Practical Considerations of DER Coordination with Distributed Optimal Power Flow

Daniel Gebbran  
University of Sydney  
Sydney, Australia  
daniel.gebbran@sydney.edu.au

Sleiman Mhanna  
University of Melbourne  
Melbourne, Australia  
sleiman.mhanna@unimelb.edu.au

Archie C. Chapman  
University of Queensland  
Brisbane, Australia  
archie.chapman@uq.edu.au

Wibowo Hardjawana  
University of Sydney  
Sydney, Australia  
wibowo.hardjawana@sydney.edu.au

Branka Vucetic  
University of Sydney  
Sydney, Australia  
branka.vucetic@sydney.edu.au

Gregor Verbič  
University of Sydney  
Sydney, Australia  
gregor.verbic@sydney.edu.au

Abstract—The coordination of prosumer-owned, behind-the-meter distributed energy resources (DER) can be achieved using a multiperiod, distributed optimal power flow (DOPF), which satisfies network constraints and preserves the privacy of prosumers. To solve the problem in a distributed fashion, it is decomposed and solved using the alternating direction method of multipliers (ADMM), which may require many iterations between prosumers and the central entity (i.e., an aggregator). Furthermore, the computational burden is shared among the agents with different processing capacities. Therefore, computational constraints and communication requirements may make the DOPF infeasible or impractical. In this paper, part of the DOPF (some of the prosumer subproblems) is executed on a Raspberry Pi-based hardware prototype, which emulates a low processing power, edge computing device. Four important aspects are analyzed using test cases of different complexities. The first is the computation cost of executing the subproblems in the edge computing device. The second is the algorithm operation on congested electrical networks, which impacts the convergence speed of DOPF solutions. Third, the precision of the computed solution, including the trade-off between solution quality and the number of iterations, is examined. Fourth, the communication requirements for implementation across different communication networks are investigated. The above metrics are analyzed in four scenarios involving 26-bus and 51-bus networks.

Index Terms—Distributed optimal power flow (DOPF), distributed energy resources (DER), ADMM, prosumers, demand response, communication latency, edge computing.

I. INTRODUCTION

Decentralizing power systems by integrating distributed energy resources (DER) at the prosumer level offers economic and technical benefits for both owners and network operators, but requires careful coordination to minimize negative impacts on the grid [1]. In this context, distributed optimal power flow (DOPF) methods have been shown to successfully coordinate DER [2]–[5], ensuring network constraints are always satisfied whilst also preserving prosumer privacy and prerogatives. However, there is currently limited literature analyzing practical applications of DOPF [2], [6], and important implementation aspects have not been discussed in sufficient detail, such as: (i) the solution time of DOPF on actual distributed hardware, (ii) operation of the algorithm in congested electrical networks, (iii) the precision of the solution and (iv) the communication requirements for implementation, such as latency requirements. To fill this gap in the literature, in this paper we present a DPOF deployment on edge computing devices, and discuss its characteristics and real-world performance.

A. Background

The AC optimal power flow problem is typically solved using interior point methods, because it is a nonconvex problem. Although these methods cannot guarantee global optimality in general (since they solve to local optimality), the resulting solution is guaranteed to be feasible. However, the OPF quickly becomes intractable when considering DER due to the sheer number of variables involved. This motivates investigations into distributed approaches, of which several methods have been applied: dual decomposition, analytic target cascading, auxiliary problem principle, optimality condition decomposition, gradient dynamics, dynamic programming with message passing, and the alternating direction method of multipliers (ADMM). A comprehensive review of their implementations can be found in [7].

ADMM [8] has been widely used to solve large-scale OPF problems [7], as it allows for for flexible decompositions of the original OPF problem. They range from network subregions [9] down to an element-wise (e.g., generators, buses, and lines) decomposition [10]. In ADMM, each of the resulting decomposed parts solves a subproblem and exchanges messages with a central aggregator (or between other agents) until convergence is achieved [11]. A decomposition at the point of connection between prosumers and the network was deemed a practical balance for DER coordination [2]–[4]. It preserves privacy of prosumers and allows for parallelization of subproblems (benefits against centralized approaches), and offers quicker solutions (smaller number of iterations) when compared to fully decentralized approaches. This approach has been demonstrated to successfully coordinate DER in real-world scenarios in a recent Australian trial [2], and can be implemented on edge computing devices (at individual...
prosumers), benefiting from subproblem parallelization to distribute the computational load \cite{6}.

Because this approach is very recent, there is sparse literature and a dearth of information regarding practical considerations for this DER coordination method.

B. Contributions

This work offers important technical insights into modeling and deploying DER coordination methods using DOPF. To offer a solid test bed, part of the subproblems is deployed on a hardware prototype, based on Raspberry Pis 3B+ (RPis) – a small, single-board computer. This allows for a more realistic analysis, emulating an edge computing archetype where prosumer computations are conducted on embedded hardware. The remainder of the problem is solved on a PC. Four different test cases are simulated, involving two networks and two time horizons, which allows for comparison across different setups. The paper focuses on four principal characteristics of the problem, which can be summarized in the following contributions:

- Quantification of computation times for the DOPF implemented across edge computing devices.
- Investigation of algorithm execution on normal operation versus congested system conditions.
- Analysis of solution precision, including trade-offs between solution quality and computational burden.
- Discussion of communication requirements for implementation on modern communication networks.

C. Paper Structure

The remainder of the paper is structured as follows: Section II formulates the DOPF, including the initial problem, the decomposition and the resulting distributed problem formulation. Section III discusses details of the implementation, including algorithm specifications, hardware description and details of the test networks. Section IV presents the results and discusses each of the four main proposed metrics. Finally, Section V presents a general discussion on the results and Section VI finishes with concluding remarks.

II. MOPF FORMULATION

The proposed approach for DER coordination is formulated as a multi-period optimal power flow (OPF) problem. It consists of two levels. At the lower level, prosumers schedule their DER, minimizing energy expenditure\cite{1}. At the upper level, the distribution network system operator (DNSP) coordinates prosumers’ actions to minimize the network objective - whilst abiding by network limits and operational constraints.

The objective function of this problem is:

\[
F(x, z) := f(x) + \sum_{h \in H} g_h(z_h)
\]

\[
= \sum_{t \in T} \left( c_2 (p_{g,t}^+)^2 + c_1 p_{g,t}^+ + c_0 + \sum_{h \in H} \left( c_{i}^{\text{out}} p_{h,t}^{+} - c_{i}^{\text{in}} p_{h,t}^{-} \right) \right),
\]

where \( f(x) \) represents the network OPF objective function (which can include, for example, loss minimization, peak load reduction or minimizing the use of backup diesel as in \cite{2}), \( g_h(z_h) \) are prosumer objective functions for each household \( h \), with a fixed time-of-use tariff for purchasing energy, and a feed-in-tariff for selling energy. \( H \) is the set of prosumers, \( x \) is the set of network variables (active/reactive power flows, and voltages, for each \( t \in T \)), and \( z_h \) is the set of internal variables of prosumer \( h \) for each \( t \in T \) (e.g., battery power flows), which compose the set of variables for all prosumers \( z := \{ z_h \}_{h \in H} \).

The network constraints for a single-phase OPF are shown below\cite{2}. They are given for each bus \( i \in B \), and for each time interval \( t \in T \):

\[
p_{g,t} - p_{h,t} = v_{i,t} \sum_{j \in B} v_{j,t} (g_{ij} \cos \theta_{ij,t} + b_{ij} \sin \theta_{ij,t}),
\]

\[
q_{g,t} - q_{h,t} = v_{i,t} \sum_{j \in B} v_{j,t} (g_{ij} \sin \theta_{ij,t} - b_{ij} \cos \theta_{ij,t}),
\]

\[
v_{r,t} = 1, \quad \theta_{r,t} = 0,
\]

\[
v_t \leq v_{i,t} \leq v_t,
\]

\[
p_{g,t} \leq p_{g,t} \leq p_{g,t}, \quad q_{g,t} \leq q_{g,t} \leq q_{g,t},
\]

where \( p_{g,t}, q_{g,t} \) are the total net active/reactive power from the reference bus, \( p_{h,t}, q_{h,t} \) are the total net active/reactive power to prosumer \( h \) connected to bus \( i \), and \( \theta_{ij,t} = \theta_{i,t} - \theta_{j,t} \) is the angle difference between bus \( i \) and its neighboring bus \( j \). Additionally, \( p_{g,t}, q_{g,t} \) model the power flow equations, \( v_t \) models the reference, and \( p_{g,t}, q_{g,t} \) represent voltage and generator (lower and upper) limits. Moreover, let \( p_{h,t} = p_{h,t}^+ - p_{h,t}^- \) be composed of the non-negative terms \( p_{h,t}^+ \), \( p_{h,t}^- \), representing imported and exported power. The same applies for \( p_{g,t} \).

Each prosumer \( h \in H \) is subject to its own constraints. The equation modeling the power balance is, for each \( t \in T, h \in H \):

\[
p_{h,t} = p_{h,t}^+ + p_{h,t}^- - P_{PV},
\]

where \( p_{h,t} \) is the total net power (exchanged with the grid) of household \( h \), with \( \frac{1}{2}p_{h,t} \leq p_{h,t} \leq \frac{1}{2}p_{h,t} \) is the scheduled battery charging power, with \( p_{h,t}^+ \leq p_{h,t} \leq p_{h,t}^- \) is the household non-controllable (fixed) demand, and \( PV \) is the PV generation power output, which can be curtailed if necessary (the total available PV power is \( p_{h,t}^+ \)).

The battery constraints are, for each \( t \in T, h \in H \):

\[
p_{h,t}^+ = P_{h,t}^+ - P_{h,t}^-,
\]

\[
\text{SoC}_{h,T} \leq \text{SoC}_{h,T},
\]

\[
\text{SoC}_{h,t} = \text{SoC}_{h,0} + \Delta t + (\alpha_{h} P_{h,t}^+ - \beta_{h} P_{h,t}^-)/\eta_{\text{dis}} \Delta t,
\]

A balanced three-phase network is assumed for simplicity. It can be modeled as a single phase. However, the single-phase model can be readily extended, e.g. including unbalanced networks with a combination of single- and three-phase connections \cite{3}, increasing the formulation’s complexity.

\footnote{\textbf{Note that because the second term in (6) is a convex piecewise linear function, at least one of the variables \( p_{h,t}^+ \) and \( p_{h,t}^- \) can be zero at time slot \( t \). This therefore obviates the need to use binary variables.}}
where \(p_{h,t}^{\text{ch}}, p_{h,t}^{\text{dis}} \geq 0\) compose the battery charging/discharging power; \(\text{SoC}_{h,t}\) is the battery state-of-charge, with \(\text{SoC}_{h,t} \leq \text{SoC}_{h,t} \leq \text{SoC}_{h,t}\); \(\hat{\eta}_h\) is the battery charge or discharge efficiency, and \(\Delta t\) is the time interval within \(T\).

To rewrite the problem in its compact form, let the network constraints (2) define a feasible set \(\mathcal{X}\) for the network variables \(x\) and prosumer constraints (3), (4) define a feasible set \(\mathcal{Z}_h\) for the variables \(z_h\) of each prosumer \(h \in \mathcal{H}\). Henceforth, \(x \in \mathcal{X}\) and \(z_h \in \mathcal{Z}_h\), with \(z \in \mathcal{Z}\) (the feasible set for all prosumer variables). We can now write:

\[
\min_{x \in \mathcal{X}, z \in \mathcal{Z}} F(x, z) \tag{5}
\]

Two problems arise if we are to solve this MOPF centrally. First, the privacy of all prosumers is violated, since all data (battery information, consumption data, etc) for each house has to be sent to the central computing entity. Second, the problem is computationally hard because it consists of a non-convex network problem \(\mathcal{H}\). Solving such a large-scale nonlinear problem is extremely challenging, especially given a potentially large number (several tens or even hundreds) of prosumer subproblems. Hence, a distributed approach is applied to solve this MOPF with DR problem.

### A. Decomposed Model

Normally, we would not be able to solve (5) in a distributed fashion. This is because the variables corresponding to the prosumer power consumption appear in both \(\mathcal{X}\) and \(\mathcal{Z}\). To enable a decomposable structure for the problem, we create two copies of all prosumer power profiles, as shown in Fig. 1 introducing the following coupling constraints:

\[
\hat{p}_{h,t} = p_{h,t}, \quad \forall h \in \mathcal{H}, \; t \in \mathcal{T}, \tag{6}
\]

where the left-hand term is a copy for the network problem, \(\hat{p}_{h,t} \in \mathcal{X}\), and the right-hand term is a copy for the prosumer problem, \(p_{h,t} \in \mathcal{Z}_h\).

Now, we can treat prosumer subproblems separately from the network, coupled only through prosumer power consumption. Problem (5) can now be decomposed because \(f(x)\) and \(g_h(z_h)\) are themselves separable. In more detail, duplicating the variables as (6) enables us to rewrite (5) as:

\[
\min_{\hat{x} \in \mathcal{X}, z \in \mathcal{Z}} F(\hat{x}, z), \tag{7a}
\]

subject to: \(\hat{p}_{h,t} = p_{h,t}, \quad \forall h \in \mathcal{H}, \; t \in \mathcal{T}\), \(\hat{z}_h = z_h\)\(^{4}\), \(\hat{\lambda}_{h,t} = \lambda_{h,t}\)\(^{4}\), \(\hat{\eta}_h = \eta_h\)\(^{4}\), \(\text{SoC}_{h,t} \leq \text{SoC}_{h,t} \leq \text{SoC}_{h,t}\). \(\Delta t\) is the time interval within \(T\).

\[\hat{\lambda}_{h,t} = \lambda_{h,t} \tag{7b}\]

where \(\hat{x}\) is the original set of problem variables with the addition of the network copy of prosumer’s power profiles (6), and \(\mathcal{X}\) is the new feasible region of the network problem. Now, the sets of variables \(\mathcal{X}\) and \(\mathcal{Z}\) are decoupled, and (7a) is separable if (7b) is relaxed. The resulting decoupled problem is illustrated in Fig. 1. We will exploit this structure to solve (7) in a distributed fashion.

\(^{4}\)Including (4b) avoids full battery depletion - without considering the next time horizon. Replacing it is recommended for algorithm implementation using a rolling horizon basis.
I. The complexity of problem (9a), which takes the longest for resulting in a total of four different test cases, as seen in Table

| Case | Network | $\mathcal{T}$ | No. of variables | No. of constraints |
|------|---------|---------------|------------------|-------------------|
| 1    | A       | $\mathcal{T}_1$ | 11088            | 9840              |
| 2    | B       | $\mathcal{T}_1$ | 21888            | 19440             |
| 3    | A       | $\mathcal{T}_2$ | 22176            | 19680             |
| 4    | B       | $\mathcal{T}_2$ | 43776            | 38880             |

where $\hat{p}$ and $p$ are vectors composed by all variables $\hat{p}_{h,t}$ and $p_{h,t}$ (7b). $\lambda^k$ is the vector composed by all $\lambda_{h,t}^k$ (9c), $\epsilon_{\text{abs}}, \epsilon_{\text{rel}} \in \mathbb{R}^+$ and their values are, in turn, part of the analysis described in Section V. Using smaller values for these tolerances yields more accurate results. However, this requires a higher number of iterations, which directly impacts the total computation time. This may lead to inefficient tolerance values, which is investigated. Finally, an adaptive residual balancing method is used to update the value of $\rho$ according to the magnitude of residuals, as described in [10].

B. Hardware description

The aggregator subproblem (9a) is solved on a 32 GB RAM, Intel i7-7700, 3.60 GHz PC. Five prosumer subproblems (9b) are solved in parallel on five different Raspberry Pis model 3B+, 1 GB RAM, BCM2837B0, 1.4 GHz (RPis), and the remaining prosumer subproblems are solved serially on the PC. All problems were implemented in Python using Pyomo [13] as a modeling interface, and solved using Ipopt v3.12.11 [14], with linear solver MA27 [15], in both the RPis and the PC. The PC is connected to the internet via WiFi, and acts as a multi-client UDP server. All RPis are connected to the internet via WiFi, and act as UDP clients in an edge computing framework.

C. Test networks

Two low-voltage distribution networks A and B, with 25 and 50 prosumers respectively, have been used for testing the proposed algorithm. They have 26 and 51 buses respectively; their configuration is illustrated in Fig. 2.

Prosumer’s load and PV data used are actual power measurements, with half-hourly resolution on a spring day (2011/11/07), of an Australian low-voltage network. As such, we initially define $\mathcal{T}_1 = \{0, 1, \ldots, 47\}$, $\Delta t_1 = 0.5$. Additionally, we have further split these into 15-minute resolution data sets, in which $\mathcal{T}_2 = \{0, 1, \ldots, 95\}$, $\Delta t_2 = 0.25$.

We have combined networks A and B with $\mathcal{T}_1$ and $\mathcal{T}_2$, resulting in a total of four different test cases, as seen in Table I. The complexity of problem (9a), which takes the longest for each iteration, is also shown.

IV. RESULTS

The results for the four test cases, with varying tolerances, are depicted on Fig. 3. Throughout our tests, we have used $\epsilon_{\text{rel}} = 10 \epsilon_{\text{abs}}$, and $\epsilon_{\text{abs}} \in [10^{-2}, 5 \times 10^{-3}, 10^{-3} \ldots, 5 \times 10^{-6}, 10^{-6}]$ for a total of nine tolerances. Fig. 3(a) shows the number of iterations $k$ each case takes to converge, across different tolerances. It is notable $k$ is very similar across all four cases, and therefore mostly independent of the problem size, which demonstrates the scalability of ADMM [10].

We discuss the results in four areas, namely: computation time, system operation under congested conditions, precision of solutions, and communication requirements.

A. Computation Time

The computation time per iteration is shown in Table II. The term $t_{(9a)}$ refers to the execution time (9a) in the PC, $t_{(9b)}$ is determined by the slowest execution time of (9b) in the RPis, and $t_{(9c)}$ refers to the dual update (9c) execution on the PC. The average total parallel computation time per iteration is shown in the last column, $t_{\text{comp}}$, representing the time per iteration a fully distributed implementation would require.

In hindsight, the solution time for the DOPF subproblem is much more predominant in the total solution time. Although the number of iterations $k$ remains very similar when increasing the size of the problem, the central computation time increases linearly, as seen in Table II and consequently, most of the computation load in Fig. 3(b) stems from solving (9a).
Table III: Solution deviation versus central optimal, maximum and average primal residuals over five different tolerances for test cases 1, 2, 3 and 4.

| $\epsilon$ | Case | $P_{\text{dev}}$ | $r_{\text{max}}$ [W] | $r$ [W] |
|------------|------|------------------|---------------------|--------|
| $10^{-2}$  | 1    | $+57.9$          | 198.64              | 45.21  |
|            | 2    | $+56.2$          | 260.50              | 58.89  |
|            | 3    | $+52.1$          | 101.25              | 31.39  |
|            | 4    | $+61.2$          | 98.12               | 38.26  |
| $10^{-3}$  | 1    | $+5.98$          | 70.958              | 5.547  |
|            | 2    | $+7.42$          | 33.697              | 6.174  |
|            | 3    | $+6.65$          | 10.000              | 3.032  |
|            | 4    | $+7.95$          | 10.000              | 3.439  |
| $10^{-4}$  | 1    | $+1.34$          | 0.8082              | 0.5882 |
|            | 2    | $+1.47$          | 0.8295              | 0.6237 |
|            | 3    | $+1.35$          | 0.4813              | 0.3351 |
|            | 4    | $+1.50$          | 1.0317              | 0.3732 |
| $10^{-5}$  | 1    | $+1.05$          | 0.2894              | 0.0495 |
|            | 2    | $+1.24$          | 0.2088              | 0.0663 |
|            | 3    | $+1.01$          | 0.0408              | 0.0052 |
|            | 4    | $+1.32$          | 0.1290              | 0.0050 |
| $10^{-6}$  | 1    | $+0.99$          | 0.0212              | 0.0031 |
|            | 2    | $+1.18$          | 0.0285              | 0.0043 |
|            | 3    | $+0.97$          | 0.0147              | 0.0011 |
|            | 4    | $+1.28$          | 0.0065              | 0.0011 |

Fig. 4: Number of iterations $k$ across different mixes of energy, for case 1 and $\epsilon_{\text{abs}} = 10^{-4}$.

B. System Operation under Congested Conditions

In real systems, demand and generation vary, which may lead to operation under congested conditions (e.g., over- or under-voltage). DOPF implementations need to be robust against these changes, even if they cause a higher number of iterations.

To test the impact of congested conditions, demand and generation have been modified for Case 1, with a fixed tolerance of $\epsilon_{\text{abs}} = 10^{-4}$. The results in Fig. 4 show the number of iterations $k$ across different mixes of energy. The points at which constraints are active (under- and over-voltage, or input feeder limit) are also denoted in the figure, showing a clear correlation of increased $k$ on operation under congested conditions. The maximum value of $k$ does not exceed 70, roughly twice as high when compared to the average $k$ for normal operation conditions. Moreover, it is visible that a system which has surplus of energy generation converges more rapidly than a system which needs to import more energy from the upstream network.

C. Precision of Solutions

A comparison between the optimal solution $F(x, z)$ of the central problem (5) and each test case is shown in the third column of Table III. The values demonstrate the evolution of the solution precision, showing that there is almost no variation to the end result when using very low tolerance values. Not only that, but the number of iterations to reach convergence (and consequently, the computation time) becomes prohibitive, as seen in Fig. 4.

The physical implication of different tolerances are shown in Table III. It depicts the maximum ($r_{\text{max}}$) and average ($r$) violations of constraint (7b) - the definition of primal residual (10a). In other words, the difference between the copies of prosumer power profiles for the network and for the household. The performance across all cases are similar even if the network sizes and $T$ are different.

D. Communication Requirements

The message size at each iteration between prosumers and aggregator is proportional to the choice of $T$. For $T_1$, the message size is smaller than 1 KB, and for $T_2$ it is smaller than 2 KB. The choice of different communication protocols (UDP/TCP/HTTP) is only marginally relevant, and they are capable of dealing with these message sizes, which are much smaller than the lower limits of current mobile broadband networks download and upload speeds [16, 17].

The actual implementation of the DOPF can utilize different structures between prosumers and the aggregator. The recent Australian trial [2] has utilized an hierarchical structure where groups of prosumers send their information to local computers (Reposit boxes [3]), which then compute prosumer subproblems and communicate to a central aggregator every iteration, sending the final solution (i.e., their scheduling information) back to prosumers when the solution is achieved. However, it is possible to make full use of decentralized implementation of prosumers with edge computing hardware, as shown by the computation times of the prosumer subproblem on RPs.

This would require communication between the aggregator and prosumers at every iteration, all of which would be located within the same geographical region (e.g., in the same low-voltage network neighborhood). The communication could be achieved, for example, with the use of last mile networks (4G and 5G). Modern network technologies offer low latencies for this kind of application. For example, 4G network latency can range from 30 to 160 ms, and upcoming 5G networks will further reduce these values [16]. In parallel, network technologies tailored for the Internet of Things [18], such as LTE-M, NB-IoT and EC-GSM-IoT, could also be used to deploy this communication. These networks have latencies of 300 to 600 ms in areas within the normal cell edge of the radio cell [19].

From the technical aspect, the solution time per iteration of the DOPF, as shown in Table I, is more predominant than the latency delay of last mile networks. If implemented in a 4G network, the latency (assume an average of 100 ms) in cases 1 to 4 would take, respectively, 4.3 %, 2.2 %, 2 % and 1 % of the total time per iteration. Economical aspects could weight...
in more when choosing the appropriate technology to deploy this infrastructure, as well as limiting factors such as low area coverage or poor internet connection (2), (11).

V. GENERAL COMMENTS

The computation time of the DOPF approach grows linearly with the size of the problem, which in turn imposes a limit on the available solution time. For instance, when using a rolling horizon, the window interval for each horizon to be completed must be compatible with the DOPF solution time. For instance, larger networks with over one hundred prosumers, as simulated by the authors in (5), require a longer computation time. This may not be compatible with a five-minute window interval as used by the DOPF in (2), with under fifty prosumers.

The choice of an appropriate tolerance and time horizon \( T \) must take into account the problem size and the available solution time. Moreover, the communication latency and other limitations imposed by the geographical location of prosumers and the aggregator must be accounted for. The computational burden introduced by transforming interval \( T_1 \) into \( T_2 \) is associated with doubling the number of variables and constraints, which in turn doubles the resolution of the problem variables.

Communication networks may not handle well the transmission of data from a very large number of prosumers to the aggregator, which happen in a very short amount of time. This may lead to congestion (data traffic above the network bandwidth) or contention (when many prosumers are trying to transmit data simultaneously) on the communication network. These problems are prone to happen when a large concentration of prosumers (over hundreds or thousands) are concentrated in the same geographical location, sharing the same communication network and a limited quantity of available resources (e.g., spectrum) from the wireless network. Nonetheless, the network latency and the message size of the communication between prosumers and aggregator are not bottlenecks when implementing the DOPF.

A. Future Work

As shown in Table II reducing the computation cost per iteration is of paramount importance for a practical implementation of the DOPF. This may include a number of strategies to reduce the computation time for each step, such as splitting (9a) into smaller subproblems, solved in parallel (2).

Moreover, a model to prevent the aforementioned congestion and contention problems is another suggestion for further research. This would allow for a better utilization of the available communication network resources, by allocating these resources and coordinating data transmission according to the characteristic of the DER coordination problem.

Finally, using an asynchronous ADMM may be of interest, which could improve the robustness of the algorithm against possible communication failures.

VI. CONCLUSION

We have implemented a DER coordination problem using DOPF, on a PC and a hardware prototype of five RPis. The central problem was decomposed and decoupled into a formulation suitable for solution using ADMM. We analyzed four different test cases, investigating the computation time and the number of iterations \( k \) across different tolerances. The effect of operation under congested conditions was shown to impact \( k \). We have shown trade-offs between convergence and computation speed according to solution precision. Finally, the communication requirements for the deployment of similar problems were discussed.

REFERENCES

[1] AEMO, Energy Networks Australia, “Open Energy Networks,” Tech. Rep., 2018.
[2] P. Scott, D. Gordon, E. Franklin, L. Jones, and S. Thiébaux, “Network-aware coordination of residential distributed energy resources,” IEEE Transactions on Smart Grid, vol. 10, no. 6, pp. 6528–6537, Nov. 2019.
[3] P. Andrianesis and M. C. Caramanis, “Optimal grid-distributed energy resource coordination,” in 2019 57th Annual Allerton Conference on Communication, Control, and Computing (Allerton), Sep. 2019.
[4] A. Attarha, P. Scott, and S. Thiébaux, “Affinely adjustable robust ADMM for residential DER coordination in distribution networks,” IEEE Transactions on Smart Grid, vol. 11, no. 2, pp. 1620–1629, Mar. 2020.
[5] J. Guerrero, D. Gebbran, S. Mhanna, A. C. Chapman, and G. Verbič, “Towards a transactive energy system for integration of distributed energy resources: Home energy management, distributed optimal power flow, and peer-to-peer energy trading,” Renewable and Sustainable Energy Reviews, vol. 132, p. 110000, Oct. 2020.
[6] D. Gebbran, G. Verbič, A. C. Chapman, and S. Mhanna, “Coordination of prosumer agents via distributed optimal power flow,” in Proceedings of the 19th International Conference on Autonomous Agents and Multi-agent Systems (AAMAS 2020), May 2020, pp. 1–3.
[7] D. K. Molzahn, F. Dörfler, H. Sandberg, S. H. Low, S. Chakrabarti, R. Baldick, and J. Lavaei, “A survey of distributed optimization and control algorithms for electric power systems,” IEEE Transactions on Smart Grid, vol. 8, no. 6, pp. 2941–2962, Nov. 2017.
[8] 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, vol. 3, pp. 1–122, Jan. 2011.
[9] B. Kim and R. Baldick, “Coarse-grained distributed optimal power flow,” IEEE Transactions on Power Systems, vol. 12, no. 2, pp. 932–939, May 1997.
[10] S. Mhanna, G. Verbič, and A. C. Chapman, “Adaptive ADMM for distributed AC optimal power flow,” IEEE Transactions on Power Systems, vol. 34, no. 3, pp. 2025–2035, May 2019.
[11] J. Guo, G. Hug, and O. K. Tonguz, “On the role of communications plane in distributed optimization of power systems,” IEEE Transactions on Industrial Informatics, vol. 14, no. 7, pp. 2903–2913, July 2018.
[12] D. Bienstock and A. Verma, “Strong NP-hardness of AC power flows feasibility,” Operations Research Letters, vol. 47, no. 6, pp. 494–501, Nov. 2019.
[13] W. E. Hart, J.-P. Watson, and D. L. Woodruff, “Pyomo: modeling and solving mathematical programs in python,” Mathematical Programming Computation, vol. 3, no. 3, pp. 219–260, Sep. 2011.
[14] A. Wächter and L. T. Biegler, “On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming,” Mathematical Programming, vol. 106, no. 1, pp. 25–57, March 2006.
[15] J. Smith, “HSL archive: A collection of Fortran codes for large scale scientific computation,” 2018. [Online]. Available: http://www.hsl.rl.ac.uk/.
[16] OpenSignal, “The State of Mobile Network Experience,” Tech. Rep., 2019.
[17] I. Grigoriek, High Performance Browser Networking. O’Reilly Media, Inc., 2013.
[18] D. Gebbran, A. C. Chapman, and G. Verbič, “The Internet of Things as a facilitator of smart building services,” in 2018 Australasian Universities Power Engineering Conference (AUPEC), Nov. 2018, pp. 1–6.
[19] O. Liberg, M. Sundberg, E. Wang, J. Bergman, and J. Sachs, Cellular Internet of Things, 1st ed. Elsevier Science, 2017.