Data-driven coordination of expensive subproblems in enterprise-wide optimization

Damien van de Berg[1], Panagiotis Petsagkourakis[1], Nilay Shah[2], and Ehecatl Del Rio-Chanona[1]

[1]Imperial College London
[2]Imperial College

July 1, 2022

Abstract

While decomposition techniques in mathematical programming are usually designed for numerical efficiency, coordination problems within enterprise-wide optimization are often limited by organizational rather than numerical considerations. We propose a 'data-driven' coordination framework which manages to recover the same optimum as the equivalent centralized formulation while allowing coordinating agents to retain autonomy, privacy, and flexibility over their own objectives, constraints, and variables. This approach updates the coordinated, or shared, variables based on derivative-free optimization (DFO) using only coordinated variables to agent-level optimal subproblem evaluation 'data'. We compare the performance of our framework using different DFO solvers (CUATRO, Py-BOBYQA, DIRECT-L, GPyOpt) against conventional distributed optimization (ADMM) on three case studies: collaborative learning, facility location, and multi-objective blending. We show that in low-dimensional and nonconvex subproblems, the exploration-exploitation trade-offs of DFO solvers can be leveraged to converge faster and to a better solution than in distributed optimization

Data-driven coordination of expensive subproblems in enterprise-wide optimization

Damien van de Berg[a], Panagiotis Petsagkourakis[a], Nilay Shah[a,*], Ehecatl Antonio del Rio-Chanona[a,**]

[a]*Sargent Centre for Process Systems Engineering, Roderic Hill Building South Kensington Campus London, SW7 2AZ, United Kingdom*

Abstract

While decomposition techniques in mathematical programming are usually designed for numerical efficiency, coordination problems within enterprise-wide optimization are often limited by organizational rather than numerical considerations. We propose a 'data-driven' coordination framework which manages to recover the same optimum as the equivalent centralized formulation while allowing coordinating agents to retain autonomy, privacy, and flexibility over their own objectives, constraints, and variables. This approach updates the coordinated, or shared, variables based on derivative-free optimization (DFO) using only coordinated variables to agent-level optimal subproblem evaluation 'data'. We compare the performance of our framework using different DFO solvers (CUATRO, Py-BOBYQA, DIRECT-L, GPyOpt) against conventional distributed optimization (ADMM) on three case studies: collaborative learning, facility location, and multi-objective blending. We show that in low-dimensional and nonconvex subproblems, the exploration-exploitation trade-offs of DFO solvers can be leveraged to converge faster and to a better solution than in distributed optimization.

Keywords: Expensive black-box; Data-driven optimization; Distributed optimization; Coordination

*Corresponding Author

**Corresponding Author

Email addresses: `n.shah@imperial.ac.uk` (Nilay Shah), `a.del-rio-chanona@imperial.ac.uk` (Ehecatl Antonio del Rio-Chanona)

1. Introduction

Companies within the process industries rely on mathematical optimization for their operations to remain competitive in an environment of increasingly stringent safety, environmental, and economic requirements [31]. This gives rise to the field of enterprise-wide optimization (EWO) with the ultimate goal to coordinate all decision-making within a company [32, 33]. EWO involves the integration of units across 1) all hierarchical levels of decision-making (from design, planning, scheduling, to control), and 2) all geographically distributed (plants, warehouses, etc.) or functional (sourcing, manufacturing, distribution) units. Conventionally, these separate entities are solved sequentially via one-way information flow [20]. For instance, higher-level planning might determine the setpoints of lower-level scheduling without explicitly accounting for lower-level constraints; or geographically separated plants might adjust their operations to accommodate the needs of other bottleneck plants in the value chain. These heuristics in coordinated decision-making, while sometimes necessary for practicality and tractability, do not guarantee optimality of the integrated problem. However, integrated model-based optimization traditionally requires the solution of a larger-scale centralized optimization model, which quickly becomes computationally intractable in the number of decision variables and constraints [20]. A centralized formulation could also in practice be obviated by organizational complexity (antitrust, privacy, . . .).

One way to alleviate the computational burden of model-based integration is to relax constraints, or replace detailed formulations with surrogate models that are easier to handle by numerical solvers [9, 44, 10]. Usually, this would come at the expense of a degradation in solution quality. However, since EWO aims to coordinate previously decoupled decision-making, the resulting optimization formulations present mathematical structures that can be exploited. The resulting problems comprise few complicating variables and constraints that lend themselves well to decomposition and distributed optimization schemes [56]. Decomposition techniques consist of the iterative solution of a relaxed upper- and reduced-order lower-level problem which can theoretically achieve the same solution quality as the original formulation, while saving computational time. Bilevel [39, 21] and Benders decomposition [58, 72, 62] are among the most prominent techniques for addressing complicating variables, and typically decompose problems over time, and stochastic realizations of uncertainty respectively. Lagrangean decomposition is particularly useful for tackling complicating constraints, as well as complicating variables by reformulation. As such, it is also useful for decomposing problems by time, space, or products [40, 54, 70, 68].

Distributed optimization builds on the concepts of dual decomposition techniques (such as Lagrangean decomposition). It has many applications in problems that are separated by complicating constraints, such as the integration of geographically dispersed warehouses or plants along a supply chain [67]. The Alternating Direction Method of Multipliers (ADMM) has received special attention as a powerful tool enabling considerable computational savings using minimal information exchange, especially in convex optimization [13]. ADMM repeatedly iterates between the solution of private, localized, lower-level subproblems, and an upper-level problem whose aim is to coordinate the solutions of the private subproblems. The possibility for solving the subproblems in parallel gives rise to significant potential computational savings. Despite often being applied in practice, ADMM loses its convergence guarantees on nonconvex problems [59]. Another drawback of ADMM is that the method practically only leads to computational savings compared to the *centralized* solution under special conditions, namely when the problem is decomposed into *numerous, convex* subproblems [13].

Similarly to how the convergence of first-order gradient descent solvers can be improved using acceleration or momentum, there are several ways to speed up the convergence of ADMM using similar schemes [14]. Houska et al. [37] have proposed ALADIN, an algorithm to address ADMM's shortcomings: it speeds up - and includes theoretical conditions for - global convergence to local minimizers on nonconvex problems. ALADIN iterates between the parallel optimization of subproblems and sequential quadratic programming (SQP) steps for the coordination around the local subproblem solutions.

While distributed optimization seems promising from a computational perspective, much of the literature discussing model-based integration in EWO with relevant solution techniques fails to consider communication and business considerations that could hinder their practical applicability [25, 71, 80]. Distributed optimization is often approached using a top-down coordination approach: Starting from a centralized model, a decomposition is applied that is expected to lead to computational savings. This presupposes that previously decoupled decision-makers 1) are willing to share their local models; 2) accept the risk of foregoing a certain degree of autonomy, flexibility, and Nash equilibria for the pursuit of the 'social optimum' of the centralized model; and 3) even have access to known, differentiable expressions as part of their optimization model. Due to a significant increase in computational power over the past few decades, software and organizational rather than numerical considerations might become the bottleneck in the integration of computational decision-making [6]. In fact, current decision-making architectures were often established within a legal and organizational framework when operations were (and often still are) guided by heuristics rather than numerical optimization. As such, the considered problem is rendered into a multi-agent coordination problem where each agent might represent a separate legal entity with its own autonomy, agenda, technical constraints, and organizational considerations [36, 22].

The organizational context matters when choosing the best coordination scheme. When all agents are willing to collaborate and share differentiable model expressions, powerful distributed optimization techniques can be leveraged for optimal numerical efficiency [23]. When coordinating (not necessarily collaborating) agents only have access to black-box simulation tools for decision-making, 'data-driven' or 'black-box' optimization tools need to be adapted for the coordination. There are many reviews on data-driven or derivative-free optimization algorithms [48, 3]. Some state-of-the-art methods have also been benchmarked on typical process systems engineering (PSE) applications in [74], and have been introduced to solve multilevel problems in [84, 8, 7]. van de Berg et al. [73] show that derivative-free optimization can be used for the data-driven coordination of black-box subproblems in multi-objective problems arising in PSE.

In this work, we build upon van de Berg et al. [73] to investigate whether derivative-free optimization (DFO) can be used as a viable alternative to distributed optimization solvers in the following coordination problems: each agent is willing to collaborate (i.e. sacrifice suboptimality in their own objective for a 'greater good') and has their own decision-making model, which does *not* have to be white-box - it could be the black-box result of a third-party, proprietary simulation software. In the context of EWO, this problem might arise when plants along the same value chain need to coordinate on material streams given that each plant has a separate objective that they optimize with the help of third-party software. In this case, the model is not readily exploitable for gradient information, such that solvers like ALADIN cannot be used as it requires exact first-order gradient information of the subproblems for its SQP step. The question arises if or data-driven optimization approaches perform best for these kinds of scenarios.

While the performance of different distributed optimization algorithms have been compared with each other and with a centralized solution [69], we thoroughly investigate under which conditions data-driven optimization outperforms typical distributed optimization solvers such as ADMM. As discussed in van de Berg et al. [73], any (potentially imperfect) gradient information becomes increasingly valuable in higher-dimensional decision spaces. Since ADMM's upper-level coordination step involves subgradient information, we only expect DFO to be competitive under specific conditions, i.e. when the number of complicating variables is few relative to the number of private decision variables. Our aim is not to outperform centralized solution methods. In the methods we compare, computational efficiency is sacrificed for agent privacy, autonomy, and flexibility.

This paper is organized as follows: In Section 2, we illustrate our data-driven methodology along with conventional ADMM. We also explain our choice of DFO algorithms (CUATRO, Py-BOBYQA, DIRECT-L and GPyOpt) for our data-driven coordination problems. In Section 3, we then introduce a motivating mathematical test function and three case studiesIn Section 4, we then present and discuss the convergence of all algorithms.We also investigate how algorithm convergence changes with the number of complicating variables, the number of coordinating agents, and the topology of the subproblem solution space.

2. Methodology

2.1. Problem statement

We are interested in solving the equivalent of the following centralized, integrated coordination problem:

$$\begin{aligned} \min_{\mathbf{x} \in \mathbb{X}} \quad & \sum_{i=1}^{N} f_i(\mathbf{x}_{i,p}, \mathbf{z}) \\ \text{s.t.} \quad & \mathbf{g}_i(\mathbf{x}_{i,p}, \mathbf{z}) \leq \mathbf{0}, \quad i = 1 \dots N \end{aligned} \tag{1}$$

where $\mathbf{x} \in \mathbb{X} \subset \mathbb{R}^{n_x}$ refers to the decision variable vector within feasibility set $\mathbb{X}$. As such, $\mathbf{x}$ includes not only the 'local', private decision variables $\mathbf{x}_{i,p} \in \mathbb{X}_{i,p} \subset \mathbb{R}^{n_{x_i}}$ of all N agents, but also the 'global', shared variables $\mathbf{z}$ within the feasibility set $\mathcal{Z} \subset \mathbb{R}^{n_z}$. As such, the complete decision vector comprises the following elements: $\mathbf{x} = [\mathbf{x}_{1,p}, \dots, \mathbf{x}_{N,p}, \mathbf{z}]$. The optimization is also subject to N local agent constraints $\mathbf{g}_i : \mathbb{R}^{n_{x_i}} \times \mathbb{R}^{n_z} \to \mathbb{R}^{n_{g_i}}$.

This generic problem formulation also implicitly allows for the inclusion of equality constraints in Eq. (1) through a reduction in the degrees of freedom of the decision variables, or through an equivalent reformulation into two inequalities. Additionally, Eq. (1) also allows for the incorporation of global, or shared, constraints and objectives. We would call any constraint g_{global} 'shared' if it only depends on the shared variables $\mathbf{z}$. Similarly, shared objective terms might either manifest as a separate term f_{global} in a single agent objective, or be incorporated into the objectives of any $M \leq N$ agents as $\frac{f_{global}(\mathbf{z})}{M}$.

2.2. Problem reformulation

Problem (1) can be reformulated into:

$$
\begin{aligned}
\min_{\mathbf{z}\in\mathcal{Z}} \quad \min_{\mathbf{x}_{i,p},\, i=1,\ldots,N} \quad & \sum_{i=1}^{N} f_i(\mathbf{x}_{i,p}, \mathbf{z}) \\
\text{s.t.} \quad & \mathbf{g}_i(\mathbf{x}_{i,p}, \mathbf{z}) \leq \mathbf{0}
\end{aligned}
\tag{2}
$$

After fixing $\mathbf{z}$, the problem becomes block separable, which makes the problem amenable to decomposition and distributed optimization. This becomes evident when rewriting Eq. (2) as its equivalent constrained (bi-level) optimization problem in (3). The coordination step involves an update in the shared variables $\mathbf{z}$. At each iteration, the subproblems are solved *in private* to find the optimal objective $F_i(\mathbf{z})$ and set of private variables $\mathbf{x}^*_{i,p}$ corresponding to a set of shared variables $\mathbf{z}$. Agents can maintain autonomy and flexibility by deciding on their own objective and constraint functions which they do not need to share with other agents. The only information that agents share with a third-party coordinator is the optimal set of private variables and local copy of shared variables $\mathbf{x}^*_{i,p}$ and $\mathbf{z}^*_i$ (2.3) *or* the optimal objective $f^*(\cdot)$ (2.4) corresponding to a suggested set of shared variables $\mathbf{z}$. While for simplicity's sake we assume that the subproblems are solved to global optimality, we do not assume that the lower-level problems are solved by exploiting known expressions - The subproblem optimization could involve black-box queries such as proprietary simulations.

$$
\begin{aligned}
\min_{\mathbf{z}\in\mathcal{Z}} \quad & F(\mathbf{z}) \\
\text{s.t.} \quad & F(\mathbf{z}) = \sum_{i}^{N} F_i(\mathbf{z}) \\
& F_i(\mathbf{z}) = \min_{\mathbf{x}_{i,p}\in\mathbb{X}_i} \quad f_i(\mathbf{x}_{i,p}, \mathbf{z}) \\
& \qquad\qquad \text{s.t.} \quad \mathbf{g}_i(\mathbf{x}_{i,p}, \mathbf{z}) \leq \mathbf{0}
\end{aligned}
\tag{3}
$$

2.3. ADMM by consensus

The conventional method that our proposed approach is benchmarked against is ADMM in its consensus form as found in (D.4) and presented in Algorithm 1. After initialization (**step 1**), ADMM iterates over **steps 2-7** until the evaluation budget is exhausted: This involves the solution of subproblems in private and parallel (**steps 3-5**) to get the local copy of shared variables $\mathbf{z}_i$, and an update in the shared variables $\mathbf{z}$ and scaled dual variables $\mathbf{u}_i$ based on $\mathbf{z}_i$ (**step 6**).

In **step 4**, each agent optimizes their copy of shared/complicating variables $\mathbf{z}_i$ that minimizes their private objective function while penalizing any deviation from the *suggested* value of the complicating variables $\mathbf{z}^k$:

$$\mathbf{x}_{i,p}^{k+1}, \mathbf{z}_i^{k+1} \quad \leftarrow \quad F_i(\mathbf{z}^k) \quad = \underset{\mathbf{x}_{i,p}, \mathbf{z}_i}{\arg\min}\, . f_i(\mathbf{x}_{i,p}, \mathbf{z}_i) + \frac{\rho}{2}||\mathbf{z}_i - \mathbf{z}^k + \mathbf{u}_i^k||_2^2 \quad \text{s.t.} \quad \mathbf{g}_i(\mathbf{x}_{i,p}, \mathbf{z}_i) \leq \mathbf{0} \tag{4}$$

where $||\cdot||_2$ refers to the Frobenius norm, and $\mathbf{u}_i$ to the scaled dual variables of agent i. $\frac{\rho}{2}||\mathbf{z}_i - \mathbf{z}^k + \mathbf{u}_i^k||_2^2$ is known as the proximal or penalty term and is useful for stabilizing convergence in the shared variables $\mathbf{z}$. It also makes the formulation robust against local constraints: when there is no feasible set of $\mathbf{x}_{i,p}$ that satisfy all constraints for a given $\mathbf{z}^k$, the solution converges to the nearest feasible $\mathbf{z}_i$ incurring a penalization in the objective. After all subproblems are solved, the set of suggested complicating variables is then updated to $\mathbf{z}^{k+1}$ in the coordination step (**step 6**) by averaging the set of optimal complicating variables resulting from the agent subproblems $\mathbf{z}_i^k$. While the update in the shared variables $\mathbf{z}^{k+1}$ aims to ensure asymptotic primal feasibility, the update in the dual variables $\mathbf{u}_i^{k+1}$ aims to ensure asymptotic dual feasibility. Each agent's dual variables are updated to $\mathbf{u}_i^{k+1}$ based on the difference between $\mathbf{u}_i^k$ and the local copy of shared variables $\mathbf{z}_i^{k+1}$, and could be interpreted as an integral error term often encountered in control.

Algorithm 1: Alternating Direction Method of Multipliers (ADMM) by consensus

Input: Agent objectives f_i and constraints $g_{i,k}, k = 1 \ldots n_{g_i}, i = 1 \ldots N$, Initial shared variable guess $\mathbf{z}^0$, Penalty parameter ρ, Maximum number of function evaluations $N_{f,max}$

1 **Initialisation:** Initial dual variables $\mathbf{u}_i^0 = [0, \ldots, 0]$
2 **for** $j = 0 \ldots N_{f,max} - 1$ **do**
3   **for** *agent* $i = 1 \ldots N$ *in parallel* **do**
4         $\mathbf{x}_{i,p}^{k+1}, \mathbf{z}_i^{k+1} \leftarrow F_i(\mathbf{z}^k)$, by solving lower-level problem (4)
5   **end**
6     $\mathbf{z}^{k+1} \leftarrow \frac{1}{N}\sum_{i=1}^{N} \mathbf{z}_i^{k+1}, \quad \mathbf{u}_i^{k+1} \leftarrow \mathbf{u}_i^k + \mathbf{z}_i^{k+1} - \mathbf{z}^{k+1}$;
7 **end**

A common drawback of ADMM is that it can take many iterations to converge to a high-accuracy solution [14, 30, 77]. This begs the question if the coordination step in $\mathbf{z}$ could be improved to speed up the convergence or find a better solution quality for a given evaluation budget.

2.4. Data-driven coordination

Problem (3) views the coordination formulation as a bilevel optimization instance. Derivative-free optimization (DFO) has already been used to solve for the upper-level variables in multilevel problems, and as such presents a promising alternative to ADMM's subgradient update step. The difference between the data-driven coordination framework and ADMM is illustrated in Figure 1 and the data-driven coordination framework is illustrated in more detail in Algorithm 2: After initialization (**step 1**), our framework iterates over **steps 2-11** until the evaluation budget is exhausted: In **step 3**, the upper level aims to find the set of complicating variables that minimize the objective function *subject to* the *optimal* solution in parallel of the agent-level subproblems in the private variables (**steps 4-9**).

Step 3 uses a DFO algorithm to update the shared variables $\mathbf{z}$ with the aim to minimize the 'black-box' upper-level objective $F(z)$ in Eq. (3).

$$\min_{\mathbf{z} \in \mathcal{Z}} \quad F(\mathbf{z}) \tag{5}$$

where the decision variables $\mathbf{z}$ are subject to box-bound constraints $\mathcal{Z}$. Any box-constrained derivative-free, black-box, data-driven, or 'zeroth-order' optimization algorithm can be used for the solution of the upper level [48, 3, 74]. Since the 'black-box evaluations' are the result of optimizations, these evaluations are considered expensive. The number of evaluations n_{next} that are sampled at each iteration in **step 3** depends on the exploitation-exploration trade-off as well as sampling strategy of the DFO method used.

$F(\mathbf{z})$ is obtained in **steps 4-9** in a similar manner to Eq. (3). $F_i(\mathbf{z})$ is treated as the result of private black-box simulations and $F(\mathbf{z})$ is equivalent to the sum of all optimal subproblem solutions in Eq. (4), with the exception that the objective omits any dual variables:

$$F(\mathbf{z}) = \sum_{i}^{N} F_i(\mathbf{z}) \quad \text{where} \quad F_i(\mathbf{z}) = \underset{\mathbf{x}_{i,p}, \mathbf{z}_i}{\text{min.}} f_i(\mathbf{x}_{i,p}, \mathbf{z}_i) + \frac{\rho}{2} ||\mathbf{z}_i - \mathbf{z}^k||_2^2 \quad \text{s.t.} \quad \mathbf{g}_i(\mathbf{x}_{i,p}, \mathbf{z}) \leq \mathbf{0} \tag{6}$$

The scaled dual is omitted as it only enhances convergence within the rigorous stability scheme of ADMM [53], and can even degrade convergence performance. The proximal error term again ensures robustness against local constraints.

Algorithm 2: Data-driven coordination framework

Input: Agent objectives f_i and constraints $g_{i,k}, k = 1 \dots n_{g_i}, i = 1 \dots N$, Box-bound constraints $\mathcal{Z} \in \mathbb{R}^{n_z \times 2}$ on $\mathbf{z}$, Initial shared variable guess $\mathbf{z}^0$, Penalty parameter ρ, Maximum number of function evaluations $N_{f,max}$

Initialisation: Function evaluation counter $n_f := 1, \mathbf{z}_{best} = \mathbf{z}^0, y_{best} = F(\mathbf{z}^0)$ where $F(\cdot)$ is obtained from (6), initial data sets $(Z := \{\mathbf{z}^0\}, \mathbf{y} := \{y_{best}\}$

while *($N_{f,max} \geq n_f$)* **do**

- Obtain n_{next} samples Z_{next} from DFO update step at $\mathbf{z}_{best}$ using (5) ;
- **for** $\mathbf{z}_j = \mathbf{z}_1 \dots \mathbf{z}_{next} \in Z_{next}$ **do**
  - **for** *agent $i = 1 \dots N$ in parallel* **do**
    - Obtain $F_i(\mathbf{z}_j)$ from (6)
  - **end**
  - Update datasets: $Z \leftarrow \{Z, \mathbf{z}_j\}, \mathbf{y} \leftarrow \{\mathbf{y}, y_{next}\}$, where $y_{next} = \sum_i^N F_i(\mathbf{z}_j)$
- **end**
- Update best iterate: $n_f \leftarrow n_f + n_{next}, \quad y_{best} \leftarrow \min_{j=1\dots n_f} \mathbf{y}_j, \quad \mathbf{z}_{best} \leftarrow \arg\min_{j=1\dots n_f} y_j$;

end

When the subproblems are solved to global optimality, the whole optimization problem can be solved (heuristically or rigorously) to global optimality depending on the function evaluation budget and the convergence certificate of the DFO solver. Since convergence is limited by the number of expensive subproblem calls, we do not include overly exploratory methods, such as particle swarm methods. In the next section, we explore any analogies to 'data-driven' ADMM and ALADIN when quadratic surrogates (CUATRO) are used for the DFO step. Additionally, we introduce the other DFO algorithms used, whose choice is informed by van de Berg et al. [74]: Py-BOBYQA as the trust region model-based method, DIRECT-L as the direct method, and GPyOpt for Bayesian Optimization. Figure 2 shows our selection of data-driven as well as distributed optimization algorithms, and their mutual relations.

2.4.1. Data-driven distributed optimization

While Formulation (6) is applicable for both direct and model-based DFO solvers, model-based DFO methods, can address Problem (5) by introducing surrogates $\hat{F}(\mathbf{z})$ in two different ways: One option would be to fit a single surrogate over the sum of the subproblem evaluations.

$$\min_{\mathbf{z}\in\mathcal{Z}} \quad \hat{F}(\mathbf{z}) \quad \text{where} \quad \hat{F}(\mathbf{z}) \approx \sum_{i}^{N} F_i(\mathbf{z}) \tag{7}$$

A second alternative would be to allow for separate surrogates to be fitted for each subproblem before the sum of surrogates is optimized in the upper level.

$$\min_{\mathbf{z}\in\mathcal{Z}} \quad \sum_{i}^{N} \hat{F}_i(\mathbf{z}) \quad \text{where} \quad \hat{F}_i(\mathbf{z}) \approx F_i(\mathbf{z}) \tag{8}$$

Similar to van de Berg et al. [73], we use convex quadratic surrogates ($\hat{F}(\mathbf{z}) = \mathbf{z}^\intercal A\mathbf{z} + \mathbf{b}^\intercal\mathbf{z} + c, \quad A \succcurlyeq 0 \in \mathbb{R}^{n_z \times n_z}, \mathbf{b} \in \mathbb{R}^{n_z \times 1}, c \in \mathbb{R}$) within the CUATRO framework. In this case, the approach used in Eq. (7) is similar to [49], and could be loosely referred to as 'Data-driven ADMM'. The approach in (8) could then be viewed as 'Data-driven ALADIN', with a crucial difference: ALADIN's quadratic surrogate coefficients are given by the gradient and Hessian (obtained via automatic differentiation) of a second-order Taylor expansion around the local subproblem solutions, while in our data-driven counterpart, the surrogates are obtained via quadratic regression based on the subproblem evaluations.

2.4.2. CUATRO

We modified CUATRO [74] - a quadratic trust-region surrogate-based DFO algorithm - to be used within the 'data-driven ADMM' (ADMM_CUATRO) and 'data-driven ALADIN' (ALADIN_CUATRO) framework. CUATRO is chosen as our quadratic surrogate-based DFO algorithm because it leverages 1) semidefinite programming, 2) a trust region framework, and 3) explicit constraint handling. As such, the CUATRO framework can be used flexibly.

Explicit constraint handling. When CUATRO is used with explicit constraint handling, the local copies of the shared variables $\mathbf{z}_i$ in the objective evaluation and constraints from (6) are replaced by the exact shared variables $\mathbf{z}$.

$$F_i(\mathbf{z}) = \min_{\mathbf{x}_{i,p}} . f_i(\mathbf{x}_{i,p}, \mathbf{z}) \quad \text{s.t.} \quad \mathbf{g}_i(\mathbf{x}_{i,p}, \mathbf{z}) \leq \mathbf{0} \tag{9}$$

where $F_i(\mathbf{z})$ is a tuple consisting of the objective and binary feasibility evaluation: $F_i : \mathbb{R}^{n_z} \to \mathbb{R} \times \{0, 1\}$. 1 denotes if the evaluation for $\mathbf{z}$ is feasible. This makes the subproblem less robust to local constraints, but by returning solver status as feasibility evaluations on top of objective evaluations, we can map the feasibility space in CUATRO by quadratic discrimination and hence concentrate the search on the expected feasible space. ADMM_CUATRO and ALADIN_CUATRO are used with explicit constraint handling if a feasible starting point can be found.

While we benchmark data-driven ADMM and ALADIN against conventional ADMM, we also benchmark them against other DFO algorithms for the upper-level coordination instead of CUATRO. Py-BOBYQA and GPyOpt are only used within the single-surrogate data-driven framework (7).

2.4.3. Py-BOBYQA

On top of CUATRO, we include another trust-region based method. Trust-region frameworks focus on exploitation as opposed to the more explorative Bayesian Optimization framework. van de Berg et al. [74] show that Py-BOBYQA [16, 17] can be competitive with state-of-the-art Bayesian Optimization, especially in higher-dimensional deterministic case studies. Py-BOBYQA is a Python implementation of Powell's BOBYQA. It iteratively constructs a linear-quadratic regression-interpolation model for the objective, and determines the next step by minimizing said model within a trust-region framework. The user can manipulate how many evaluations are used for each surrogate, determining if the surrogates used resemble more linear or quadratic surrogates. We use Py-BOBYQA with its standard options but enable the multiple restarts heuristic to avoid getting stuck in local minima.

2.4.4. GPyOpt

Apart from exploitative, trust-region model-based DFO solvers, we also include a Bayesian optimization (BO) implementation. BO is generally regarded as the go-to framework for black-box optimization within chemical engineering [66, 24, 55, 18, 57, 51] due to its data efficiency and ability to navigate the exploration-exploitation trade-off. As such, BO manages to make significant progress in few evaluations. However, it is known to scale poorly with the number of dimensions and evaluation budget [74]. Informed by Cartis et al. [17], we are using GPyOpt as our implementation as we prioritize convergence within the low-accuracy regime given by our tight budget. GPyOpt [5] is a Python open-source library of BO and builds on GPy, a Python framework for Gaussian process modelling. We use GPyOpt with its default hyperparameters. The interested reader is referred to Garnett [29] for more information on Gaussian Processes and Bayesian Optimization.

2.4.5. DIRECT-L

Finally, we also include a 'direct' (model-free) DFO method. Informed by van de Berg et al. [74], we choose DIRECT-L as a competitive direct solver which displays consistency in convergence and a good exploitation-exploration trade-off. This work's randomized DIRECT-L implementation is taken from the NLopt nonlinear optimization package library [41]. This implementation is based on the 1993 DIviding RECTangles algorithm for global optimization, originally written in FORTRAN [42]. DIRECT is a Lipschitzian, deterministic search algorithm, based on systematic partitioning of the search space into smaller hyperrectangles. Gablonsky and Kelley [27] then made the algorithm biased towards local search for problems that only have a few local minima. Johnson's NLopt's implementation uses a randomized version of the locally-biased DIRECT, which involves randomness in deciding on the dimension to partition along next when function evaluations are close.

2.5. Algorithms and software implementation

We use Pyomo [35, 15] as Python-based optimization software with the numerical solvers Ipopt [75] or Gurobi [34] to optimize the continuous or mixed-integer lower-level subproblems given by (4, 6, or 9). Information from these problem instances are then extracted to be used in the upper-level distributed optimization or DFO. We use readily available Python packages for GPyOpt, Py-BOBYQA, and DIRECT-L, and an in-house Python implementation of ADMM and CUATRO. The generalized framework for our proposed framework and its comparison to ADMM is found in Figure 1, while Figure 2 illustrates how the DFO methods fit into our framework. The code for the algorithms and benchmarking is available under https://github.com/OptiMaL-PSE-Lab/Data-driven-coordination.

2.6. Game-theoretical and other considerations

Coordination problems are interdisciplinary in nature, and are rooted in a rich body of literature within the field of game theory [26, 50]. While we are less interested in the game-theoretical underpinnings of these problems, we need to state some assumptions that justify our proposed method and investigated case studies. First, ADMM and our proposed 'data-driven coordination' techniques involve an upper-level, centralized 'coordination' step. This presumes the existence of a coordinator agent or software that is acting in good faith, which should be a reasonable assumption in EWO. We are also assuming that all agents are honest-but-curious, i.e. that no agent is trying to trick the coordinator or launch any adversarial 'attacks', which is the scope of a whole subfield of literature [4] .

Finally, we want to acknowledge that coordination within business settings is subject to many different kinds of other considerations: While we investigate algorithms that share as little information as possible, the coordinator-agent and indirectly agent-agent exchange requires an involved legal framework and software infrastrucutre [43]. While ADMM and our proposed data-driven coordination algorithms in principle allow for privacy-preservation, this would in practice require a thorough investigation into differential privacy and cryptography schemes. The interested reader is referred to Rodríguez-Barroso et al. [60] and [81] for a thorough discussion.

3. Case studies

Data-driven coordination is expected to shine in applications that are low-dimensional and nonconvex. As such, we start with a motivating example before presenting three EWO-specific examples.

3.1. Motivating example

We first consider the following synthetic toy problem:

$$\begin{aligned} \min_{x_1,x_2,x_3}. \quad & (x_1-7)^2 + (x_1x_3-3)^2 + (x_2+2)^+(x_2x_3-2)^2 \\ \text{s.t.} \quad & x_1 \geq 0, \quad x_1 + x_3 = 5 \\ & -10 \leq x_1, x_2, x_3 \leq 10 \end{aligned} \tag{10}$$

We can see that after fixing x_3, the problem becomes trivially separable into 2 subproblems. This means that this problem can be reformulated into a one-dimensional DFO problem. As such, we introduce z to take the place of x_3, and introduce local copies of z, namely z^I and z^{II}. Then, we penalize the deviation between z and its local copy using a proximal term in the objective:

$$\begin{aligned} 1) \quad F_1(z) = \quad & \min_{x_1,z^I}. \quad (x_1-7)^2 + (x_1z^I-3)^2 + \frac{\rho}{2}(z^I - z)^2 \\ & \text{s.t.} \quad x_1 \geq 0, \quad x_1 + z^I = 5, \quad -10 \leq x_1, z^I \leq 10 \\ 2) \quad F_2(z) = \quad & \min_{x_2,z^{II}}. (x_2+2)^2 + (x_2z_2^{II}-3)^2 + \frac{\rho}{2}(z^{II} - z)^2 \\ & \text{s.t.} -10 \leq x_2, z^{II} \leq 10 \end{aligned} \tag{11}$$

Py-BOBYQA, DIRECT-L, GPyOpt aim to find $\mathbf{z}$ that optimizes $F_1(z) + F_2(z)$. ADMM uses the same subproblems with the exception that the proximal term includes the addition of u^I and u^{II} following (4).

For a trivial problem like this, we can find a feasible starting point in the upper-level variables for Problem (10). We use $z = 4.5$ as starting point, for which we can find x_1 and x_2 in the subproblems that satisfy all of the constraints. As such, we use ADMM_CUATRO and ALADIN_CUATRO in its constrained form, meaning that we omit z_I and z^{II} as decision variables, omit the proximal term in the subproblem objective, replace z^I and z^{II} with z in the objective and constraints, and return the solver status as a binary feasibility evaluation on top of the objective as described in (9):

$$
\begin{aligned}
1) \quad F_1(z) = \quad & \min_{x_1}. \quad (x_1 - 7)^2 + (x_1 z - 3)^2 \\
& \text{s.t.} \quad x_1 \geq 0, \quad x_1 + z = 5, \quad -10 \leq x_1, z \leq 10 \\
2) \quad F_2(z) = \quad & \min_{x_2} .(x_2 + 2)^2 + (x_2 z - 3)^2 \\
& \text{s.t.} - 10 \leq x_2, z \leq 10
\end{aligned}
\tag{12}
$$

Figure 3, which plots the upper-level objective evaluation of the bilevel formulation (3) as a function of the shared variable z, shows an inflection point around $z = 3.75$ which can hinder the convergence of ADMM and hence call for our proposed methods.

3.2. Collaborative model training

3.2.1. Federated Learning

The first case study is motivated by Federated Learning (FL) [79]. FL is a subfield within Machine Learning (ML), popularized by Google, where multiple clients collaborate under the supervision of a centralized coordinator to train a model while respecting privacy considerations. As such, the aim could be to train a text prediction ML model on decentralized edge devices' (i.e. phones) data while preserving user privacy. For deep neural networks, this usually involves an iteration over the following steps as described in the FedAVG and FedSGD [52] algorithms: The model is broadcast to a selection of training agents. The agents perform a model parameter update on local data based on a stochastic gradient descent step obtained by backpropagation. These model updates are then averaged among all participating agents, potentially preceded by an encryption or differential privacy step. The interested reader is referred to Kairouz et al. [43] for an overview of typical FL challenges.

3.2.2. Cross-silo 'learning'

We are more interested in a 'cross-silo' [43] rather than 'cross-device' setting, where the number of participating agents is fewer but the primary bottleneck resides in the model update computation, rather than in communication. Additionally, first-order methods such as stochastic gradient descent may not always be applicable if models cannot be (cheaply) differentiated for gradient information (i.e. if the model to be trained is constrained, dynamic, ...). As such, we investigate a generalized coordination scheme for collaborative model training using distributed optimization or data-driven coordination. Similar to ADMM, the conventional FL scheme also involves an averaging step of the model parameters as 'shared variables'. But as opposed to FedAVG [52], the subproblem local variable update cannot be obtained under closed form. Instead, we fall back on the more general optimization formulation used in (4).

3.2.3. Case study

Our considered case study is based on [76, 78] and addresses collaborative linear regression with a nonconvex truncated loss term augmented by a 1-norm regularization term. The centralized problem is trivially separable such that:

$$F_i(\mathbf{z}) = \quad \min_{\mathbf{z}_i}. \ \frac{\zeta}{2M_i}\sum_{j=1}^{M_i}(\log\left(1+\frac{(y_{i,j}-\mathbf{z}_i^\intercal\mathbf{x}_{i,j})^2}{\zeta}\right)) + \xi_i||\mathbf{z}_i||_1 + \frac{\rho}{2}||\mathbf{z}_i-\mathbf{z}||_2^2 \tag{13}$$

The truncated loss term is used to make the regression more robust against outliers, while the regularization term penalizes non-sparsity in the regression coefficients. In the linear regression, $\mathbf{x}_{i,j} \in \mathbb{R}^d$ denotes the j^{th} sample's predictors of the i^{th} agent, and are normally distributed. $\mathbf{z}$ denotes the regression coefficients. $y_{i,j} \in \mathbb{R}$ denotes the j^{th} observed data sample of the i^{th} agent, and is synthesized according to $y_j = \mathbf{z}^{*\intercal}\mathbf{x}_{i,j} + v_{i,j}$ where $v_{i,j}$ is random Gaussian noise with standard deviation spanning a tenth of the number of dimensions. $\mathbf{z}^*$ denotes the ground truth model coefficients, sampled uniformly from $[-1,1]^d$, where d denotes the dimensionality of the problem, namely the number of model coefficients. We use 3,000 data samples in total, such that each of the N agents has $M_i = 3,000/N$ data samples. ζ and ξ_i, which control the level of truncation and regularization are set to 3 and 0.01 respectively.

3.2.4. Experiments

The subproblems do not contain any local constraints. The objectives in (13) are again tailored towards the implementation of ADMM according to (4). CUATRO is used in its standard, non-constrained form because the problem itself is not constrained. Starting from an initial solution of $\mathbf{z} = [0, ..., 0]^\intercal$, we investigate the following configurations of the case studies: We explore how the comparison of the different algorithms changes with increasing dimensionality ($d = 2, 10, 50$) at a fixed number of agents ($N = 2$). The second set of experiments explores what happens when the number of agents is increased ($N = 2, 4, 8$) when the dimensionality of the prediction coefficients is fixed ($d = 6$).

3.3. Facility allocation

3.3.1. Value chains

A key part of EWO is the design and operation of supply chains [65]. In an idealized setting, all stakeholders within a given value chain are willing to collaborate and share model information with a centralized coordinator. In practice however, antitrust and game-theoretical considerations might prevent stakeholders from fully collaborating. There is ample literature about 'Stackelberg Leader-Follower games' [82], where supply chain agents' take the first step' in deciding on the optimal location of their plants subject to other players reacting optimally with respect to their private objective. Yet, there is much value to be captured in moving away from these 'Nash equilibria', and approaching a coordinated optimum along the 'Pareto front'. To this end, a coordinator can optimize a (fairness-guided) game-theoretical operator that scalarizes and trades off the conflicting criteria of competing stakeholders [19, 20, 83, 2, 46].

This is especially relevant for the design of emerging supply chains with distinct characteristics such as biomass [61, 28] or (bio)pharmaceutical value chains [64, 63]. These 'social optima' are often obtained as the result of centralized optimization formulations, which can be decomposed for numerical tractability, or in our case to fit organizational considerations.

3.3.2. Case study

We consider a continuous facility location problem in two-dimensional continuous space, which belongs to the general class of Capacitated Multi-facility Weber Problems. The objective is to find the location, production, and connecting flows of all facilities that minimize a total cost. These 'shared' variables are few relative to the number of private variables and parameters, the latter including local cost parameters, technical upper and lower bounds, binary variables, and distances to/from facilities to name but a few. We use the same formulation as Lara et al. [47] with some key differences. In particular, we assume the presence of two suppliers and markets each. We fix the number of facilities to be built to either one or two, which still gives rise to a Generalized Disjunctive Problem (GDP). We also define the distances between agents and facilities using the 1-norm, rather than the 2-norm for computational efficiency. The GDP is finally reformulated into an MINLP using big-M constraints, implemented in Pyomo [35, 15], and solved using Gurobi [34]. The entire formulation can be found in Appendix E.

3.3.3. Decomposition

We are considering two different decompositions, motivated by two separate business scenarios. In the first scenario, the 2 types of nodes, suppliers and markets, each consisting of two nodes, are part of the same legal entity and are able to share model information. For each problem, we need to find the following shared variables: The two-dimensional location of the facilities ($2K$ variables) and their production (K variables). For our case, where the number of processing facilities is set to one or two ($K = 1, 2$), the problem contains either three or six shared variables and can be decomposed into two subproblems. The exact decomposition can be found in Appendix E

In our second scenario, we consider all of the four nodes (supplier and customer) to be their own separate legal entity with privacy considerations. As such, we need to decompose the problem into four. Unfortunately, the presence of complicating constraints - linking the total amount transported to and from a facility - prevent each agent from independently deciding on the amount that is transported between their node and the facilities. As such, the transport variables $f_{i,k}$ and $f_{k,j}$ become part of the set of shared variables. This would in principle add another $4K$ shared variables. However, these complicating constraints on the transport variables, only depending on the shared variables, can be used to reduce the number of degrees of freedom in the shared variables, such that these problems involve five or ten shared variables when one or two facilities are built respectively. The exact decompositions can again be found in Appendix E.

3.4. Multi-objective coordination

3.4.1. Case study

We consider the same synthetic problem as van de Berg et al. [73] where two stakeholders want to find the feedstock composition that optimizes a sum consisting of a cost and environmental impact term. Since both stakeholders are secretive about the intricacies of their proprietary optimization and simulation software, we can either use ADMM or data-driven coordination. In our considered case study, after fixing the feedstock composition variables $\mathbf{z}$, the problem becomes trivially decomposable. Agent A optimizes an economic blending problem, while Agent B optimizes the output of an environmental input simulation.

$$F_A(\mathbf{z}) = \min_{\mathbf{z}^I \in \mathbb{R}^{n_z}, \mathbf{y} \in \{0,1\}^{n_z}} . \quad \sum_i^{n_z} (z_i^I C_i + \frac{\rho}{2}(z_i^I - z_i)^2) \tag{14a}$$

$$\text{s.t.} \quad \sum_i^{n_z} z_i^I = 1, \quad \mathbf{l}_{qual} \leq \mathbf{z}^{I\intercal} A_{qual} \leq \mathbf{u}_{qual} \tag{14b}$$

$$\mathbf{0} \leq \mathbf{z}^I \leq \mathbf{y}, \quad \sum_i^{n_z} y_i \leq N_{int} \tag{14c}$$

$$F_B(\mathbf{z}) = \min_{\mathbf{z}^{II} \in \mathbb{R}^{n_z}} . \quad \sum_i^{n_z} (e_i {z_i^{II}}^{a_i} + \sum_{j \in J_i} x_i x_j + \frac{\rho}{2}(z_i^{II} - z_i)^2) \tag{15a}$$

Essentially, the economic blending problem minimizes the feedstock cost given component costs C_i, as well as upper and lower quality constraints $(\mathbf{u}_{qual}, \mathbf{l}_{qual})$ for quality matrix A_{qual}. Each dummy composition variable z_i^I is also subject to its binary variable y_i, which is active if the associated feedstock is non-zero. The number of active composition variables is constrained by N_{int}. As such, Agent A's formulation is a mixed-integer convex quadratic problem. Agent B's objective term is composed of a sum of linear or quadratic terms (each feedstock variable has its own power $a_i \in \{1, 2\}$). Additionally, each feedstock i has its sparse set of bilinear interactions J_i. The cost, quality, and environmental data are adopted from an animal feedstock database [1].

3.4.2. Black-box simulations in the subproblems

If the proximal term in the environmental subproblem is strongly penalized with a very high ρ, its optimal solution tends towards the solution corresponding to $\mathbf{z}^{II} = \mathbf{z}$. In this case, since the environmental subproblem does not involve any local constraints, the solution to this problem could theoretically be the result of a black-box simulation rather than optimization problem.

In practice, our data-driven alternatives could readily handle problems where the lower-level is obtained via simulation instead of an optimization, since progress only relies on objective evaluations. ADMM however relies on an update in the local copies of the shared variables. If the lower-level is obtained via a simulation, then $\mathbf{z}^{II}$ is never updated from the suggested $\mathbf{z}$. So at each iteration, $\mathbf{z}$ only approaches $\mathbf{z}^I$ rather than a compromise between $\mathbf{z}^I$ and $\mathbf{z}^{II}$, essentially omitting any environmental considerations. Hence, for ADMM, the subproblems need to be given by optimization. For convergence towards a collaborative optimum, $\mathbf{z}^{II}$ needs to slightly shift away from the suggested $\mathbf{z}$ towards the 'selfish' solution of the environmental problem optimized without economic considerations.

As such, in Section 4.5, we do not include the case where the subproblem is given by simulation, as this would bias the comparison between ADMM and the data-driven framework in favour of the data-driven coordination.

3.4.3. Experiments

We perform the mixed-integer, nonconvex coordination problem given by (14) and (15) on an increasing number of shared, feedstock variables ($n_z = 5, 10, 15, 20, 25$). Additionally, we perform the same experiment on a nonlinear but convex version of the previous problem. This is obtained by relaxing all mixed-integer constraints in (14c), and by omitting the bilinear terms in the environmental problem (15).

In the next section, we discuss the observed general convergence results, before discussing the particularities of each case study separately.

4. Results and Discussion

In the next section, we present general observed trends. These findings are then backed up in Figures 4 to 9 that present the convergence plots for each case study. Then, we investigate the relative algorithm performance based on characteristics discussed in the next section.

4.1. General observations

A high-level comparison between the performance of the data-driven coordination framework and ADMM based on problem considerations is summarised in Table 1.

Table 1: Performance comparison between ADMM and data-driven coordination based on mathematical problem and desired solution characteristics

Consideration	**ADMM**	**Data-driven coordination**
Number of shared variables $\mathbf{z}$	**Scales better** with higher dimensions	Shines in lower dimensions
Convergence speed	**Quick initial progress** but dependent on penalty parameter ρ	Can use exploitative DFO solver to **better fine-tune optimum**
Convergence guarantee	Guaranteed **for convex problems**.	**Depends on DFO solver** e.g. DIRECT-L for global convergence guarantee
Solution space topology in $\mathbf{z}$	Can get stuck at nonconvexities	Can use **explorative DFO** solver to **escape local minima**
Organizational and software	Requires numerical optimization for the subproblem solution	**More flexibility** in the subproblem solution (black-box simulation, heuristic evaluation, ...)

ADMM. ADMM manages to converge to at least a *local* minimizer if given enough function evaluations. If the proposed starting point is far from the optimum, initial progress with ADMM is generally fast. However, ADMM is found to be ill-suited for fine-tuning near-optimal solutions, which is in line with literature [13, 14].

Data-driven coordination. The performance of all data-driven coordination alternatives improves with respect to ADMM the more ill-behaved the solution space and the lower the dimensionality in the shared variables is. For the EWO case studies, we investigate a lower- and higher-dimensional configuration respectively, where for the lower-dimensional case, there is always at least one DFO variant that outperforms ADMM. Understanding the way these algorithms approach the exploration-exploitation trade-off is key to this observation. The coordination step in ADMM is purely exploitative. It cheaply extracts subgradient information from the subproblems to approach the coordinated optimum as quickly as possible. The relative performance of DFO algorithms against ADMM and explorative versus exploitative methods is determined by the mathematical properties of the case study.

DFO variants. Highly explorative frameworks like Bayesian Optimization perform well in lower-dimensional applications, where thorough exploration is more likely to be rewarded by faster convergence to the optimum. The exploration of some DFO algorithms can also be useful in escaping local optima. DIRECT-L, as a global optimization algorithm, usually makes slow progress as its function evaluations are used to thoroughly explore all partitions of the solution space, unless the optimum happens to be in the center of one of the initial partitions. CUATRO, with its decreasing trust region framework, encourages extensive initial exploration and later exploitation. ADMM_CUATRO usually displays superior performance over ALADIN_CUATRO. However, the choice between these two variants is in practice more motivated by organizational considerations concerning the sharing of either agent-level objective or surrogate information.

Py-BOBYQA. The previously discussed DFO algorithms tend to outperform ADMM only in lower-dimensional applications. Py-BOBYQA is the only DFO algorithm that has the potential to be competitive with ADMM in higher-dimensional applications given its similar focus on exploitation. As such, Py-BOBYQA tends to converge to the same local optima as ADMM. While ADMM displays faster convergence to low-accuracy regimes of the solution, Py-BOBYQA can find higher-accuracy solutions at the expense of more function evaluations.

Significance of the penalty parameter ρ. In theory, the value of the penalty parameter ρ should not influence the quality of the solutions found. In fact, the penalty term should approach zero at the optimum, since the local copies of the shared variables $\mathbf{z}_i$ should approach the suggested shared variables $\mathbf{z}$. In practice however, the choice of the penalty parameter ρ influences the accuracy and speed of convergence. If ρ is too weak, more deviation between $\mathbf{z}_i$ and $\mathbf{z}$ is allowed at the theoretical optimum, which can lead to more infeasibility in the returned solution. Some DFO variants find a better total evaluation than would theoretically be possible from the centralized solution, which explains why some algorithms do not display monotonically decreasing performance in the convergence plots. However, increasing ρ slows down convergence, since at each iteration $\mathbf{z}_i$ is bound closer to $\mathbf{z}$. As such, the conclusions on the relative convergence of the considered methods are influenced by ρ. There is ample literature on how ρ influences the convergence of ADMM [30]. There are multiple heuristics that can speed up ADMM, such as iteratively increasing ρ to allow for more exploration and faster convergence initially while encouraging fine-tuning in later iterations [77]. However, ρ is kept constant across our algorithm benchmarking since our analysis is based on comparing function evaluations. For the DFO methods, changing ρ would introduce 'noise' into the system, as the same sample would give different evaluations when sampled in later iterations with a higher ρ.

In Sections 4.2 to 4.5, we present best function evaluation versus number of function evaluation and convergence to the centralized solution optimum versus number of function evaluation plots for all algorithms on each considered case study configuration. ADMM and Py-BOBYQA do not involve any stochasticity. Since the underlying subproblems are also deterministic, we only need to include a single realization. While

CUATRO randomly samples function evaluations within the trust region, we also only include a single realization of both CUATRO versions at the default seed. For DIRECT-L and GPyOpt however, we involve 5 realizations each on all case studies. We plot their median evaluation with their min-max range shaded. While the best function evaluation plots illustrate low-accuracy convergence (especially relevant for a tight function evaluation budget), convergence plots are needed to compare high-accuracy convergence.

4.2. Motivating example

The motivating example encompasses all of the properties that call for data-driven coordination. The problem uses a penalty parameter ρ of 1,000, is one-dimensional with a starting point at $z = 4.5$, and an inflection point around $z = 3.5$, which might hinder convergence of purely exploitative methods. Figure 4c shows the solution space convergence of CUATRO and ADMM. While ADMM fails to pass the inflection point, all data-driven methods apart from ADMM converge to a near-optimal solution. The best function evaluation plot (Figure 4a) shows that the DFO variants converge to a low-accuracy solution in the following order from first to last: DIRECT-L, GPyOpt (Bayesian Optimization), Py-BOBYQA, ALADIN_CUATRO and ADMM_CUATRO. In this one-dimensional case study, initial exploration in DIRECT-L and GPyOpt encourages escaping the saddle point as quickly as possible. Figure 4b then shows that both CUATRO versions achieve a convergence of around 10^{-8} and 10^{-10} for the ADMM and ALADIN versions respectively, while the other DFO variants only achieve a median convergence up to 10^{-3} or 10^{-5}. This is due to the small trust region radius of the two CUATRO versions in later iterations favouring fine-tuning.

4.3. Collaborative model training

In this section, we first discuss the effect that dimensionality has on a nonconvex truncated linear regression problem when the number of agents is fixed to two. Then, we discuss the effect that an increase in the number of participating agents has when the dimensionality is fixed to six. All configurations use a penalty parameter ρ of 10.

4.3.1. Effect of the number of shared variables

Much of the algorithm convergence discussion in this section follows that of the motivating example because both problems present a nonconvex objective. The DFO variants perform particularly well on the lower two-dimensional case study (Figure 5a), taking up to 20 evaluations to converge compared to the 100 of ADMM. The convergence plot (Figure 5d) shows that Py-BOBYQA and both CUATRO variants converge to a high degree of accuracy in 20 evaluations, which takes DIRECT-L around 40 evaluations to reach. ADMM and GPyOpt only reach a (median) convergence of 10^{-5} and 10^{-7} respectively compared to 10^{-10} of the other three methods. Like in the motivating example, GPyOpt displays significant variance in its final convergence.

Figure 5b shows that when the dimensionality is increased to ten, the CUATRO variants lose their competitiveness with ADMM. Figure 5e illustrates how DIRECT-L displays a similar median convergence speed to ADMM. Py-BOBQA and GPyOpt make substantial progress in the first 20 evaluations. The best final convergence realization of GPyOpt matches that of ADMM (10^{-7}), while Py-BOBYQA is the best at fine-tuning solution accuracy (10^{-10}).

In the 50-dimensional case, we would expect ADMM to significantly outperform all DFO variants given its competitive advantage as a subgradient method, which becomes increasingly important in higher dimensions. However, in Figure 5c, we see that after around 100 evaluations, Py-BOBYQA is the only variant to find the optimum. This makes sense given that the starting point of the case study $\mathbf{z} = [0, \dots, 0]$ is already quite close to the optimal solution. This gives exploitative methods (ADMM and Py-BOBYQA) the upper hand. Finally, ADMM, despite making consistent progress, is slower at fine-tuning the optimum than Py-BOBYQA.

4.3.2. Effect of the number of agents

Starting with a dimensionality of six and two coordinating agents, we observe a similar convergence pattern in Figures 6a and 6d to the ten-dimensional case in the previous section (Figures 5b and 5e). ADMM, Py-BOBYQA, ALADIN_CUATRO and GPyOpt display a similar relative performance, while ADMM_CUATRO makes consistent progress and matches the convergence found for BO and Py-BOBYQA in 35 and 55 evaluations respectively.

Overall, we observe that with an increasing number of coordinating agents, the optimality gap increases between the final total function evaluation and the centralized optimum. There will always be small numerical differences between $\mathbf{z}_i$ and $\mathbf{z}$, which are penalized in the proximal terms $\frac{\rho}{2}||\mathbf{z}_i - \mathbf{z}||_2^2$. With an increasing number of agents, the relative importance of these proximal terms is strengthened, which becomes even more apparent when the optimal objective evaluation is close to the starting point as is the case for these problems.

The relative performance of the DFO algorithms with respect to each other and ADMM does however not seem to change with an increasing number of coordinating agents. ALADIN_CUATRO makes poor progress, DIRECT-L tracks the convergence of ADMM, while Py-BOBYQA and ADMM_CUATRO quickly find the best function evaluations. GPyOpt again makes quick initial progress but displays a lot of variance in best evaluation found. It is interesting to see that for this particular case, the convergence speeds of Py-BOBYQA, ADMM_CUATRO, and GPyOpt tend to remain similar even with an increasing number of agents.

4.4. Facility Location

In this section, we investigate the convergence and best function evaluation plots for the one- and two-facility location problem where the decisions of the two suppliers and two customers are operated by a single supplier and customer decision-maker each. We follow up this investigation with the case where all two suppliers and two customers present their own separate decision-making agent. All configurations use a penalty parameter ρ of 100,000.

4.4.1. 2 agents: supply and customer nodes belong to the same supply and demand decision-makers

Figure 7a shows that for the two-agent three-dimensional case, the exploitative methods ADMM and Py-BOBYQA display a similar convergence speed and are the only methods to converge quickly to the optimum. The two CUATRO versions converge to the same similar suboptimal point. The median DIRECT-L run manages to find the same optimum as ADMM and Py-BOBYQA after around 90 evaluations. GPyOpt only makes little progress. Figure 7b then shows that in the six-dimensional case, Py-BOBYQA and ADMM again outperform both CUATRO variants and Bayesian Optimization. As expected, the subgradient information of ADMM leads to faster convergence compared to Py-BOBYQA when the dimensionality is increased. Interestingly, DIRECT-L outperforms both exploitative methods, suggesting that the optimum is close to the center of one of the initial partitions used by DIRECT-L, which depends mostly on the user-given box bounds on the shared variables. The convergence versus number of function evaluations plots are omitted since they do not provide any additional information, as ADMM, Py-BOBYQA, and DIRECT-L converge to around the same accuracy.

4.4.2. 4 agents: each supplier and customer node as a separate decision-maker

Figures 8a and 8b show a significantly different relative algorithm performance for the four-agent case to that seen in the two-agent case of the last section. This is partially caused by the inclusion of additional shared variables in the form of facility to agent node transport links that need to be coordinated between all supplier and customer nodes. Additionally, these shared variables introduce ill-behaviour in the new solution space, which could be due to the way the shared variables are handled. In the multi-agent coordination case study, the number of shared variables is kept the same, but any infeasibilities in the shared variables are implicitly penalized through the proximal term. In this case study, complicating constraints are handled by reducing the degrees of freedom in the shared variables using material balances on the facility nodes. The choice on how best to handle complicating constraints is case study-specific. As a rule of thumb however, reducing the degrees of freedom using constraints favours convergence for data-driven methods, as ADMM struggles to deal with ill-behaved solution spaces. However, the DFO variants are not *guaranteed* to outperform ADMM even in lower dimensions when constraints are handled this way.

In fact, for the lower-dimensional case in Figure 8a, only Py-BOBYQA manages to navigate the solution space better than ADMM and is the only method to converge to the optimum. GPyOpt finds a similar optimum to ADMM, while the other DFO variants display worse performance to ADMM. Figure 8b shows a peculiar convergence pattern for the ten-dimensional case, apart for ALADIN_CUATRO which again makes no progress whatsoever. Py-BOBYQA displays similar convergence patterns to ADMM, and both find a slightly better optimum than ADMM_CUATRO. GPyOpt's final evaluations compete with ADMM. DIRECT-L is the only method to converge to the optimum in its median evaluation but displays significant variability in its convergence. Like in the higher-dimensional two-agent case, the solution space topology and input-bounds give rise to partitions whose center is close to the optimum. Convergence versus number of function evaluation plots are omitted again as they provide no additional information.

4.5. Multi-agent coordination

Figure 9 gives the best function evaluation and convergence plots for the 10- and 25-dimensional multi-agent coordination problems in their convex and nonconvex variant. All configurations use a penalty parameter ρ of 5,000.

Figures 9a and 9c show that all methods manage to find at least a low-accuracy optimum in the convex ten-dimensional variant. However, ADMM converges considerably faster than the fastest DFO variant (20 and 200 evaluations for ADMM and Py-BOBYQA to achieve the same accuracy respectively). It makes sense that both ADMM and Py-BOBYQA outperform more explorative methods for the considered configuration given their purely exploitative behaviour. This case also highlights the respective strengths of ADMM and Py-BOBYQA. ADMM is in general very fast to converge to a neighbourhood of a local optimizer. However, ADMM struggles to fine-tune the optimum. When the evaluation budget allows for it, Py-BOBYQA takes more evaluations to find this neighbourhood, but is more efficient at finding a better solution quality. Figure 9c shows that ADMM's final convergence is orders of magnitude worse than that of Py-BOBYQA (10^2 and 10^0 respectively). The discussion of the 25-dimensional convex configuration follows that of the 10-dimensional convex one. The relative performance of the algorithms is very similar, with the exception that ADMM's final convergence still displays a considerable optimality gap (Figure 9b). Py-BOBYQA is the only method to converge to a high-accuracy solution given its exploitative nature and its ability for fine-tuning.

The ten-dimensional nonconvex configuration presents conditions that favour data-driven approaches. The relative performance of the algorithms in Figure 9e is similar to that of its convex counterpart in Figure 9a with two notable exceptions: ADMM and Py-BOBYQA - both purely exploitative methods - converge to a *local* minimizer in 100 evaluations. ADMM is slightly quicker again, but Py-BOBYQA finds a slightly better solution. ADMM_CUATRO and the median run of DIRECT-L, due to their extensive initial exploration manage to escape a local minimizer and converge to a low-accuracy neighbourhood of the global optimum. The discussion of the higher-dimensional nonconvex case again follows that of its convex counterpart. ADMM and Py-BOBYQA are the only methods again to converge to at least a near-optimal solution. Py-BOBYQA finds a better solution quality, but takes significantly longer than ADMM. This emphasizes the importance of (sub)gradient information with increasing dimensionality.

5. Conclusion

Our proposed 'data-driven' framework is shown to be able to find the same solution as the equivalent centralized formulation for optimization-based coordination problems. Our approach differs from ADMM in that it uses derivative-free optimization (DFO) to find the shared variables that optimize lower-level subproblem evaluations. We consider CUATRO, Py-BOBYQA, DIRECT-L, and GPyOpt as DFO solvers and benchmark them against ADMM as a distributed optimization solver on a motivating example and three case studies with expensive subproblems. We examine the effect that dimensionality and solution topology in the shared variables have on the relative algorithm performance. We also discuss organizational considerations and how they inform the choice of coordinating algorithm: autonomy and flexibility, privacy, software, black-box subproblems, and organizational structure. We show that our approach outperforms ADMM when the number of shared variables between agents is few, and when the shared variable to shared objective evaluations call for exploration rather than exploitation. As opposed to distributed optimization, our method does not need the capacity for numerical optimization at the agent-level, since the subproblems can also be obtained as the result of a black-box objective simulation. We argue that our approach is especially relevant when the decomposition is limited by organizational rather than numerical considerations.

Our work is the first to benchmark several 'data-driven' algorithms against distributed optimization on multiple case studies relevant to enterprise-wide optimization. While the relative performance of DFO algorithms is in line with current literature, there are several avenues for future work: A practical implementation of a 'data-driven' approach would require a more thorough investigation into privacy (differential privacy, cryptography, ...) and into how much agent-level data could be inferred from optimal subproblem evaluations or variables. ADMM loses its convergence guarantees when the subproblems are ill-behaved. As such, our data-driven optimization approach might have a competitive advantage if the subproblem evaluations are not solved to global optimality, if evaluations are noisy and potentially inconsistent, or if they might change over time. This would be the case when objective evaluations are obtained by querying human decision-makers as opposed to optimization or simulation software. This would enable coordination between business units where some decision-makers still use expert-guided heuristics rather than numerical optimization.

Acknowledgements. D.v.d.B. gratefully acknowledges financial support from the Bansal bursary. D.v.d.B. would also like to thank all BASF S.E. collaborators, especially his co-supervisors Dr. Debora Morgenstern, Dr. Daniel Engel, and Dr. Inga-Lena Darkow, scientific exchange coordinator Dr. Christian Holtze, and Dr. Sangbum Lee for their invaluable support in conceptualizing this work.

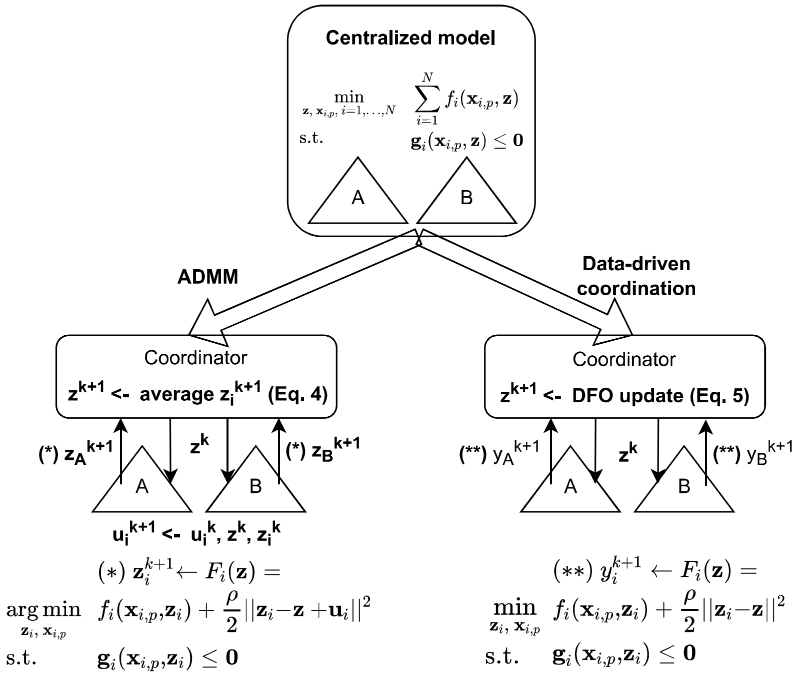


Figure 1: We can solve an equivalent centralized formulation either by ADMM or data-driven coordination ($\Downarrow$). In either case, a coordinator ($\square$) iteratively sends an updated proposed set of shared variables $\mathbf{z}^k$ ($\downarrow$) to all coordinating agents . The coordinating agents ($\triangle$) then optimize their private objective according to (*) or (**) for ADMM and data-driven coordination respectively. The subproblems differ in whether they include the scaled dual variables $\mathbf{u}_i^{k+1}$ - updated in the preceding step for ADMM - in their construction of the proximal term. Additionally, in ADMM, the subproblems return the optimal local copies of the proposed shared variables $\mathbf{z}_i^{k+1}$, whereas our framework returns the optimal subproblem objective evaluations y_i^{k+1} ($\uparrow$). In both frameworks, the penalty parameter ρ determines the extent to which the deviation between the suggested $\mathbf{z}^k$ and optimal set of private variables $\mathbf{z}_i^{k+1}$ is penalized. In the last step of the iteration, ADMM updates the proposed set of shared variables $\mathbf{z}^{k+1}$ by averaging its local copies $\mathbf{z}_i^{k+1}$, while our data-driven framework updates $\mathbf{z}^{k+1}$ using derivative-free optimization (DFO) and shared variable $\mathbf{z}^{k+1}$ to optimal evaluation y_i^{k+1} input-output data. In this case, we only show 2 coordinating agents, but this scheme can be generalized to any number of coordinating agents.

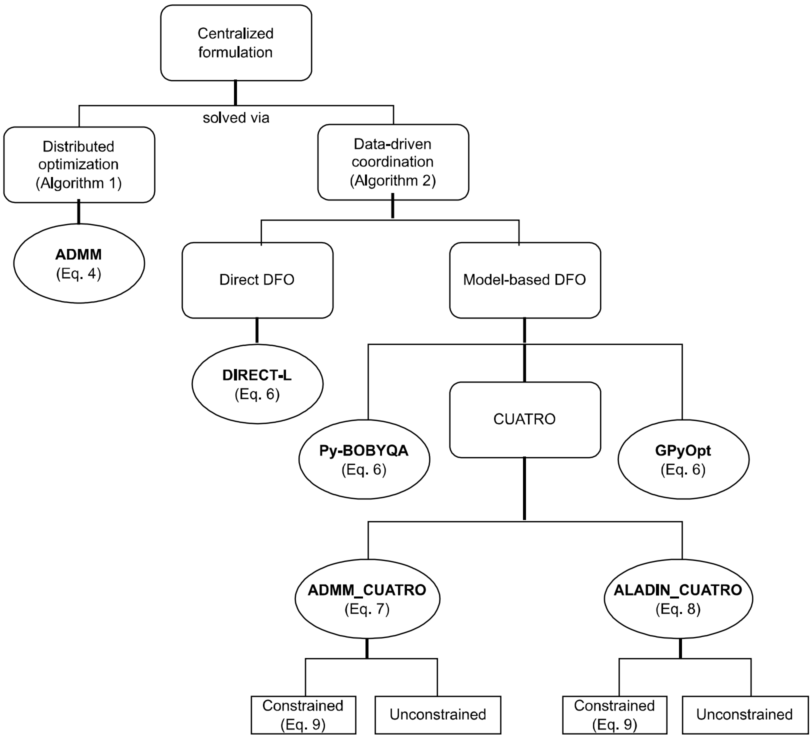


Figure 2: Classification of considered algorithms with associated problem formulations in parentheses: We can decide to solve our equivalent centralised formulation either via distributed optimization, namely ADMM, or via our proposed data-driven coordination framework. Within our framework, we can choose any derivative-free optimization (DFO) algorithm. We choose DIRECT-L as a direct DFO algorithm. Among model-based DFO algorithms, we consider Py-BOBYQA and CUATRO as quadratic trust region frameworks and GPyOpt as Bayesian Optimization. We also distinguish between ADMM_CUATRO and ALADIN_CUATRO in the way quadratic surrogates are formulated. Each CUATRO version also has the choice of explicit constraint satisfaction

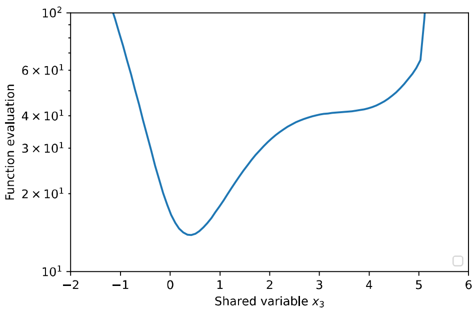
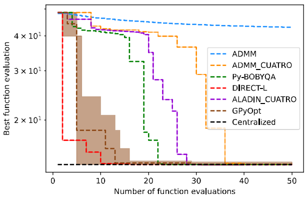
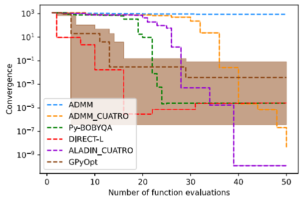


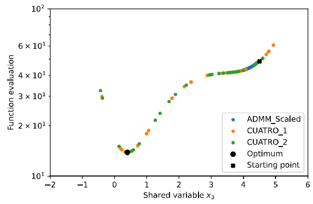
Figure 3: Upper-level objective of the motivating example as a function of the shared variable



(a) Best function evaluation versus number of function evaluations for all methods.



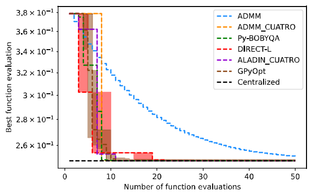
(b) Convergence versus number of function evaluations for all methods.



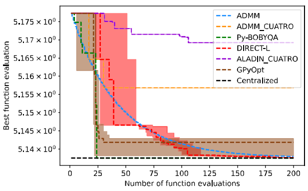
(c) Evaluation versus shared variable solution space for ADMM, ADMM_CUATRO and ALADIN_CUATRO

5

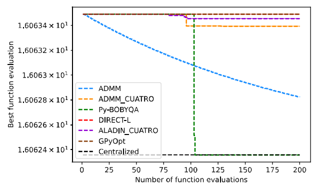
Figure 4: Convergence plots for the motivating example. For DIRECT-L and GPyOpt, the median best evaluation with shaded min-max range is given over 5 runs



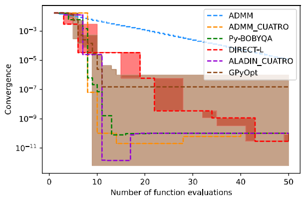
(a) Best function evaluation versus number of function evaluations using 2 agents and 2 shared variables.



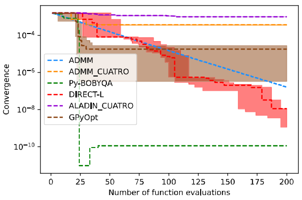
(b) Best function evaluation versus number of function evaluations using 2 agents and 10 shared variables.



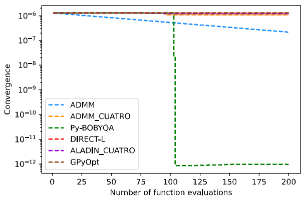
(c) Best function evaluation versus number of function evaluations using 2 agents and 50 shared variables.



(d) Convergence versus number of function evaluations using 2 agents and 2 shared variables.

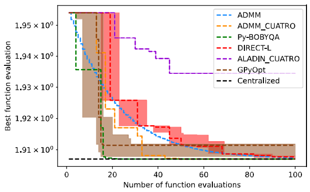


(e) Convergence versus number of function evaluations using 2 agents and 10 shared variables.

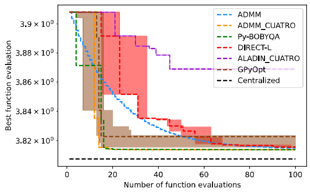


(f) Convergence versus number of function evaluations using 2 agents and 50 shared variables.

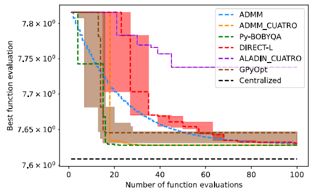
Figure 5: Effect of the number of shared variables on the truncated regression case study. For DIRECT-L and GPyOpt, the median best evaluation with shaded min-max range is given over 5 runs



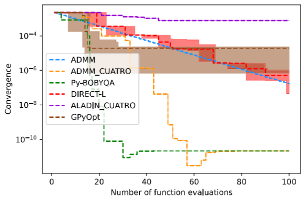
(a) Best function evaluation versus number of function evaluations using 2 agents and 6 shared variables.



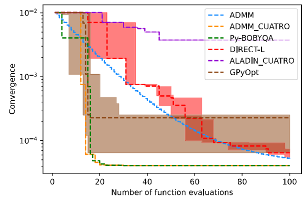
(b) Best function evaluation versus number of function evaluations using 4 agents and 6 shared variables.



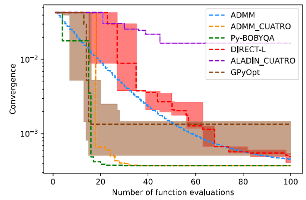
(c) Best function evaluation versus number of function evaluations using 8 agents and 6 shared variables.



(d) Convergence versus number of function evaluations using 2 agents and 6 shared variables.

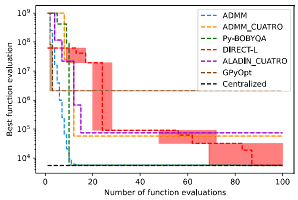


(e) Convergence versus number of function evaluations using 4 agents and 6 shared variables.

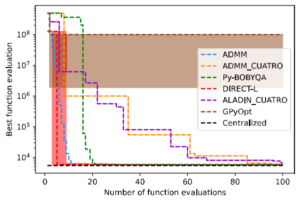


(f) Convergence versus number of function evaluations using 8 agents and 6 shared variables.

Figure 6: Effect of the number of agents on the truncated regression case study. For DIRECT-L and GPyOpt, the median best evaluation with shaded min-max range is given over 5 runs

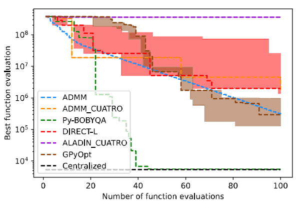


(a) Best function evaluation versus number of function evaluations using 2 agents and 3 shared variables.

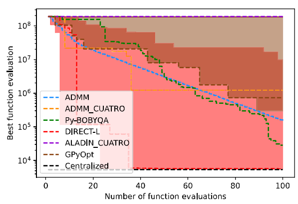


(b) Best function evaluation versus number of function evaluations using 2 agents and 6 shared variables.

Figure 7: Facility location convergence plots with two decision-makers. For DIRECT-L and GPyOpt, the median best evaluation with shaded min-max range is given over 5 runs

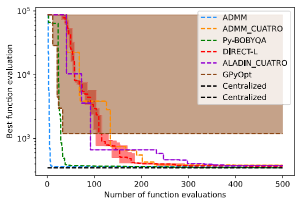


(a) Best function evaluation versus number of function evaluations using 4 agents and 5 shared variables.

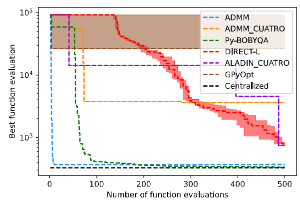


(b) Best function evaluation versus number of function evaluations using 4 agents and 10 shared variables.

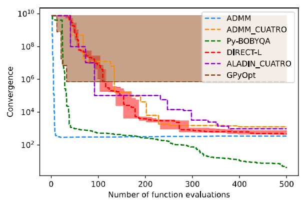
Figure 8: Facility location convergence plots with four decision-makers. For DIRECT-L and GPyOpt, the median best evaluation with shaded min-max range is given over 5 runs



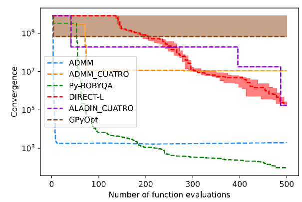
(a) Best function evaluation versus number of function evaluations on 10-dimensional convex version.



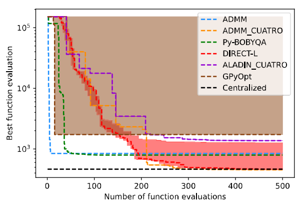
(b) Best function evaluation versus number of function evaluations on 25-dimensional convex version.



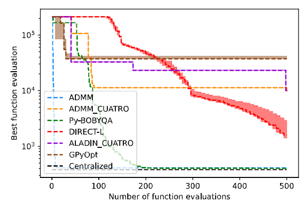
(c) Convergence versus number of function evaluations on 10-dimensional convex version.



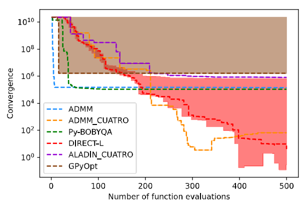
(d) Convergence versus number of function evaluations on 25-dimensional convex version.



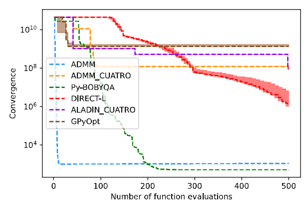
(e) Best function evaluation versus number of function evaluations on 10-dimensional nonconvex version.



(f) Best function evaluation versus number of function evaluations on 25-dimensional nonconvex version.



(g) Convergence versus number of function evaluations on 10-dimensional nonconvex version.



(h) Convergence versus number of function evaluations on 25-dimensional nonconvex version.

Figure 9: Multi-agent coordination convergence plots. For DIRECT-L and GPyOpt, the median best evaluation with shaded min-max range is given over 5 runs

References

[1] Tables of composition and nutritional values of feed materials inra cirad afz. URL https://www.feedtables.com/.

[2] Andrew Allman and Qi Zhang. Distributed fairness-guided optimization for coordinated demand response in multi-stakeholder process networks. *Computers & Chemical Engineering*, 161:107777, 2022. ISSN 0098-1354. doi: https://doi.org/10.1016/j.compchemeng.2022.107777. URL https://www.sciencedirect.com/science/article/pii/S0098135422001181.

[3] Satyajith Amaran, Nikolaos V Sahinidis, Sharda Bikram, and Scott J Bury. Simulation optimization: A review of algorithms and applications. 2017. doi: 10.1007/s10479-015-2019-x. URL https://link.springer.com/article/10.1007/s10479-015-2019-x.

[4] Teresa Arauz Pison, Paula Chanfreut, and J.M. Maestre. Cyber-security in networked and distributed model predictive control. *Annual Reviews in Control*, 11 2021. doi: 10.1016/j.arcontrol.2021.10.005.

[5] The GPyOpt authors. Gpyopt: A bayesian optimization framework in python. http://github.com/SheffieldML/GPyOpt, 2016.

[6] Albert S. Berahas, Liyuan Cao, Krzysztof Choromanski, and Katya Scheinberg. A Theoretical and Empirical Comparison of Gradient Approximations in Derivative-Free Optimization. *arXiv*, may 2019. URL http://arxiv.org/abs/1905.01332.

[7] Burcu Beykal, Styliani Avraamidou, Ioannis P.E. Pistikopoulos, Melis Onel, and Efstratios N. Pistikopoulos. DOMINO: Data-driven Optimization of bi-level Mixed-Integer NOnlinear Problems. *Journal of Global Optimization*, 78(1), 9 2020. ISSN 15732916. doi: 10.1007/s10898-020-00890-3.

[8] Burcu Beykal, Styliani Avraamidou, and Efstratios Pistikopoulos. *Bi-level Mixed-Integer Data-Driven Optimization of Integrated Planning and Scheduling Problems*, volume 50, pages 1707–1713. 01 2021. ISBN 9780323885065. doi: 10.1016/B978-0-323-88506-5.50265-5.

[9] Atharv Bhosekar and Marianthi Ierapetritou. Advances in surrogate based modeling, feasibility analysis, and optimization: A review, 2018. ISSN 00981354.

[10] Lorenz T. Biegler, Yi dong Lang, and Weijie Lin. Multi-scale optimization for process systems engineering. *Computers and Chemical Engineering*, 60:17–30, 1 2014. ISSN 00981354. doi: 10.1016/j.compchemeng.2013.07.009.

[11] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, mar 2004. doi: 10.1017/cbo9780511804441. URL https://www.cambridge.org/core/books/convex-optimization/17D2FAA54F641A2F62C7CCD01DFA97C4.

[12] Stephen Boyd, Lin Xiao, and Almir Mutapcic. Notes on Decomposition Methods. Technical report, 2003.

[13] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers, 2010. ISSN 19358237. URL http://www.nowpublishers.com/article/Details/MAL-016.

[14] Alessandro Buccini, Pietro Dell'Acqua, and Marco Donatelli. A general framework for admm acceleration. *Numerical Algorithms*, 85, 11 2020. doi: 10.1007/s11075-019-00839-y.

[15] Michael L. Bynum, Gabriel A. Hackebeil, William E. Hart, Carl D. Laird, Bethany L. Nicholson, John D. Siirola, Jean-Paul Watson, and David L. Woodruff. *Pyomo–optimization modeling in python*, volume 67. Springer Science & Business Media, third edition, 2021.

[16] Coralia Cartis, Lindon Roberts, and Oliver Sheridan-Methven. Escaping local minima with derivative-free methods: a numerical investigation. Technical report, 2019.

[17] Coralia Cartis, Lindon Roberts, and Oliver Sheridan-Methven. Escaping local minima with local derivative-free methods: a numerical investigation. *https://doi.org/10.1080/02331934.2021.1883015*, 2021. ISSN 10294945. doi: 10.1080/02331934.2021.1883015. URL https://www.tandfonline.com/doi/abs/10.1080/02331934.2021.1883015.

[18] E. A.del Rio Chanona, P. Petsagkourakis, E. Bradford, J. E.Alves Graciano, and B. Chachuat. Real-time optimization meets Bayesian optimization and derivative-free optimization: A tale of modifier adaptation. *Computers and Chemical Engineering*, 147:107249, 4 2021. ISSN 00981354. doi: 10.1016/j.compchemeng.2021.107249.

[19] Cheng-Liang Chen and Wen-Cheng Lee. Multi-objective optimization of multi-echelon supply chain networks with uncertain product demands and prices. *computers & Chemical Engineering*, 28:1131–1144, 06 2004. doi: 10.1016/j.compchemeng.2003.09.014.

[20] Yunfei Chu and Fengqi You. Model-based integration of control and operations: Overview, challenges, advances, and opportunities. *Computers and Chemical Engineering*, 83:2–20, 2015. ISSN 00981354. doi: 10.1016/j.compchemeng.2015.04.011. URL http://dx.doi.org/10.1016/j.compchemeng.2015.04.011.

[21] Muge Erdirik Dogan and Ignacio E. Grossmann. A decomposition method for the simultaneous planning and scheduling of single-stage continuous multiproduct plants. *Industrial & Engineering Chemistry Research*, 45:299–315, 2006.

[22] Roberto Dominguez and Salvatore Cannella. Insights on multi-agent systems applications for supply chain management. *Sustainability (Switzerland)*, 12(5):1–13, mar 2020. ISSN 20711050. doi: 10.3390/su12051935. URL www.mdpi.com/journal/sustainability.

[23] Alexander Engelmann, Yuning Jiang, Boris Houska, and Timm Faulwasser. Decomposition of Non-convex Optimization via Bi-Level Distributed ALADIN. *IEEE Transactions on Control of Network Systems*, 7(4):1848–1858, dec 2020. doi: 10.1109/TCNS.2020.3005079.

[24] Kobi C. Felton, Jan G. Rittig, and Alexei A. Lapkin. Summit: Benchmarking Machine Learning Methods for Reaction Optimisation. *Chemistry–Methods*, 1(2):116–122, 2 2021. ISSN 2628-9725. doi: 10.1002/cmtd.202000051. URL https://onlinelibrary.wiley.com/doi/10.1002/cmtd.202000051.

[25] Robert Fourer, Jun Ma, and Kipp Martin. Optimization services: A framework for distributed optimization. *Operations Research*, 58(6):1624–1636, 2010. ISSN 0030364X, 15265463. URL http://www.jstor.org/stable/40984032.

[26] Drew Fudenberg and Jean Tirole. *Game Theory*. MIT Press, Cambridge, MA, 1991. Translated into Chinesse by Renin University Press, Bejing: China.

[27] J. M. Gablonsky and C. T. Kelley. A Locally-Biased form of the DIRECT Algorithm. *Journal of Global Optimization*, 21(1):27–37, 2001. ISSN 09255001. doi: 10.1023/A:1017930332101. URL https://link.springer.com/article/10.1023/A:1017930332101.

[28] Daniel J. Garcia and Fengqi You. The water-energy-food nexus and process systems engineering: A new focus. *computers & Chemical Engineering*, 91:49–67, 2016. ISSN 0098-1354. doi: https://doi.org/10.1016/j.compchemeng.2016.03.003. URL https://www.sciencedirect.com/science/article/pii/S0098135416300552. 12th International Symposium on Process Systems Engineering & 25th European Symposium of Computer Aided Process Engineering (PSE-2015/ESCAPE-25), 31 May - 4 June 2015, Copenhagen, Denmark.

[29] Roman Garnett. *Bayesian Optimization*. Cambridge University Press, 2022. in preparation.

[30] Euhanna Ghadimi, André Teixeira, Iman Shames, and Mikael Johansson. Optimal parameter selection for the alternating direction method of multipliers (admm): Quadratic problems. *IEEE Transactions on Automatic Control*, 09 2014. doi: 10.1109/TAC.2014.2354892.

[31] Chrysanthos E. Gounaris and Ignacio E. Grossmann. A preface to the special issue on enterprise-wide optimization, dec 2019. ISSN 15732924. URL https://doi.org/10.1007/s11081-019-09468-9.

[32] Ignacio Grossmann. Enterprise-wide optimization: A new frontier in process systems engineering. *AIChE Journal*, 51(7):1846–1857, jul 2005. ISSN 0001-1541. doi: 10.1002/aic.10617. URL http://doi.wiley.com/10.1002/aic.10617.

[33] Ignacio E. Grossmann. Advances in mathematical programming models for enterprise-wide optimization. *Computers & Chemical Engineering*, 47:2–18, dec 2012. ISSN 0098-1354. doi: 10.1016/J.COMPCHEMENG.2012.06.038. URL https://www.sciencedirect.com/science/article/abs/pii/S0098135412002220.

[34] Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2022. URL https://www.gurobi.com.

[35] William E Hart, Jean-Paul Watson, and David L Woodruff. Pyomo: modeling and solving mathematical programs in python. *Mathematical Programming Computation*, 3(3):219–260, 2011.

[36] Manuel Herrera, Marco Pérez-Hernández, Ajith Kumar Parlikad, and Joaquín Izquierdo. Multi-Agent Systems and Complex Networks: Review and Applications in Systems Engineering. *Processes*, 8(3):312, mar 2020. ISSN 2227-9717. doi: 10.3390/pr8030312. URL https://www.mdpi.com/2227-9717/8/3/312.

[37] Boris Houska, Janick Frasch, and Moritz Diehl. AN AUGMENTED LAGRANGIAN BASED ALGORITHM FOR DISTRIBUTED NONCONVEX OPTIMIZATION *. 26(2):1101–1127. doi: 10.1137/140975991. URL http://www.siam.org/journals/ojsa.php.

[38] Boris Houska, Janick Frasch, and Moritz Diehl. An augmented Lagrangian based algorithm for distributed nonconvex optimization. *SIAM Journal on Optimization*, 2016. ISSN 10526234. doi: 10.1137/140975991.

[39] Ramaswamy R. Iyer and Ignacio E. Grossmann. A bilevel decomposition algorithm for long-range planning of process networks. *Industrial & Engineering Chemistry Research*, 37:474–481, 1998.

[40] Jennifer R. Jackson and Ignacio E. Grossmann. Temporal decomposition scheme for nonlinear multisite production planning and distribution models. *Industrial & Engineering Chemistry Research*, 42(13): 3045–3055, 2003. doi: 10.1021/ie030070p. URL https://doi.org/10.1021/ie030070p.

[41] Steven G. Johnson. *The NLopt nonlinear-optimization package*, 2020. URL http://github.com/stevengj/nlopt.

[42] D. R. Jones, C. D. Perttunen, and B. E. Stuckman. Lipschitzian optimization without the Lipschitz constant. *Journal of Optimization Theory and Applications*, 79(1):157–181, 10 1993. ISSN 00223239. doi: 10.1007/BF00941892. URL https://link.springer.com/article/10.1007/BF00941892.

[43] Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, Rafael G. L. D'Oliveira, Hubert Eichner, Salim El Rouayheb, David Evans, Josh Gardner, Zachary Garrett, Adrià Gascón, Badih Ghazi, Phillip B. Gibbons, Marco Gruteser, Zaid Harchaoui, Chaoyang He, Lie He, Zhouyuan Huo, Ben Hutchinson, Justin Hsu, Martin Jaggi, Tara Javidi, Gauri Joshi, Mikhail Khodak, Jakub Konecný, Aleksandra Korolova, Farinaz Koushanfar, Sanmi Koyejo, Tancrède Lepoint, Yang Liu, Prateek Mittal, Mehryar Mohri, Richard Nock, Ayfer Özgür, Rasmus Pagh, Hang Qi, Daniel Ramage, Ramesh Raskar, Mariana Raykova, Dawn Song, Weikang Song, Sebastian U. Stich, Ziteng Sun, Ananda Theertha Suresh, Florian Tramèr, Praneeth Vepakomma, Jianyu Wang, Li Xiong, Zheng Xu, Qiang Yang, Felix X. Yu, Han Yu, and Sen Zhao. Advances and open problems in federated learning. *Foundations and Trends in Machine Learning*, 14(1–2):1–210, 2021. ISSN 1935-8237. doi: 10.1561/2200000083. URL http://dx.doi.org/10.1561/2200000083.

[44] Sun Hye Kim, · Fani Boukouvala, S H Kim, and F Boukouvala. Machine learning-based surrogate modeling for data-driven optimization: a comparison of subset selection for regression techniques. *Optimization Letters*, 14:989–1010, 2020. doi: 10.1007/s11590-019-01428-7. URL https://doi.org/10.1007/s11590-019-01428-7.

[45] Dimitris Kouzoupis, Rien Quirynen, Boris Houska, and Moritz Diehl. A block based ALADIN scheme for highly parallelizable direct Optimal Control. In *Proceedings of the American Control Conference*, 2016. ISBN 9781467386821. doi: 10.1109/ACC.2016.7525066.

[46] D. Krishnamoorthy. A distributed feedback-based online process optimization framework for optimal resource sharing. *Journal of Process Control*, 97:72–83, 2021.

[47] Cristiana L. Lara, Francisco Trespalacios, and Ignacio E. Grossmann. Global optimization algorithm for capacitated multi-facility continuous location-allocation problems. *undefined*, 71(4):871–889, 8 2018. ISSN 15732916. doi: 10.1007/S10898-018-0621-6.

[48] Jeffrey Larson, Matt Menickelly, and Stefan M. Wild. Derivative-free optimization methods. *Acta Numerica*, 2019. ISSN 14740508. doi: 10.1017/S0962492919000060.

[49] Zhongguo Li, Zhen Dong, Zhongchao Liang, and Zhengtao Ding. Surrogate-based distributed optimisation for expensive black-box functions. *Automatica*, 125:109407, 2021. ISSN 0005-1098. doi: https://doi.org/10.1016/j.automatica.2020.109407. URL https://www.sciencedirect.com/science/article/pii/S0005109820306099.

[50] J.M. Maestre, D. Pena, and Eduardo Camacho. Distributed model predictive control based on a cooperative game. *Optimal Control Applications and Methods*, 32:153 – 176, 03 2011. doi: 10.1002/oca.940.

[51] George Makrigiorgos, Angelo Domenico Bonzanini, Victor Miller, and Ali Mesbah. Performance-oriented model learning for control via multi-objective bayesian optimization. *Computers & Chemical Engineering*, page 107770, 03 2022. doi: 10.1016/j.compchemeng.2022.107770.

[52] H. B. McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *AISTATS*, 2017.

[53] Robert Nishihara, Laurent Lessard, Benjamin Recht, Andrew Packard, and Michael I. Jordan. A general analysis of the convergence of admm. *ArXiv*, abs/1502.02009, 2015.

[54] F. Oliveira, Vijay Gupta, Silvio Hamacher, and Ignacio E. Grossmann. A lagrangean decomposition approach for oil supply chain investment planning under uncertainty with risk considerations. *Comput. Chem. Eng.*, 50:184–195, 2013.

[55] Simon Olofsson, Eduardo S Schultz, Adel Mhamdi, · Alexander Mitsos, · Marc, Peter Deisenroth, and Ruth Misener. Design of Dynamic Experiments for Black-Box Model Discrimination. Technical report, 2021.

[56] Daniel P. Palomar and Mung Chiang. A tutorial on decomposition methods for network utility maximization. *IEEE Journal on Selected Areas in Communications*, 24(8):1439–1451, aug 2006. ISSN 07338716. doi: 10.1109/JSAC.2006.879350.

[57] Joel Paulson, George Makrigiorgos, and Ali Mesbah. Adversarially robust bayesian optimization for efficient auto-tuning of generic control structures under uncertainty. *AIChE Journal*, 01 2022. doi: 10.1002/aic.17591.

[58] M.S. Pishvaee, J. Razmi, and S.A. Torabi. An accelerated benders decomposition algorithm for sustainable supply chain network design under uncertainty: A case study of medical needle and syringe supply chain. 2014. doi: 10.1016/j.tre.2014.04.001.

[59] Jose Rodriguez, Bethany Nicholson, Carl Laird, and Victor Zavala. Benchmarking admm in nonconvex nlps. *Computers & Chemical Engineering*, 119, 08 2018. doi: 10.1016/j.compchemeng.2018.08.036.

[60] Nuria Rodríguez-Barroso, Goran Stipcich, Daniel Jiménez-López, José Antonio Ruiz-Millán, Eugenio Martínez-Cámara, Gerardo González-Seco, M. Victoria Luzón, Miguel Angel Veganzones, and Francisco Herrera. Federated learning and differential privacy: Software tools analysis, the sherpa.ai fl framework and methodological guidelines for preserving data privacy. *Information Fusion*, 64:270–292, 2020. ISSN 1566-2535. doi: https://doi.org/10.1016/j.inffus.2020.07.009. URL https://www.sciencedirect.com/science/article/pii/S1566253520303213.

[61] Apoorva M. Sampat, Edgar Martin, Mariano Martin, and Victor M. Zavala. Optimization formulations for multi-product supply chain networks. *computers & Chemical Engineering*, 104:296–

310, 2017. ISSN 0098-1354. doi: https://doi.org/10.1016/j.compchemeng.2017.04.021. URL https://www.sciencedirect.com/science/article/pii/S0098135417301849.

[62] Tjendera Santoso, Shabbir Ahmed, Marc Goetschalckx, and Alexander Shapiro. *A stochastic programming approach for supply chain network design under uncertainty*. 07 2003. doi: 10.18452/8297.

[63] Miriam Sarkis, Andrea Bernardi, Nilay Shah, and Maria M. Papathanasiou. Emerging challenges and opportunities in pharmaceutical manufacturing and distribution. *Processes*, 9(3), 2021. ISSN 2227-9717. doi: 10.3390/pr9030457. URL https://www.mdpi.com/2227-9717/9/3/457.

[64] Nilay Shah. Pharmaceutical supply chains: key issues and strategies for optimisation. *computers & Chemical Engineering*, 28(6):929–941, 2004. ISSN 0098-1354. doi: https://doi.org/10.1016/j.compchemeng.2003.09.022. URL https://www.sciencedirect.com/science/article/pii/S0098135403002333. FOCAPO 2003 Special issue.

[65] Nilay Shah. Process industry supply chains: Advances and challenges. *computers & Chemical Engineering*, 29(6):1225–1235, 2005. ISSN 0098-1354. doi: https://doi.org/10.1016/j.compchemeng.2005.02.023. URL https://www.sciencedirect.com/science/article/pii/S009813540500013X. Selected Papers Presented at the 14th European Symposium on Computer Aided Process Engineering.

[66] Benjamin J Shields, Jason Stevens, Jun Li, Marvin Parasram, Farhan Damani, Jesus I Martinez Alvarado, Jacob M Janey, Ryan P Adams, and Abigail G Doyle. Bayesian reaction optimization as a tool for chemical synthesis. *Nature*, 590:89, 2021. doi: 10.1038/s41586-021-03213-y. URL https://doi.org/10.1038/s41586-021-03213-y.

[67] Sungho Shin, Philip Hart, Thomas Jahns, and Victor M. Zavala. A hierarchical optimization architecture for large-scale power networks. *IEEE Transactions on Control of Network Systems*, 6(3):1004–1014, 2019. doi: 10.1109/TCNS.2019.2906917.

[68] Rui Sousa, Songsong Liu, Lazaros Papageorgiou, and Nilay Shah. Global supply chain planning for pharmaceuticals. *Chemical Engineering Research & Design - CHEM ENG RES DES*, 89:2396–2409, 11 2011. doi: 10.1016/j.cherd.2011.04.005.

[69] Wentao Tang and Prodromos Daoutidis. Distributed control and optimization of process system networks: A review and perspective. *Chinese Journal of Chemical Engineering*, 27(7):1461–1473, jul

2019. ISSN 1004-9541. doi: 10.1016/J.CJCHE.2018.08.027. URL https://www.sciencedirect.com/science/article/abs/pii/S100495411830853X.

[70] Sebastian Terrazas-Moreno and Ignacio Grossmann. A multiscale decomposition method for the optimal planning and scheduling of multi-site continuous multiproduct plants. *Chemical Engineering Science - CHEM ENG SCI*, 66:4307–4318, 10 2011. doi: 10.1016/j.ces.2011.03.017.

[71] Konstantinos I. Tsianos, Sean Lawlor, and Michael G. Rabbat. Consensus-based distributed optimization: Practical issues and applications in large-scale machine learning. In *2012 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1543–1550, 2012. doi: 10.1109/Allerton.2012.6483403.

[72] Halit Uster, Gopalakrishnan Easwaran, Elif Akcali, and Sila Çetinkaya. Benders decomposition with alternative multiple cuts for a multi-product closed-loop supply chain network design model. *Naval Research Logistics (NRL)*, 54:890 – 907, 12 2007. doi: 10.1002/nav.20262.

[73] Damien van de Berg, Panagiotis Petsagkourakis, Nilay Shah, and Ehecatl Antonio del Rio-Chanona. Data-driven distributed optimization for systems consisting of expensive black-box subproblems. In *14th International Symposium on Process Systems Engineering (accepted)*, 2022.

[74] Damien van de Berg, Thomas Savage, Panagiotis Petsagkourakis, Dongda Zhang, Nilay Shah, and Ehecatl Antonio del Rio-Chanona. Data-driven optimization for process systems engineering applications. *Chemical Engineering Science*, 248:117135, 2 2022. ISSN 0009-2509. doi: 10.1016/J.CES.2021.117135.

[75] Andreas Wächter and Lorenz T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57, may 2006. ISSN 00255610. doi: 10.1007/s10107-004-0559-y. URL https://link.springer.com/article/10.1007/s10107-004-0559-y.

[76] Zhiguo Wang, Jiawei Zhang, Tsung-Hui Chang, Jian Li, and Zhi-Quan Luo. Distributed Stochastic Consensus Optimization with Momentum for Nonconvex Nonsmooth Problems. 2020.

[77] Yi Xu, Mingrui Liu, Qihang Lin, and Tianbao Yang. Admm without a fixed penalty parameter: Faster convergence with new adaptive penalization. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 1267–1277, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

[78] Yi Xu, Shenghuo Zhu, Sen Yang, Chi Zhang, Rong Jin, and Tianbao Yang. Learning with Non-Convex Truncated Losses by SGD. *35th Conference on Uncertainty in Artificial Intelligence, UAI 2019*, 5 2018. URL https://arxiv.org/abs/1805.07880v1.

[79] Qiang Yang, Yang Liu, Yong Cheng, Yan Kang, Tianjian Chen, and Han Yu. 2019.

[80] Tao Yang, Xinlei Yi, Junfeng Wu, Ye Yuan, Di Wu, Ziyang Meng, Yiguang Hong, Hong Wang, Zongli Lin, and Karl H. Johansson. A survey of distributed optimization. *Annual Reviews in Control*, 47: 278–305, 2019. ISSN 1367-5788. doi: https://doi.org/10.1016/j.arcontrol.2019.05.006. URL https://www.sciencedirect.com/science/article/pii/S1367578819300082.

[81] Andrew C. Yao. Protocols for secure computations. In *23rd Annual Symposium on Foundations of Computer Science (sfcs 1982)*, pages 160–164, 1982. doi: 10.1109/SFCS.1982.38.

[82] Dajun Yue and Fengqi You. Stackelberg-game-based modeling and optimization for supply chain design and operations: A mixed integer bilevel programming framework. *Computers & Chemical Engineering*, 102:81–95, jul 2017. ISSN 0098-1354. doi: 10.1016/J.COMPCHEMENG.2016.07.026. URL https://www.sciencedirect.com/science/article/pii/S0098135416302460?via%3Dihub.

[83] V.M. Zavala. Chapter seven - managing conflicts among decision-makers in multiobjective design and operations. In Gerardo Ruiz-Mercado and Heriberto Cabezas, editors, *Sustainability in the Design, Synthesis and Analysis of Chemical Engineering Processes*, pages 169–180. Butterworth-Heinemann, Oxford, 2016. ISBN 978-0-12-802032-6. doi: https://doi.org/10.1016/B978-0-12-802032-6.00007-4. URL https://www.sciencedirect.com/science/article/pii/B9780128020326000074.

[84] Fei Zhao, Ignacio E. Grossmann, Salvador García-Muñoz, and Stephen D. Stamatis. Flexibility index of black-box models with parameter uncertainty through derivative-free optimization. *AIChE Journal*, 2021. ISSN 15475905. doi: 10.1002/aic.17189.