Manuel Baumann, Sara Grundel, Philipp Sauerteig*, and Karl Worthmann

Surrogate Models in Bidirectional Optimization of Coupled Microgrids

Ersatzmodelle in bidirektionaler Optimierung gekoppelter Microgrids

Received 05-07-2019; revised 04-09-2019; accepted 06-09-2019

Abstract: The energy transition entails a rapid uptake of renewable energy sources. Besides physical changes within the grid infrastructure, energy storage devices and their smart operation are key measures to master the resulting challenges like, e.g., a highly fluctuating power generation. For the latter, optimization based control has demonstrated its potential on a microgrid level. However, if a network of coupled microgrids is considered, iterative optimization schemes including several communication rounds are typically used. Here, we propose to replace the optimization on the microgrid level by using surrogate models either derived from radial basis functions or neural networks to avoid this iterative procedure. We prove well-posedness of our approach and demonstrate its efficiency by numerical simulations based on real data provided by an Australian grid operator.

Keywords: Smart Grids, Model Predictive Control, Distributed Optimization, Surrogate Models, Bidirectional Optimization, Neural Networks, Radial Basis Functions

Zusammenfassung: Die Energiewende bringt einen raschen Zuwachs eneuerbarer Energiequellen mit sich. Neben den physikalischen Veränderungen der Netzinfrastruktur spielen Energiespeichereinheiten und deren intelligente Nutzung eine entscheidende Rolle, um die sich ergebenden Probleme wie z.B. die stark schwankende Energieerzeugung zu bewältigen. In Bezug auf Letztere haben optimierungsbasierte Steuerungstechniken ihr Potential auf Microgrid-Ebene unter Beweis gestellt. Betrachtet man jedoch ein Netzwerk gekoppelter Microgrids, werden üblicherweise iterative Optimierungsansätze gewählt, welche mit mehreren Kommunikationsrunden einhergehen. Um derartigen Kommunikationsschleifen vorzubeugen, schlagen wir vor, den Optimierungsschritt auf Microgrid-Ebene durch den Einsatz geeigneter Ersatzmodelle zu vermeiden. Den hier verwendeten Ersatzmodellen liegen zum einen radiale Basisfunktionen und zum anderen neuronale Netze zugrunde. Wir zeigen, dass unser Ansatz wohlgestellt ist und demonstrieren die Effizienz anhand numerischer Simulationen basierend auf realen Daten eines australischen Verteilnetzbetreibers.

Schlagwörter: Smart Grids, Modellprädiktive Regelung, Verteilte Optimierung, Ersatzmodelle, Bidirektionale Optimierung, Neuronale Netze, Radiale Basisfunktionen

1 Introduction

The share of renewable energy sources rapidly increases; also due to more and more installed devices like e.g., solar panels at household-level. Hence, households become *prosumers*, i.e., power is not only consumed but also produced. Therefore, energy generation and distribution takes place in a distributed way. In particular, energy can be transmitted bidirectionally between the grid and the prosumers, which results in a paradigm shift in the grid organization. In addition, prosumers may also possess some kind of energy storage device in order to manipulate their power demand profiles by either charging or discharging. From the grid operator's perspective it might be beneficial that charging decisions are not made based on local information only. Instead taking into account information on the entire grid may improve the system-wide operation, e.g., to flatten the overall power demand within the grid in order to facilitate the power supply. [1]. In order to achieve this goal, communication is needed. In the future, each household shall be equipped with a smart meter which yields so-called *smart homes*. Smart meters collect data and communicate with the grid operator automatically.

A straight-forward way to optimally operate the overall system is to formulate one large-scale optimization problem and to solve it in a centralized way, see, e.g. [2]. This approach, however, is hard to realize in practice. One

Manuel Baumann, Sara Grundel, Max Planck Institute for Dynamics of Complex Technical Systems,
{baumann,grundel}@mpi-magdeburg.mpg.de
***Corresponding author: Philipp Sauerteig,** Technische Universität Ilmenau, philipp.sauerteig@tu-ilmenau.de
Karl Worthmann, Technische Universität Ilmenau,
karl.worthmann@tu-ilmenau.de

of the disadvantages is that some central node needs the complete information about the grid, which is, e.g. due to data privacy, not desirable. Alternatives are decentralized or distributed optimization algorithms. In [3] the authors propose a decentralized approach to steer energy storage systems in order to avoid over-capacity of pole transformers while maintaining a high charging amount of energy storage systems in low-voltage distribution systems. The other option mentioned above are distributed optimization methods such as distributed dual ascent [4], Alternating Direction Method of Multipliers (ADMM) [5] or Augmented Lagrangian based Alternating Direction Inexact Newton (ALADIN) [6]. These algorithms use a star-shaped communication topology, i.e. each smart home communicates only with the grid operator and does not share any information with its neighbours. Nevertheless, in every iteration each household has to transmit specific (personal) data to the grid operator, see also [7] and [8] for an application of ADMM and ALADIN to electrical networked systems, respectively. In order to exploit the potential of these algorithms they are typically embedded within a Model Predictive Control (MPC) framework. MPC is a state-of-the-art technique to tackle optimal control problems by solving finite-dimensional optimization problems successively, see e.g. [9] for an introduction to MPC and [10, 11] for MPC approaches in electrical networks.

An alternate option to steer the power demand of local agents besides battery control is to schedule so-called controllable loads. Controllable loads can be shifted in time to avoid bottlenecks in the energy supply, see e.g. [12, 13]. There is also a large potential in the context of stochastic optimization of smart grids. For weather forecasting methods we refer to [14]. How to integrate electrical vehicles into the electricity network under uncertainties is described in [15].

Considering the power networks described so far, it is assumed that exchange of energy within the grid is possible at any time and does not cause any losses or additional costs, which might (approximately) hold for domestic nets, e.g. a town. In this paper, we refer to these grids as microgrids (MGs). In [16, 17], the concept of coupled MGs is used to tackle large-scale problems incorporating several MGs. In the latter, the authors show that even if each single MG is optimally operated, there is still room for improvement if energy can be exchanged among MGs. Therefore, a second optimization problem is solved on a higher grid level in order to optimally exchange energy resulting in a bilevel optimization problem [18].

In [19], the authors propose to replace the distributed optimization routine on the lower grid level by a surrogate model in order to speed-up the calculation and further reduce communication effort. Here, Radial Basis Functions (RBFs) [20] are used to approximate the input-output behaviour of ADMM within the framework of coupled MGs established in [17]. Besides RBFs there are various methods to learn the behaviour of a complex function. Artificial Neural Networks (NNs) are one of the most popular representatives of modern artificial intelligence techniques and are often used in practice due to their success in various application fields, see e.g. the survey article [21]. In [22] the authors forecast loads in a power grid using NNs, whereas in [23] NNs are used in an optimal power flow framework. The main advantage of using surrogates is that communication effort can be reduced.

In this paper, we extend the idea of coupled microgrids established in [17] by proposing an iterative *bi-directional* optimization routine in order to improve the overall performance. Due to its iterative structure, however, our method comes along with a strong need for communication between smart homes and grid operator. As a remedy we present two approaches to reduce the communication effort by substituting the optimization on microgrid level via surrogate models. A main difference compared to [19] lies in the different input-output map that is replaced by the surrogate models, for which we can show that each input uniquely determines an (optimal) output. Furthermore, we also take NNs as potential surrogate models into account and study the performance of the resulting approximations numerically in an MPC framework. Our simulations show that the proposed method approximately recovers the performance based on using ADMM but significantly reduces the communication burden. The effect of applying surrogate models within MPC extends our previous work [19] where a surrogate model based on RBFs was only applied in a static optimization problem.

The paper is structured as follows: In Section 2 we formulate a mathematical model for coupled microgrids that consists of two hierarchy levels, and introduce optimization problems corresponding to each of them. In the consecutive section, we propose an iterative scheme that requires the solution of a distributed optimization problem on the lower level which is solved using ADMM. In Section 4, we investigate the impact of disturbances w.r.t. the lower-level solution on the performance measured in terms of the upper-level objective function. Based on the results, we propose to replace ADMM by surrogates in order to reduce communication effort and computation time. The performance of the optimization scheme incorporating surrogates is analysed in an MPC framework in Section 5.

2 A model for coupled microgrids

We consider a system of coupled microgrids (MGs) and call it a *smart grid*. Each MG consists of several residential energy systems (agents) coupled through the grid operator, which can be seen as Central Entity (CE). The coupling of the microgrids is done through a network, where some MGs are connected by a transmission line and others are not connected, cf. Figure 1.

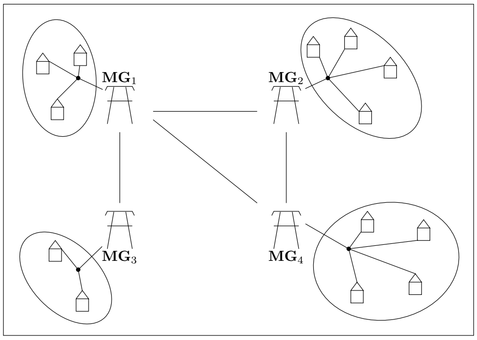

Fig. 1: Upper-level model: Schematic representation of four coupled MGs. Energy exchange is possible only via transmission lines.

2.1 Upper-level model: Energy exchange

We assume that we have $\Xi \in \mathbb{N}$ many MGs which are partially coupled via transmission lines and can be interpreted as nodes of a non-complete graph, see Figure 1 where $\Xi = 4$. Each MG κ, for $\kappa \in [1:\Xi]$, consists of $\mathcal{I}_\kappa \in \mathbb{N}$ agents modeled in detail in Subsection 2.2. We assume that each MG κ has an average power demand $\bar{z}_\kappa(n)$ at time n. Given this, we can compute the total power demand $\mathcal{I}_\kappa \bar{z}_\kappa$ of a MG. The control goal is to exchange power among the MGs in a way such that a desired quantity $\bar{\zeta}(n)$ is targeted by controlling the residential storage units. We specify $\bar{\zeta}(n)$ in Subsection 2.2, and assume for the moment that this quantity is known to the grid and has advantages for the grid operation. We assume that this desired quantity is independent of κ, but this is not necessary for the rest of the discussion.

Let $\delta_{\nu\kappa}$ describe the percentage of power $\mathcal{I}_\nu \bar{z}_\nu(n)$ that is transferred from MG ν to MG κ. We enforce $\delta_{\nu\kappa}$ equals zero if there is no transmission line between the two MGs. Otherwise, the power demand of a MG κ is given by its own total power demand $\delta_{\kappa\kappa}\mathcal{I}_\kappa \bar{z}_\kappa$, where $\delta_{\kappa\kappa}$ is what remains at the MG, and the sum over the power received from connected MGs, $\sum_{\nu\neq\kappa} \delta_{\nu\kappa}\mathcal{I}_\nu \bar{z}_\nu$. For each time step in our prediction horizon of length $N \in \mathbb{N}_{\geq 2}$, we want to match this to the desired power demand starting at time k for N timesteps of each MG in a least-squares sense. The objective function is thus given by $\mathcal{J} : \mathbb{R}^{\Xi N} \times \mathbb{R}^{\Xi\times\Xi\times N} \to \mathbb{R}$,

$$(\bar{z}, \delta) \mapsto \sum_{n=k}^{k+N-1} \sum_{\kappa=1}^{\Xi} \left(\bar{\zeta}(n)\mathcal{I}_\kappa - \sum_{\nu=1}^{\Xi} \delta_{\nu\kappa}(n)\eta_{\nu\kappa}\mathcal{I}_\nu \bar{z}_\nu(n) \right)^2 . \tag{1}$$

Here, the vector $\bar{z} = (\bar{z}(k), \ldots, \bar{z}(k+N-1)^\top)^\top$ with $\bar{z}(\cdot) \in \mathbb{R}^\Xi$ stacks the average power demand per MG and time step while the matrix $\eta = (\eta_{\nu\kappa})_{\kappa,\nu=1}^{\Xi} \in [0,1]^{\Xi\times\Xi}$ incorporates efficiencies along the transmission lines.

We are interested in minimizing (1) under the following constraints: All exchange rates $\delta_{\nu\kappa}$ are within the interval $[0,1]$, sum up to 1, and only transfer power in one direction, meaning that either $\delta_{\nu\kappa}$ or $\delta_{\kappa\nu}$ is zero. Moreover, note that at this grid level, the average power demands per MG are known. Following [17], the optimization problem of the upper-level is, thus, formulated over the exchange rates δ,

$$\min_{\delta\in\Delta} \quad \mathcal{J}(\bar{z}, \delta) \tag{2a}$$

$$\text{s.t.} \quad \sum_{\kappa=1}^{\Xi} \delta_{\nu\kappa}(n) = 1 \tag{2b}$$

$$\delta_{\nu\kappa}(n) \cdot \delta_{\kappa\nu}(n) \leq 0, \ \ \kappa \neq \nu \tag{2c}$$

$$\forall\, \nu, \kappa \in [1:\Xi],\ n \in [k : k+N-1],$$

where $\delta_{\nu\kappa}(n)$ denotes the power exchange rate from MG ν to MG κ at time instance n, and $\Delta := \left\{ \delta \in [0,1]^{\Xi\times\Xi\times N} \,|\, \delta_{\nu\kappa}(n) = 0, \text{ if no line from } \nu \text{ to } \kappa \right\}$. Constraints (2b) and (2c) ensure that the whole energy of each MG is *scheduled* and that exchanges via transmission lines can only occur in one direction during one time step. The definition of the set Δ encodes the grid topology and, hence, avoids scheduling energy exchange in the absence of a transmission line, cf. Figure 1. We denote the feasible set of (2) by

$$\mathbb{D}^\delta = \left\{ \delta \in \Delta \subseteq [0,1]^{\Xi\times\Xi\times N} \;\middle|\; \text{(2b) - (2c) hold} \right\}.$$

The efficiency of a transmission line does not depend on the direction of the transfer, i.e. the matrix η is symmetric. Furthermore, we assume no loss without transport, i.e. $\eta_{\kappa\kappa} \equiv 1$ for all $\kappa \in [1:\Xi]$, in the rest of the paper.

2.2 Lower-level model: Single microgrid

As we have seen in the previous section, we consider an average power demand at each MG as well as some desired quantity $\bar{\zeta}(n)$. In order to understand these quantities better, we explain the modeling of the MG in all detail. The basis of our considerations forms the model presented in [24] and its extension [7]. Therefore, let each subsystem be equipped with an energy generation device (e.g. roof top photo-voltaic panels) and some storage device (e.g. a battery). Then, the i-th system, $i \in [1 : \mathcal{I}_\kappa]$ in MG κ, can be described by the discrete time system dynamics,

$$x_{\kappa_i}(n+1) \;=\; \alpha_{\kappa_i} x_{\kappa_i}(n) + T(\beta_{\kappa_i} u^+_{\kappa_i}(n) + u^-_{\kappa_i}(n)) \quad \text{(3a)}$$

$$z_{\kappa_i}(n) \;=\; w_{\kappa_i}(n) + u^+_{\kappa_i}(n) + \gamma_{\kappa_i} u^-_{\kappa_i}(n), \quad \text{(3b)}$$

where $x_{\kappa_i}(n)$ and $z_{\kappa_i}(n)$ denote the State of Charge (SoC) of the battery and the power demand at time instance $n \in \mathbb{N}_0$, respectively. The latter incorporates the net consumption, $w_{\kappa_i}(n) = \ell_{\kappa_i}(n) - g_{\kappa_i}(n)$, as the difference of load and power generation, cf. Figure 2.

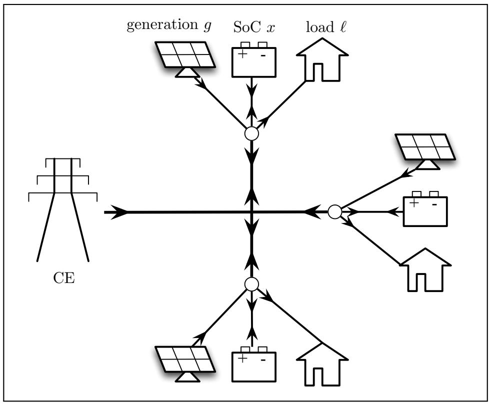

Fig. 2: Lower-level model: Star-shaped network of residential energy systems and central entity (CE). The quantity $w = \ell - g$ in (3b) is obtained from the data set [25].

The system can be controlled by charging $u^+_{\kappa_i}$ and discharging $u^-_{\kappa_i}$ the battery at each time step. The length of a time step in hours is denoted by $T > 0$, e.g. $T = 0.5$ corresponds to a half-hour time window. The constants $\alpha_{\kappa_i}, \beta_{\kappa_i}, \gamma_{\kappa_i} \in (0, 1]$ represent efficiencies w.r.t. self discharge and energy conversion. Furthermore, the initial SoC $x_{\kappa_i}(k) = \hat{x}_{\kappa_i}$, where $k \in \mathbb{N}_0$ denotes the current time instance, is assumed to be known. State and input are subject to the inequality constraints,

$$\begin{aligned} 0 \;&\leq\; x_{\kappa_i}(n) \;\leq\; C_{\kappa_i} && \text{(4a)}\\ \underline{u}_{\kappa_i} \;&\leq\; u^-_{\kappa_i}(n) \;\leq\; 0 && \text{(4b)}\\ 0 \;&\leq\; u^+_{\kappa_i}(n) \;\leq\; \bar{u}_{\kappa_i} && \text{(4c)}\\ 0 \;&\leq\; \frac{u^-_{\kappa_i}(n)}{\underline{u}_{\kappa_i}} + \frac{u^+_{\kappa_i}(n)}{\bar{u}_{\kappa_i}} \;\leq\; 1. && \text{(4d)} \end{aligned}$$

Here, $C_{\kappa_i} \geq 0$ denotes the battery capacity. The last constraint ensures that the bounds on discharging (4b) and charging (4c) hold even if the battery is both discharged and charged during one time step. Note that the case $C_{\kappa_i} = 0$ covers the case, where not all systems have a storage device. Since the future net consumption is not known in advance, it is assumed to be reliably predictable on a short time horizon of size N, $N \in \mathbb{N}_{\geq 2}$, time steps.

For a concise notation we introduce the set $\mathbb{X}_{\kappa_i} = [0, C_{\kappa_i}]$ of feasible states, the set $\mathbb{U}_{\kappa_i} = \left\{ (u^+_{\kappa_i}, u^-_{\kappa_i})^\top \in \mathbb{R}^2 \;\middle|\; \text{(4b)} - \text{(4d) hold} \right\}$ of feasible control pairs and the set

$$\mathbb{D}_{\kappa_i} = \left\{ z_{\kappa_i} \in \mathbb{R}^N \;\middle|\; \begin{array}{c} z_{\kappa_i} = (z_{\kappa_i}(k), \ldots, z_{\kappa_i}(k+N-1))^\top \\ \exists\, u_{\kappa_i} \in \mathbb{U}^N \text{such that} \\ x_{\kappa_i}(k) = \hat{x}_{\kappa_i}, \text{(3) and (4) hold} \end{array} \right\}$$

of feasible outputs over the next N time steps, $i \in [1 : \mathcal{I}_\kappa]$, $\kappa \in [1 : \Xi]$. Referring to the feasible sets of a MG κ we use the Cartesian product, e.g. $\mathbb{D}^{(\kappa)} = \mathbb{D}_{\kappa_1} \times \ldots \times \mathbb{D}_{\kappa_{\mathcal{I}_\kappa}}$ and $z^{(\kappa)} \in \mathbb{D}^{(\kappa)}$. Note that the sets $\mathbb{D}_{\kappa_i}$, $i \in [1 : \mathcal{I}_\kappa]$, and hence $\mathbb{D}^{(\kappa)}$ are non-empty, compact, and convex.

The output quantity in (3b) is the power demand z_{κ_i} of an individual agent in MG κ. The average power demand $\bar{z}_\kappa = \frac{1}{\mathcal{I}_\kappa} \sum_{i=1}^{\mathcal{I}_\kappa} z_{\kappa_i}$ in each MG can then be computed from the individual power demands, and is used as an input to (1). We define $\bar{\zeta}$ in (1) as a stable reference trajectory by averaging over a past time horizon and over all individual residential units of all microgrids, the so-called overall average net consumption as proposed in [7],

$$\bar{\zeta}(n) \;=\; \frac{1}{\mathcal{I} \cdot \min\{N, n+1\}} \sum_{j=n-\min\{n,N-1\}}^{n} \; \sum_{i=1}^{\mathcal{I}} w_i(j) \quad \text{(5)}$$

where $\mathcal{I} = \sum_{\kappa=1}^{\Xi} \mathcal{I}_\kappa$ denotes the total number of agents within the entire smart grid. Due to averaging, the trajectory $\bar{\zeta}$ has little fluctuations and yields advantages for the grid operation.

Let us for the moment ignore the coupling described in Subsection 2.1. Then, $\delta(n)$ equals the identity for all $n \in [k : k+N-1]$ and $\mathcal{J}$ becomes

$$\bar{z} \mapsto \sum_{n=k}^{k+N-1} \sum_{\kappa=1}^{\Xi} \left(\bar{\zeta}(n)\mathcal{I}_\kappa - \mathcal{I}_\kappa \bar{z}_\kappa(n) \right)^2 .$$

Therefore, the overall objective $\mathcal{J}$ can be decoupled yielding the local optimization problem

$$\min_{z^{(\kappa)} \in \mathbb{D}^{(\kappa)}} \quad g(\bar{z}_\kappa) \quad \text{(6)}$$

per MG with local objective function $g : \mathbb{R}^N \to \mathbb{R}_{\geq 0}$,

$$g(\bar{z}_\kappa) = \left\| \bar{\zeta} - \bar{z}_\kappa \right\|_2^2 .$$

2.3 Fully coupled optimization problem

We are interested in optimizing the function (1). This function, in general, depends on δ as well as on $\bar{z}$. As seen in the previous section, the average power demand $\bar{z} = \bar{z}(u)$ depends on the control u, which we have to find in such a way that $\mathcal{J}$ is optimal. The overall optimization problem can be written as

$$\min_{\delta \in \mathbb{D}^{\delta}, z^{(\kappa)} \in \mathbb{D}^{(\kappa)}} \quad \mathcal{J}(\bar{z}, \delta). \tag{7}$$

Note that due to constraint (2c) the optimization of $\mathcal{J}$ w.r.t. δ is non-convex. Furthermore, the large scale of the optimization w.r.t. $\bar{z}$ causes the use of a centralized solver to be expensive. In addition, using a centralized solver assumes the existence of a global entity gathering the information of the whole grid, in particular the personal data of each household, which is undesirable in practice. Hence, solving (7) centralized is impractical. In the subsequent section we present an approach to tackle (7) by solving the upper and lower-level problem iteratively. Doing so we avoid a node with full knowledge in the grid and only communicate specific aggregated information in each iteration among the agents.

3 Bidirectional optimization

We propose to tackle the optimization problem (7) in a *bidirectional* way, i.e. we first find an optimal $\bar{z}$ for δ being the identity and then optimize (7) w.r.t. δ for fixed $\bar{z}$ in order to find the optimal exchange strategy. This typically already gives a considerable improvement, and has been also done e.g. in [17]. To solve (2) for a fixed $\bar{z}$ is straight forward, and we use a standard Sequential Quadratic Programming (SQP) solver. We refer to [26] for an introduction to SQP methods. In this paper we show how to incorporate the computed energy exchange from the upper level into the lower-level optimization problem in order to improve the overall performance.

3.1 Bidirectional optimization scheme

Assume that each MG κ, $\kappa \in [1:\Xi]$, within the smart grid has already solved its inherent optimization problem (6) and based on the corresponding solutions $\bar{z}_\kappa$ an energy exchange policy $\delta^\star$ has been computed according to (2). This exchange yields an updated power demand

$$\bar{z}_\kappa^+(n) = \frac{1}{\mathcal{I}_\kappa} \sum_{\nu=1}^{\Xi} \delta_{\nu\kappa}^\star(n)\, \eta_{\nu\kappa}\, \mathcal{I}_\nu\, \bar{z}_\nu(n), \;\; n \in [k : k+N-1],$$

and hence, the difference $\Delta\bar{z}_\kappa(n) = \bar{z}_\kappa(n) - \bar{z}_\kappa^+(n)$ in power demand for all MGs. We are interested in updating $\bar{z}$ in such a way that our cost function (1),

$$\begin{aligned} \mathcal{J}(\bar{z}, \delta^\star) &= \sum_{n=k}^{k+N-1} \sum_{\kappa=1}^{\Xi} \left(\bar{\zeta}(n)\, \mathcal{I}_\kappa - \sum_{\nu=1}^{\Xi} \delta_{\nu\kappa}^\star(n)\, \eta_{\nu\kappa}\, \mathcal{I}_\nu\, \bar{z}_\nu(n) \right)^2 \\ &= \sum_{\kappa=1}^{\Xi} \mathcal{I}_\kappa^2 \left\| \bar{\zeta} - \bar{z}_\kappa^+ \right\|_2^2 = \sum_{\kappa=1}^{\Xi} \mathcal{I}_\kappa^2 \left\| (\bar{\zeta} + \Delta\bar{z}_\kappa) - \bar{z}_\kappa \right\|_2^2 \end{aligned}$$

is minimized further. One could think about fixing δ and finding an optimal $\bar{z}$. This, however, leads to an optimization problem not avoiding communication and coupling all microgrids. The trick here is now to fix not only the δ but also the $\bar{z}$-components from all the microgrids but one. This leads to optimizing $\mathcal{I}_\kappa^2 \left\| (\bar{\zeta} + \Delta\bar{z}_\kappa) - \bar{z}_\kappa \right\|_2^2$ locally in each MG, where $\Delta\bar{z}_\kappa$ is computed by using the $\bar{z}_\kappa$'s and δ's from the previous optimization step. Intuitively, the difference $\Delta\bar{z}_\kappa$, $\kappa \in [1:\Xi]$, can be interpreted as an additional load or generation, and, therefore, as a change of the desired power demand profile for MG κ. This yields the modified lower-level optimization problem

$$\min_{z^{(\kappa)} \in \mathbb{D}^{(\kappa)}} \quad g_\kappa(\bar{z}_\kappa) = \left\| \bar{\zeta}_\kappa^+ - \bar{z}_\kappa \right\|_2^2, \tag{8}$$

where $\bar{\zeta}_\kappa^+ = \bar{\zeta} + \Delta\bar{z}_\kappa$ In this formulation the updated reference trajectories $\bar{\zeta}_\kappa^+$, $\kappa \in [1:\Xi]$, differ among the single MGs and depend on a given δ and a given previous $\bar{z}_\kappa$. The solution of the newly derived lower-level optimization problem can be solved with ADMM for all microgrids independently and in a parallel way.

Based on the updated reference value we solve (8) and (2) to improve the battery usage and the energy exchange and repeat the optimization until some terminal condition is satisfied, e.g. performance improvement less than a pre-defined tolerance or maximal number of iterations exceeded. This procedure is summarized in Algorithm 1. Note that we only update the reference $\bar{\zeta}$ on the lower level, since the upper-level optimization problem (1) does not change.

Neither convergence nor the interpretation of a potential limit of Algorithm 1 is clear a priori. Figure 3, however, experimentally shows convergence of the proposed scheme and a continuous improvement of the upper-level performance index. Here, we ran 10 iterations of the optimization scheme and plotted both the objective function values before and after the energy exchange within each iteration. The values stagnate after four iterations indicating that additional iterations do not further improve the overall performance. The next subsection elaborates on how to solve (8) in a fully distributed way using ADMM.

Algorithm 1 Iterative bidirectional optimization scheme

```
Input: Current time instance $k \in \mathbb{N}_0$, current SoC $x_{\kappa_i}(k) \in \mathbb{X}_{\kappa_i}$, prediction horizon $N \in \mathbb{N}_{\geq 2}$, predicted net consumption $(w_{i_\kappa}(k), \ldots, w_{i_\kappa}(k+N-1))^\top \in \mathbb{R}^N$, $i_\kappa \in [1 : \mathcal{I}_\kappa]$, $\kappa \in [1 : \Xi]$, reference trajectory $(\bar{\zeta}(k), \ldots, \bar{\zeta}(k+N-1))^\top \in \mathbb{R}^N$, maximal number $j_{\max} \in \mathbb{N}$ of iterations, and tolerance $\varepsilon > 0$.
Initialization:
1. Set $j = 0$ and $\delta^0(n) = I_\Xi$ for all $n \in [k : k+N-1]$.
2. Lower level (parallel in $\kappa$). Compute $\bar{z}_\kappa^j$ as the solution of (6) using ADMM.
3. Upper level. Given $\bar{z}_\kappa^j$, solve (2) for $\delta^j$ using SQP.
While $j < j_{\max}$ and $\mathcal{J}(\bar{z}^{j-1}, \delta^{j-1}) - \mathcal{J}(\bar{z}^j, \delta^j) > \varepsilon$
Do:
4. Lower level (parallel in $\kappa$).
   (a) Compute $\bar{\zeta}_\kappa^+$ from $\bar{z}_\kappa^j$ and $\delta^j$.
   (b) Solve (8) using ADMM and send $\bar{z}_\kappa^{j+1}$ to the upper level.
5. Upper level. Given $\bar{z}_\kappa^{j+1}$, solve (2) for $\delta^{j+1}$ using SQP.
6. $j \to j+1$
```

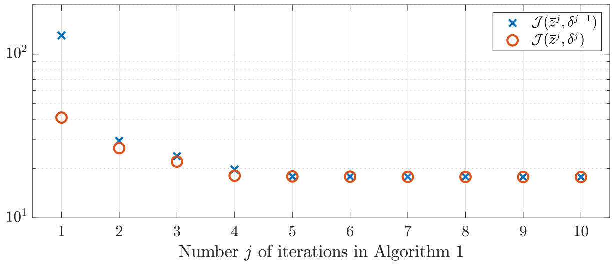

Fig. 3: Evolution of the costs before and after energy exchange computed according to the bidirectional optimization scheme described in Algorithm 1, i.e. $\mathcal{J}(\bar{z}^j, \delta^{j-1})$ and $\mathcal{J}(\bar{z}^j, \delta^j)$, resp. Note that $\mathcal{J}(\bar{z}^1, \delta^0)$ yields the costs without microgrid coupling.

Remark 1. The results displayed in Figure 3 and our numerical investigations indicate convergence to the global minimum of (7). However, a formal (and rigorous) derivation of this conjecture is left for future research.

3.2 Distributed optimization via ADMM

In this section we briefly discuss how to solve the lower-level optimization problem (6) or (8) using an Alternating Direction Method of Multipliers (ADMM) approach. We consider a single MG and therefore omit the index κ. Since the averaged output quantity appears in the objective function (6) or (8), we need to introduce an auxiliary variable a in order to decouple the lower-level optimization in the following way,

$$\min_{z,a} \quad g(\bar{a}) = \|\bar{a} - \bar{\zeta}\|_2^2 \tag{9a}$$

$$\text{s.t.} \quad \frac{1}{\mathcal{I}} \sum_{i=1}^{\mathcal{I}} a_i - \bar{a} = 0, \quad z_i - a_i = 0 \tag{9b}$$

$$z_i \in \mathbb{D}_i \quad \forall\, i \in [1 : \mathcal{I}]. \tag{9c}$$

Note that (9c) is a short-hand notation for the battery dynamics (3)-(4), and yields a fully decoupled constraint in the variable z. ADMM is an optimization scheme to solve (9) based on the augmented Lagrangian $\mathcal{L}_\rho : \mathbb{R}^{\mathcal{I}N} \times \mathbb{R}^{\mathcal{I}N} \times \mathbb{R}^{\mathcal{I}N} \to \mathbb{R}$, for $\rho > 0$,

$$\mathcal{L}_\rho(z, a, \lambda) = g(\bar{a}) + \sum_{i=1}^{\mathcal{I}} \left(\lambda_i^\top (z_i - a_i) + \frac{\rho}{2} \|z_i - a_i\|_2^2 \right),$$

in a distributed way, cf. [5]. Following [7], the ADMM algorithm for (9) yields the three-step iteration $\ell \mapsto \ell + 1$,

$$z_i^{\ell+1} = \operatorname*{arg\,min}_{z_i \in \mathbb{D}_i} z_i^\top \lambda_i^\ell + \frac{\rho}{2} \left\| z_i - a_i^\ell \right\|_2^2 \tag{10a}$$

$$a^{\ell+1} = \operatorname*{arg\,min}_{a \in \mathbb{R}^{\mathcal{I}N}} g(\bar{a}) - \sum_{i=1}^{\mathcal{I}} a_i^\top \lambda_i^\ell + \frac{\rho}{2} \|z_i^{\ell+1} - a_i\|_2^2 \tag{10b}$$

$$\lambda_i^{\ell+1} = \lambda_i^\ell + \rho(z_i^{\ell+1} - a_i^{\ell+1}) \tag{10c}$$

until some termination condition is satisfied. Note that (10b) is an unconstrained optimization problem and can be solved explicitly. The problem (10a) can be solved in parallel by each battery in the MG introduced for our model in Section 2.2. Note that scheme (10) assumes communication within the MG, more precisely, each system i, $i \in [1 : \mathcal{I}]$, sends its optimal solution z_i to the CE and receives both the updated auxiliary a_i and dual variable λ_i in return. The variant discussed in [7] avoids unnecessary communication overhead by returning a broadcast variable instead, which only incorporates information on the aggregated values.

According to Theorem 3.1 in [7] the optimization scheme (10) converges in the following sense.

Theorem 1. Consider Problem (9) with g being strictly convex, closed and proper and let the iterates $(z^\ell, a^\ell, \lambda^\ell)$ be computed according to (10). Then the following following statements hold true:

1. $(z^\ell - a^\ell)_{\ell \in \mathbb{N}_0}$ converges to zero for $\ell \to \infty$,
2. $(g(\bar{a}^\ell))_{\ell \in \mathbb{N}_0}$ converges to the optimal value $g^\star$ of (9),
3. $(\lambda^\ell)_{\ell \in \mathbb{N}_0}$ converges to the dual optimal $\lambda^\star$ of (9).

According to [7] and the references [5, Section 3] and [4, Appendix C] therein, problem (6) fulfils the assumptions of Theorem 1.

4 Surrogate models for ADMM

This section is dedicated to surrogate models for the optimization routine (10) within a single MG. For simplicity of notation we omit the index κ.

Due to the distributive structure of ADMM, the residential energy systems do not need to share information with their neighbours but only with the CE, see also the star-structure in Figure 2. In each iteration ℓ of ADMM, subsystem i has to transmit its solution z_i^ℓ of (10a) to the CE. The optimization scheme presented in Algorithm 1, however, suggests to run ADMM more than once in order to improve the performance w.r.t. to the objective function (1). In order to avoid unnecessary communication, we propose to use surrogate models to approximate the optimization routine (10). More precisely, we are interested in a function which approximates $\varphi : \mathbb{R}^N \times \mathbb{R}^{\mathcal{I}} \times \mathbb{R}^N \to \mathbb{R}^N$:

$$\varphi(\bar{w}, x(k), \bar{\zeta}) \quad = \quad \bar{z}, \tag{11}$$

for all feasible $(\bar{w}, x(k), \bar{\zeta}) \in \mathbb{R}^N \times \mathbb{X} \times \mathbb{R}^N$. Note that we do neither assume knowledge on the local net consumption w_i, $i \in [1 : \mathcal{I}]$, nor on the future SoC $x(n)$, $n > k$.

Figure 4 (top) shows that if the approximation (11) is sufficiently accurate, the impact on the performance of the optimization scheme is negligible. Here, the costs $\mathcal{J}(\bar{z}, \delta)$ after optimization are visualized for 48 consecutive time steps (equals 24-hours simulation time). In the experiment, we disturbed the ADMM solution in Algorithm 1 by uniformly distributed additive noise, i. e., $\bar{z} + 10^{-p} \cdot d$, where the vector $d \sim \mathcal{U}(-1, 1)$, and $p \in \mathbb{N}_0$ denotes the intensity of the disturbance.

Note that (11) might yield approximations to the solution $\bar{z}$ that either aggravate the performance w.r.t. (6) compared to the solution $\bar{w}$ associated with $u \equiv 0$ or solutions that correspond to an infeasible control $\hat{u} \notin \mathbb{U}$. As a remedy, we propose to apply ADMM once after replacing it by a surrogate in the optimization scheme. More precisely, first we run Algorithm 1 using a surrogate in Step 4(b) until the while loop terminates and then we additionally repeat Steps 4 and 5 using ADMM.

4.1 Well-posedness

The following proposition states that for equality in (11), a proper mapping is defined. For a concise notation we replace the index κ_i by i here.

Proposition 1. Consider φ given by (11), where $\bar{z}$ describes the optimal solution of (6) computed via ADMM,

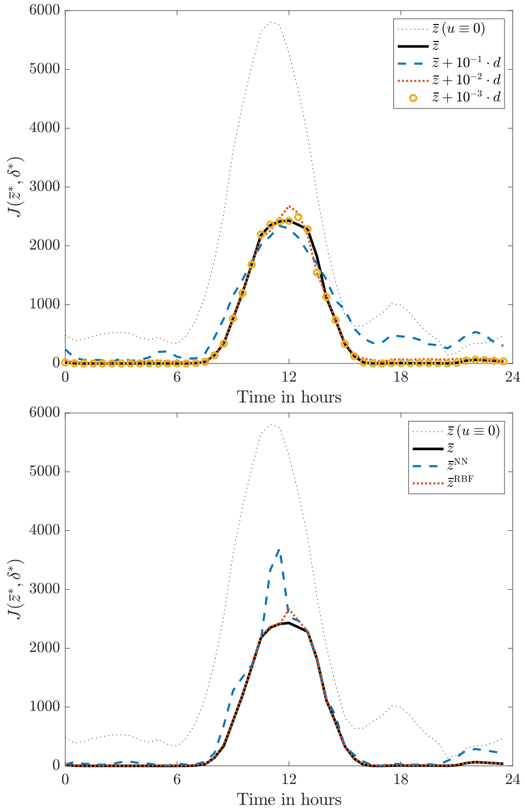

Fig. 4: Effect of mapping error in (11) (top) and of the approximation via radial basis functions (RBFs) and a neural network (NN) on the open-loop performance $\mathcal{J}(\bar{z}, \delta)$ within 48 consecutive time steps (bottom). We use $T = 0.5$h in (3a).

i.e. $\bar{z} = \bar{z}(u^\star)$. We assume all hyper-parameter to be fixed meaning that $\{T, \alpha_i, \beta_i, \gamma_i, C_i, \bar{u}_i, \underline{u}_i\}$ in (3)-(4) are constant over time for all $i \in [1 : \mathcal{I}]$. Then φ is a mapping, i.e. for all $(\bar{w}, x(k), \bar{\zeta}) \in \mathbb{R}^N \times \mathbb{X} \times \mathbb{R}^N$, there exists a uniquely determined $\bar{z} \in \mathbb{R}^N$ such that $\bar{z}$ is the solution to the optimization problem (6).

Proof. First note that ADMM yields the unique solution of (6), see e.g. [5]. Furthermore, there are no constraints on z_i, $i \in [1 : \mathcal{I}]$, and the future SoC can be interpreted as an affine function of the current SoC and the future (dis-)charging rate. Hence, expansion of (3a) and averaging of (3b) yield,

$$\min_{u} \quad \left\| \bar{z}(u) - \bar{\zeta} \right\|_2^2, \quad \text{subject to}$$

$$x_i(k+1+n) = \alpha_i^{n+1} x_i(k) + T \sum_{\ell=k}^{k+n} \alpha_i^{n+k-\ell} \big(\beta_i u_i^+(\ell) + u_i^-(\ell) \big),$$

$$x_i(k) = \hat{x}_i, \quad \text{and constraints (4)},$$

$$\bar{z}(k+n) = \bar{w}(k+n) + \bar{u}^+(k+n) + \bar{\gamma}\bar{u}^-(k+n),$$

$$n \in [0 : N-1],$$

where $\bar{\cdot}$ denotes the corresponding average value w.r.t. all subsystems, in particular $\bar{z}(n) = \frac{1}{\mathcal{I}} \sum_{i=1}^{\mathcal{I}} z_i(n)$. This representation of (6) illustrates that the (predicted) average values $\bar{\zeta} = (\bar{\zeta}(k), \dots, \bar{\zeta}(k+N-1))^\top$ and $\bar{w} = (\bar{w}(k), \dots, \bar{w}(k+N-1))$ and the current SoC $\{x_i(k)\}_{i=1}^{\mathcal{I}}$, uniquely determine the optimal solution $\bar{z}(u^\star)$ obtained by ADMM. □

Remark 2. Note that $\bar{\zeta} \mapsto \bar{z}$ as introduced in [19] does not define a mapping since the solution $\bar{z}$ of (6) not only depends on the reference value $\bar{\zeta}$ but also on the future net consumption $\bar{w}$ and the current SoC $x_i(k)$.

4.2 Radial basis functions approximation

Radial Basis Functions (RBFs) are used to interpolate functions based on a set of sampling data. We briefly recap some basics on RBFs. For a detailed introduction to theory and application see e.g. [20], for a similar approach where RBFs are used to replace ADMM we refer to [19].

Let $M \in \mathbb{N}$ denote the number of samples. Then, the interpolation function of (11) is given as the sum of basis functions $\psi_m : \mathbb{R}^N \times \mathbb{X} \times \mathbb{R}^N \to \mathbb{R}$, $m \in [1:M]$, and a regularization term $q : \mathbb{R}^N \times \mathbb{X} \times \mathbb{R}^N \to \mathbb{R}^N$. More precisely,

$$\bar{z} \approx \varphi^{\mathrm{RBF}}(\chi) = \sum_{m=1}^{M} \psi_m(\chi)\alpha_m + q(\chi), \tag{12}$$

where $\chi = (\bar{w}, x(k), \bar{\zeta})$ is the joint inputs of Proposition 1, and $\alpha_m \in \mathbb{R}^N$, $m \in [1:M]$. The basis functions are so-called radial basis functions of the form, $\psi_m(\chi) = \psi(\|\chi - \chi_m\|)$, where the kernel ψ yields support close to the sampling data χ_m, $m \in [1:M]$. We choose an affine linear regularization $q(\chi) = \beta_0 + B\chi$. Note that different choices are possible. The missing parameters α_m, β_0 and B are determined by interpolation conditions, cf. [19, 20].

In Figure 5, a possible fit via RBFs is visualized. Here, we interpolated given data from two-weeks of optimization (4540 data points) based on sampling data picking each 25-th data point to train (12). Then, we tested φ^{RBF} on the following day, and plotted the fitting. Our implementation is based on the `Matlab` toolbox `DACE` [27]. Note that the evaluation time of the RBF approximation grows with the number of data points used. Already with 180 data points to train (12) with $N = 6$ causes the function evaluation of φ^{RBF} to be expensive, see Table 1. Using more data points would no longer yield an advantage over using ADMM w.r.t. computation time.

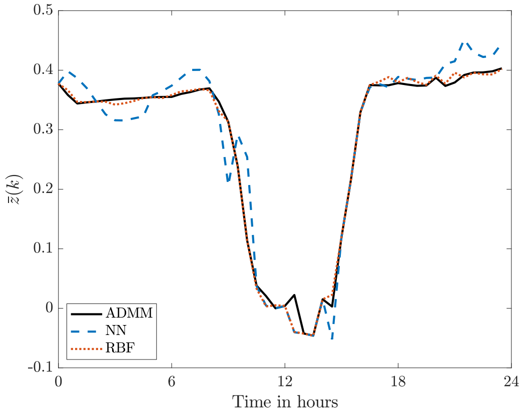

Fig. 5: RBF and neural net fitting of the first component $\bar{z}(k)$ of $\bar{z} = (\bar{z}(k), \dots, \bar{z}(k+N-1))^\top$ within 48 consecutive time steps indicating the quality of the approximation (11).

4.3 Neural networks approximation

Neural Networks (NNs) are a state-of-the-art method in artificial intelligence frameworks. Based on huge amounts of data $M \gg 1$ they are able to learn and recognize patterns in complex systems. We consider a NN of l-layers as an approximation to the mapping (11), i.e.,

$$\bar{z} \approx \varphi^{\mathrm{NN}}(\chi) = \sigma\left(W^{[l]} \; \dots \; \sigma(W^{[2]}\chi + b^{[2]}) \; \dots \; + b^{[l]}\right), \tag{13}$$

where σ denotes the sigmoid function, and the weights $W^{[l]}$ and biases $b^{[l]}$ are determined during the training phase. Here, the number of neurons at layer $l-1$ and at layer l determine the number of rows and columns of $W^{[l]}$, respectively. Note that a separate neural network is trained for each MG. For an introduction to deep learning and neural networks we refer the reader to [28, 29]. To train the NNs, we used `Matlab`'s built-in toolbox `nftool`.

The overall goal of the approximation (13) is to be sufficient in the sense of the MPC performance shown in Figure 6. Our experiments in Figures 4 (bottom) and Figure 5 show that with one hidden layer of ten neurons only, a satisfying approximation on a 24-hours time window can be achieved if the training data is large enough. Note that NNs benefit from big data. In our case study, we trained the NN only on data corresponding to two weeks.

5 Numerical proof-of-concept

Model Predictive Control (MPC) is a method to tackle optimal control problems on an infinite time horizon by solving a series of finite dimensional optimization problems instead, see e.g. [9] for an introduction to non-linear MPC.

5.1 Model predictive control (MPC)

Consider the optimal control problem (6). In order to provide an optimal control sequence over an arbitrary long time horizon we use MPC. To this end, at current time instance $k \in \mathbb{N}_0$ we assume the future net consumption $(w_i(k), w_i(k+1), \ldots, w_i(k+N-1))^\top \in \mathbb{R}^N$ to be predicted for all subsystems $i \in [1:\mathcal{I}]$. Based on the prediction, Algorithm 1 is executed (per MG) to determine control sequences u_i, $i \in [1:\mathcal{I}]$, and an exchange strategy δ. Then, only the first instances $u_i(k)$ and $\delta(k)$ are implemented and the time instance is incremented. Algorithm 2 outlines this MPC scheme.

Algorithm 2 MPC for coupled MGs

```
Input: Current time instance $k \in \mathbb{N}_0$, current SoC $x_i(k) \in \mathbb{X}_{\kappa_i}$, prediction horizon $N \in \mathbb{N}_{\geq 2}$, and reference trajectory $(\bar{\zeta}(k), \ldots, \bar{\zeta}(k+N-1))^\top \in \mathbb{R}^N$.
Repeat:
  1. Measure current state $x_i(k)$ and update the forecast $(w_i(k), w_i(k+1), \ldots, w_i(k+N-1))^\top$, $i \in [1:\mathcal{I}]$.
  2. Run Algorithm 1 for all MG to get optimal control sequences $u_i^\star = (u_i^\star(k), \ldots, u_i^\star(k+N-1))^\top$ for all subsystems $i \in [1:\mathcal{I}]$, and an optimal exchange strategy $\delta^\star = (\delta^\star(k), \ldots, \delta^\star(k+N-1))$.
  3. Implement $u_i^\star(k)$, $i \in [1:\mathcal{I}]$, and $\delta^\star(k)$ and shift the time instance $k \to k+1$.
```

Note that Problems (6) or (8) and (2) have to be solved in order to determine $\bar{z}^\star$ and $\delta^\star$ in each MPC iteration. Therefore, the open-loop costs $\mathcal{J}(\bar{z}^\star, \delta^\star)$ can be computed in each iteration as well (cf. Figure 4). However, since Step 3 in Algorithm 2 suggests to only implement the first instance of the controls computed in Step 2, these costs are not attained. Instead the stage costs

$$\sum_{\kappa=1}^{\Xi} \left(\bar{\zeta}(k)\mathcal{I}_\kappa - \sum_{\nu=1}^{\Xi} \delta^\star_{\nu\kappa}(k)\eta_{\nu\kappa}\mathcal{I}_\nu \bar{z}^\star_\nu(k) \right)^2 \tag{14}$$

are realized at each time step $k \in \mathbb{N}_0$ (cf. Figure 6).

5.2 Usage of surrogate models in MPC

We compare the performances using ADMM, RBFs, and NNs on the lower-level, i.e. in Step 4(b) of Algorithnm 1. In all numerical simulations we set $T = 0.5$, $N = 6$, $\Xi = 4$, $\mathcal{I}_1 = 50$, and $\mathcal{I}_2 = \mathcal{I}_3 = \mathcal{I}_4 = 10$.[1] The battery parameters were randomly chosen with mean values $C = 0.98$, $\underline{u} = -0.24$, and $\bar{u} = 0.25$. Based on the battery capacities we set $\hat{x}_i = 0.5C_i$. In order to incorporate losses along the transmission lines, we used the efficiency matrix,

$$\eta = \begin{bmatrix} 1.0 & 0.9 & 0.9 & 0.85 \\ 0.9 & 1.0 & 0.0 & 0.85 \\ 0.9 & 0.0 & 1.0 & 0.0 \\ 0.85 & 0.85 & 0.0 & 1.0 \end{bmatrix} \text{ in (1).}$$

For simplicity of the numerical computation, we only replaced the lower-level optimization routine for MG 1 and, thus, avoid training a separate surrogate model for each MG. We used Matlab for implementation.

Results on the MPC closed loop can be found in Figure 6 and Table 1. In Figure 6 the closed-loop performances of ADMM (black line) compared to perturbed ADMM, and ADMM (black line) compared to the two

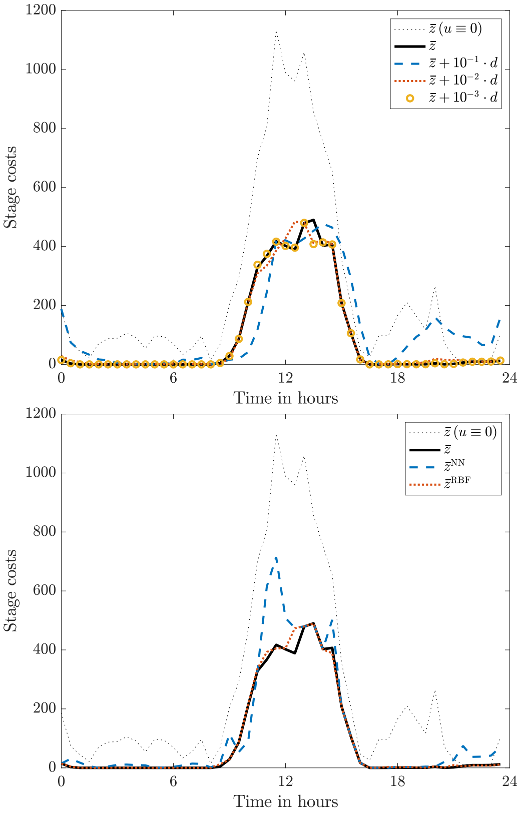

Fig. 6: Impact of mapping error (top) and approximation via RBF and NN (bottom) on the stage costs (14) within 48 consecutive time steps.

[1] Note that this setting yields the global optimization problem (7) with more than 1000 variables.

	closed-loop cost	**runtime [ms]**
no control	**12,228**	—
ADMM	**4,416**	**2.5**
RBFs	**4,529**	**1.2**
NNs	**5,598**	**0.05**

Tab. 1: Comparison of the summed MPC closed-loop performance $\sum_{k=0}^{47} \sum_{\kappa=1}^{\Xi} \left(\bar{\zeta}(k)\mathcal{I}_\kappa - \sum_{\nu=1}^{\Xi} \delta_{\nu\kappa}^\star(k)\eta_{\nu\kappa}\mathcal{I}_\nu \bar{z}_\nu^\star(k) \right)^2$ and runtime (per call): ADMM vs. RBFs vs. NNs.

surrogate models are visualized. Similar to the open-loop case, small disturbances in ADMM have little impact and RBFs outperform the NN. The first column of Table 1 compares the sum of all MPC closed-loop performances using ADMM, RBFs and a NN while in column 2 the average runtimes of these approaches are reported. Note that when using a surrogate, we call ADMM once per MPC iteration. As elaborated in [7] in each ADMM iteration an N-dimensional vector has to be transmitted twice. Hence, both surrogates reduce the need for communication. Two great advantages of ADMM are that the local optimization (10a) can be parallelized and the global optimization is independent of the size of the MG. However, a single function evaluation such as (12) or (13) is faster than running the entire ADMM optimization routine.

Note that in column 2 of Table 1 we ignored the communication between smart homes and CE which is needed to apply ADMM in practice. However, the runtime of ADMM impairs when executed in an actual smart grid while surrogates do not require additional communication.

In order to improve the performance of the NN, more sampling data has to be generated to increase the training set significantly. To avoid large offline computation times, we chose $N = 6$, i.e. a prediction horizon of three hours, which is rather short compared to [7, 19].

Remark 3. We point out two implementation details to solve (2) efficiently. First, note that the optimization (2) can be parallelized in n, since there is no coupling. Furthermore, we replace (2c) by

$$\delta_{\kappa\nu}(n) \cdot \delta_{\nu\kappa}(n) \;\leq\; \varepsilon$$

for some tolerance $\varepsilon > 0$ to smooth the feasible set.

6 Conclusions

In this paper we recalled an optimization problem arising in large-scale electrical networks. We proposed an iterative *bidirectional* optimization scheme to tackle this problem in a distributed way, and showed numerically that a small error on the lower level does not have noticeable impact on the performance. Based on this observation, we replaced the lower-level optimization by surrogate models using radial basis functions and artificial neural networks. The numerical results show the potential of using these surrogates to reduce communication effort and computational time in MPC while preserving the overall performance.

Funding: The authors gratefully acknowledge funding by the Federal Ministry for Education and Research (BMBF; grants 05M18EVA and 05M18SIA). Karl Worthmann is indebted to the German Research Foundation (DFG; grant WO 2056/6-1).

References

[1] S. Parhizi, H. Lotfi, A. Khodaei, and S. Bahramirad. State of the Art in Research on Microgrids: A Review. *IEEE Access*, 3(1):890–925, 2015.

[2] D. E. Olivares, C. A. Cañizares, and M. Kazerani. A centralized optimal energy management system for microgrids. In *2011 IEEE Power and Energy Society General Meeting*, pages 1–6, 2011.

[3] R. Okubo, S. Yoshizawa, Y. Hayashi, S. Kawano, T. Takano, and N. Itaya. Decentralized Charging Control of Battery Energy Storage Systems for Distribution System Asset Management. In *2019 IEEE Milan PowerTech*, pages 1–6, 2019.

[4] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Belmont, MA, USA: Athena Scientific, 1989.

[5] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

[6] B. Houska, J. Frasch, and M. Diehl. An Augmented Lagrangian Based Algorithm For Distributed Nonconvex Optimization. *SIAM J. Optim.*, 26(2):1101–1127, 2016.

[7] P. Braun, T. Faulwasser, L. Grüne, C. M. Kellett, S. R. Weller, and K. Worthmann. Hierarchical distributed ADMM for predictive control with applications in power networks. *IFAC J. Syst. Control*, 3:10 – 22, 2018.

[8] A. Engelmann, Y. Jiang, T. Mühlpfordt, B. Houska, and T. Faulwasser. Towards Distributed OPF using

ALADIN. *IEEE Trans. Power Syst.*, 34(1):584–594, 2019.

[9] L. Grüne and J. Pannek. *Nonlinear Model Predictive Control. Theory and Algorithms.* Springer, London, 2 edition, 2017.

[10] M. Khalid and A. V. Savkin. A model predictive control approach to the problem of wind power smoothing with controlled battery storage. *Renewable Energy*, 35(7):1520–1526, 2010.

[11] A. Parisio, E. Rikos, and L. Glielmo. A model predictive control approach to microgrid operation optimization. *IEEE Trans. Control Syst. Technol.*, 22(5):1813–1827, 2014.

[12] G. Graditi, M. L. Di Silvestre, R. Gallea, and E. R. Sanseverino. Heuristic-Based Shiftable Loads Optimal Management in Smart Micro-Grids. *IEEE Trans. Ind. Informat.*, 11(1):271–280, 2015.

[13] P. Braun, L. Grüne, C. M. Kellett, S. R. Weller, and K. Worthmann. Model Predictive Control of Residential Energy Systems Using Energy Storage & Controllable Loads. *Progress in Industrial Mathematics at ECMI 2014. Mathematics in Industry*, 22:617–623, 2016.

[14] R. R. Appino, J. Á. G. Ordiano, R. Mikut, T. Faulwasser, and V. Hagenmeyer. On the use of probabilistic forecasts in scheduling of renewable energy sources coupled to storages. *Applied Energy*, 210:1207–1218, 2018.

[15] R. R. Appino, M. Muñoz-Ortiz, J. Á. G. Ordiano, R. Mikut, V. Hagenmeyer, and T. Faulwasser. Reliable Dispatch of Renewable Generation via Charging of Time-varying PEV Populations. *IEEE Trans. Power Syst.*, 34(2):1558–1568, 2018.

[16] R. H. Lasseter. Smart distribution: Coupled microgrids. *Proceedings of the IEEE*, 99(6):1074–1082, 2011.

[17] P. Braun, P. Sauerteig, and K. Worthmann. Distributed optimization based control on the example of microgrids. In M. J. Blondin, P. M. Pardalos, and J. S. Sáez, editors, *Computational Intelligence and Optimization Methods for Control Engineering*, volume 150 of *Springer Optimization and Its Applications*, pages 173–200. Springer International Publishing, 2019.

[18] A. Sinha, P. Malo, and K. Deb. A Review on Bilevel Optimization: From Classical to Evolutionary Approaches and Applications. *arXiv preprint arXiv: 1705.06270*, 2017.

[19] S. Grundel, P. Sauerteig, and K. Worthmann. Surrogate Models For Coupled Microgrids. In I. Faragó, F. Izsák, and P. Simon, editors, *Progress in Industrial Mathematics at ECMI 2018*, volume 30. Springer International Publishing, 2019.

[20] M. D. Buhmann. *Radial basis functions: Theory and implementations*, volume 12. Cambridge university press, 2003.

[21] O. I. Abiodun, A. Jantan, A. E. Omolara, K. V. Dada, N. A. Mohamed, and H. Arshad. State-of-the-art in artificial neural network applications: A survey. *Heliyon*, 4(11), 2018.

[22] H. Zhang, F. Xu, and L. Zhou. Artificial neural network for load forecasting in smart grid. In *2010 International Conference on Machine Learning and Cybernetics*, volume 6, pages 3200–3205, 2010.

[23] P. Siano, C. Cecati, H. Yu, and J. Kolbusz. Real Time Operation of Smart Grids via FCN Networks and Optimal Power Flow. *IEEE Trans. Ind. Informat.*, 8(4):944–952, 2012.

[24] K. Worthmann, C. M. Kellett, P. Braun, L. Grüne, and S. R. Weller. Distributed and Decentralized Control of Residential Energy Systems Incorporating Battery Storage. *IEEE Trans. Smart Grid*, 6(4):1914–1923, 2015.

[25] E. L. Ratnam, S. R. Weller, C. M. Kellett, and A. T. Murray. Residential load and rooftop PV generation: an Australian distribution network dataset. *Internat. J. Sustain. Energy*, 2015.

[26] J. Nocedal and S. J. Wright. *Numerical Optimization.* Springer, 2006.

[27] S. N. Lophaven, H. B. Nielsen, and J. Søndergaard. DACE-A Matlab Kriging toolbox, version 2.0. Technical report, 2002.

[28] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning.* MIT Press, 2016.

[29] C. F. Higham and D. J. Higham. Deep Learning: An Introduction for Applied Mathematicians. *arXiv preprint arXiv:1801.05894*, 2018.