationary case of gas network optimization. Math Program 105:563–582

Nannicini G, Belotti P, Liberti L (2008) A local branching heuristic for MINLPs. arXiv:0812.2188v1 [math.CO]

Nocedal J, Wright S (2000) Numerical optimization. Springer, Berlin

Ozdogan U (2004) Optimization of well placement under time-dependent uncertainty. Master's thesis, Stanford University

Pisinger D (1995) An expanding-core algorithm for the exact 0–1 knapsack problem. Eur J Oper Res 87:175–187

Pisinger D, Toth P (1998) Knapsack problems. In: Du DZ, Pardalos P (eds) Handbook of combinatorial optimization. Kluwer, Berlin, pp 1–89

Quesada I, Grossmann IE (1992) An LP/NLP based branch-and-bound algorithm for convex MINLP optimization problems. Comput Chem Eng 16:937–947

Reinke CM, la Mata Luque TMD, Su MF, Sinclair MB, El-Kady I (2011) Group-theory approach to tailored electromagnetic properties of metamaterials: an inverse-problem solution. Phys Rev E 83(6):06660–1–18 3

Rudin LI, Osher S, Fatemi E (1992) Nonlinear total variation based noise removal algorithms. Physica D 60(1–4):259–268. https://doi.org/10.1016/0167-2789(92)90242-F

Ruthotto L, Treister E, Haber E (2017) jinv-a flexible julia package for pde parameter estimation. SIAM J Sci Comput 39(5):S702–S722

Sager S (2006) Numerical methods for mixed–integer optimal control problems. Ph.D. thesis, Universität Heidelberg. https://mathopt.de/PUBLICATIONS/Sager2005.pdf

Sager S (2009) Reformulations and algorithms for the optimization of switching decisions in nonlinear optimal control. J Process Control 19(8):1238–1247 https://mathopt.de/PUBLICATIONS/Sager2009b.pdf

Sager S, Bock H, Diehl M (2012) The integer approximation error in mixed-integer optimal control. Math Program A 133(1–2):1–23 https://mathopt.de/PUBLICATIONS/Sager2012a.pdf

Sager S, Jung M, Kirches C (2011) Combinatorial integral approximation. Math Methods Oper Res 73(3):363–380. https://doi.org/10.1007/s00186-011-0355-4

Scherzer O, Grasmair M, Grossauer H, Haltmeier M, Lenzen F (2013) Variational methods in imaging. Springer, Berlin
Sharma S (2013) Mixed-integer nonlinear programming heuristics applied to a shale gas production optimization problem. Master's thesis, Norwegian University of Science and Technology. http://www.diva-portal.org/smash/get/diva2:646797/FULLTEXT01.pdf
Sigmund O, Maute K (2013) Topological optimization approaches. Struct Multidiscip Opt 48:1031–1055
Sigmund O, Maute K (2013) Topology optimization approaches: a comparative review. Struct Multidiscip Optim 48(6):1031–1055
Simon R (2008) Multigrid solver for saddle point problems in PDE-constrained optimization. Ph.D. thesis, Johannes Kepler Universitat Linz
Steinbach MC (2007) On PDE solution in transient optimization of gas networks. J Comput Appl Math 203(2):345–361
Still C, Westerlund T (2006) Solving convex MINLP optimization problems using a sequential cutting plane algorithm. Comput Optim Appl 34(1):63–83
Stubbs R, Mehrotra S (1999) A branch-and-cut method for 0–1 mixed convex programming. Math Program 86:515–532
Tawarmalani M, Sahinidis NV (2002) Convexification and global optimization in continuous and mixed-integer nonlinear programming: theory, algorithms, software, and applications. Kluwer Academic Publishers, Boston
Vogel CR (1999) Sparse matrix computations arising in distributed parameter identification. SIAM J Matrix Anal Appl 20:1027–1037
Vogel CR (2002) Computational methods for inverse problems. Soc Ind Appl Math. doi 10(1137/1):9780898717570
You F, Leyffer S (2010) Oil spill response planning with MINLP. SIAG/OPT Views-and-News 21(2):1–8
You F, Leyffer S (2011) Mixed-integer dynamic optimization for oil-spill response planning with integration of a dynamic oil weathering model. AIChe J. https://doi.org/10.1002/aic.12536
Yu J, Anitescu M (2019) Multidimensional sum-up rounding for integer programming in optimal experimental design. Mathe. https://doi.org/10.1007/s10107-019-01421-z
Zavala VM (2014) Stochastic optimal control model for natural gas networks. Comput Chem Eng 64:103–113
Zhang P, Romero D, Beck J, Amon C (2013) Integration of AI and OR techniques in constraint programming for combinatorial optimization problems, chapter solving wind farm layout optimization with mixed integer programming and constraint programming. Springer, Berlin