A Domain Decomposition Approach to Solve Dynamic Optimal Power Flow Problems in Parallel

Nils Schween, Philipp Gerstner, Nico Meyer-Hübner, Viktor Slednev, Thomas Leibfried, Wolf Fichtner, Valentin Bertsch, and Vincent Heuveline

Abstract We propose a parallel solver for linear systems of equations arising from the application of Primal Dual Interior Point methods to Dynamic Optimal Power Flow problems. Our solver is based on the Generalised Minimal Residual method in combination with an additive Schwarz domain decomposition method as preconditioner. This preconditioner exploits the structure of Dynamic Optimal Power Flow problems which, after linearization, is given as a matrix with large diagonal blocks and only a few off-diagonal elements. These elements correspond to intertemporal couplings due to ramping and energy storage constraints and are partially neglected in order to solve the problem in parallel. We test our method on a large-scale optimisation problem and show that a parallel speedup can be obtained.

Keywords Dynamical OPF · Multi-period OPF · Time domain decomposition · Additive Schwarz method · Preconditioner

N. Schween · P. Gerstner · V. Heuveline (✉)
Heidelberg Institute for Theoretical Studies (HITS) and Engineering Mathematics and Computing Lab (EMCL), Heidelberg University, Heidelberg, Germany
e-mail: nils.schween@uni-heidelberg.de; philipp.gerstner@uni-heidelberg.de; vincent.heuveline@uni-heidelberg.de

N. Meyer-Hübner · T. Leibfried
Institute of Electric Energy Systems and High-Voltage Technology, Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany

V. Slednev · W. Fichtner
Chair of Energy Economics, Institute for Industrial Production, Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany

V. Bertsch
Department of Energy Systems Analysis, Institute of Engineering Thermodynamics, German Aerospace Center (DLR) and Institute for Building Energetics, Thermotechnology and Energy Storage, University of Stuttgart, Stuttgart, Germany

V. Bertsch et al. (eds.), *Advances in Energy System Optimization*, Trends in Mathematics, https://doi.org/10.1007/978-3-030-32157-4_4

1 Introduction

While today's power grid infrastructure has been designed for centralised power generation in conventional power plants, the ever increasing share of renewable energy sources (*RES*) leads to an increasingly uncertain, volatile and decentralised generation. In order to ensure a reliable grid operation and to maintain today's security of supply, it is necessary to develop methods to simulate accurately a power grid at high temporal resolutions. This can be done by efficient methods for power grid optimisation, including an accurate consideration of the non-linear and non-convex AC power flow constraints. And these are to be improved.

The task to find the optimal operating state of a given power grid, also known as Optimal Power Flow (*OPF*), is formalized as the minimisation of a cost function with respect to a vector of continuous optimisation variables like the generated active power and the generated reactive power. Dynamic Optimal Power Flow (*DOPF*) considers several OPF problems, each corresponding to one specific point in time. In this case, the power grid's power demand is time-dependent and additional constraints must be taken into account. These constraints couple some of the optimisation variables corresponding to different time steps. Among others, these couplings are introduced by energy storage facilities and ramping constraints for conventional power plants [20]. Since DOPF is a large-scale non-linear and non-convex optimisation problem, solving it in parallel is of crucial importance.

For this kind of problems Primal Dual Interior Point Methods (*PDIPM*) have proven to be among the most powerful algorithms. The main computational effort, when PDIPM is applied, lies in solving linear systems of equations [14]. And this a task suitable to be carried out in parallel on multi-core CPUs.

There exist a few works on solving linear systems, that arise from PDIPM applied to DOPF, in parallel. One approach is based on parallel matrix factorization by means of Schur complement techniques [6, 20]. Other strategies use Benders Decomposition to decompose the complete optimisation problem into smaller ones [1].

Our contribution is the use of a *Schwarz Domain Decomposition Method* as preconditioner for Krylov-type iterative methods for solving these linear systems of equations in parallel. The original Schwarz method was formulated in 1870 as theoretical tool for proving existence of elliptic partial differential equations (*PDEs*) on complicated domains [17]. Later on, modifications of it have been used as stand-alone iterative methods for solving PDEs and have become a standard technique for preconditioning Krylov-type methods in the context of PDEs [18]. We apply this technique to DOPF by decomposing the time domain of interest into several smaller subdomains, which allows the use of multiple processors for computation.

2 Dynamic Optimal Power Flow

We will now formulate the DOPF problem in a mathematically rigorous way. This formulation has already been stated in more detail by several authors, e.g. in [12, 20] and [16]. We would like to mention that we consider a continuous formulation of the DOPF problem, i.e. one without any discrete decision variables. A non-linear mixed-integer formulation of an OPF is considered, for example, in [19].

2.1 *Formulation of the DOPF*

In order to formulate the DOPF problem for a given time period of interest $[0, T]$, we consider a uniform partition $0 = T_1 < T_2 < \ldots < T_{N_T} = T$ with constant step size $\tau = T_t - T_{t-1}$ for all $t \in \mathbf{T} \setminus \{1\}$, where $\mathbf{T} := \{1, \ldots, N_T\}$.

The power grid consisting of $N_\mathbf{B}$ nodes denoted by $\mathbf{B} := \{1, \ldots, N_\mathbf{B}\}$ and $N_\mathbf{E}$ transmission lines denoted by $\mathbf{E} \subset \mathbf{B} \times \mathbf{B}$, is described by an admittance matrix

$$\begin{aligned} &Y = G + jB \in \mathbb{C}^{N_\mathbf{B} \times N_\mathbf{B}} \\ &\text{with } G, B \in \mathbb{R}^{N_\mathbf{B} \times N_\mathbf{B}} \text{ and } j = \sqrt{-1}. \end{aligned} \tag{1}$$

The admittance matrix is symmetric, i.e. $Y = Y^T$, and furthermore $Y_{lk} = Y_{kl} \neq 0$ if and only if there is a branch connecting node k and l, i.e., $(k, l) \in \mathbf{E}$ and $(l, k) \in \mathbf{E}$. Therefore, Y is a sparse matrix for most real world power grids.

The complex voltage at node $k \in \mathbf{B}$ for time step $t \in \mathbf{T}$ is given as the complex number

$$V_k^t = E_k^t + jF_k^t \tag{2}$$

with real part $E_k^t \in \mathbb{R}$ and imaginary part $F_k^t \in \mathbb{R}$. We define $E^t := [E_k^t]_{k\in\mathbf{B}}$ and $F^t := [F_k^t]_{k\in\mathbf{B}}$.

Moreover, we define the following sets, which number:

$\mathbf{C} := \{1, \ldots, N_\mathbf{C}\}$	conventional power plants
$\mathbf{R} := \{1, \ldots, N_\mathbf{R}\}$	renewable energy sources
$\mathbf{S} := \{1, \ldots, N_\mathbf{S}\}$	storage facilities
$\mathbf{LS} := \{1, \ldots, N_\mathbf{LS}\}$	load shedding possibilities
$\mathbf{DC} := \{1, \ldots, N_\mathbf{DC}\}$	generators modelling DC lines
$\mathbf{V} := \{1, \ldots, N_\mathbf{V}\}$	generators modelling import/export

The resulting vector of variables corresponding to *active* power and energy for time step t is given by

$$P^t = \begin{pmatrix} [P^t_{g,i}]_{i\in\mathbf{C}} \\ [P^t_{r,i}]_{i\in\mathbf{R}} \\ [P^t_{sg,i}]_{i\in\mathbf{S}} \\ [P^t_{sl,i}]_{i\in\mathbf{S}} \\ [SOC^t_i]_{i\in\mathbf{S}} \\ [P^t_{l,i}]_{i\in\mathbf{LS}} \\ [P^t_{dc+,i}]_{i\in\mathbf{DC}} \\ [P^t_{dc-,i}]_{i\in\mathbf{DC}} \\ [P^t_{ex,i}]_{i\in\mathbf{V}} \\ [P^t_{im,i}]_{i\in\mathbf{V}} \end{pmatrix} \begin{matrix} \text{(injected conventional power)} \\ \text{(injected renewable power)} \\ \text{(injected storage power)} \\ \text{(extracted storage power)} \\ \text{(stored energy)} \\ \text{(load sheds)} \\ \text{(injected power by DC lines)} \\ \text{(extracted power by DC lines)} \\ \text{(power export)} \\ \text{(power import)} \end{matrix} \tag{3}$$

with $P^t \in \mathbb{R}^{N_P}$.

In a similar way, the vector of reactive power injections is defined by

$$Q^t = \begin{pmatrix} [Q^t_{g,i}]_{i\in\mathbf{C}} \\ [Q^t_{r,i}]_{i\in\mathbf{R}} \\ [Q^t_{sg,i}]_{i\in\mathbf{S}} \\ [Q^t_{sl,i}]_{i\in\mathbf{S}} \\ [Q^t_{l,i}]_{i\in\mathbf{LS}} \end{pmatrix} \in \mathbb{R}^{N_Q}. \tag{4}$$

Every variable related to active and reactive power can be assigned to a specific node of the power grid by means of the power injection and extraction matrices $C_P \in \{-1, 0, 1\}^{N_\mathbf{B}\times N_P}$ and $C_Q \in \{-1, 0, 1\}^{N_\mathbf{B}\times N_Q}$ [21, p.23]. Since the states of charge (SOC) of storage facilities do not directly influence the power flow equations, the corresponding rows in C_P are equal to the zero vector.

2.1.1 Constraints

At first, we consider the equality constraints. We begin with the power flow equations: Denoting the active and reactive power load for time step t by L^t_p and L^t_q, respectively, and assigning the loads to the nodes with the help of the matrix $C_L \in \{0, 1\}^{N_\mathbf{B}\times N_L}$, the AC power flow equations are given by [22]

$$C_P P^t - C_L L^t_p - Pf_p(E^t, F^t) = 0 \in \mathbb{R}^{N_\mathbf{B}}, \tag{5}$$

$$C_Q Q^t - C_L L^t_q - Pf_q(E^t, F^t) = 0 \in \mathbb{R}^{N_\mathbf{B}}, \tag{6}$$

where

$$Pf_{p,k}(E,F) = \sum_{l=1}^{N_{\mathbf{B}}} E_k(G_{kl}E_l - B_{kl}F_l) + F_k(G_{kl}F_l + B_{kl}E_l),$$

$$Pf_{q,k}(E,F) = \sum_{l=1}^{N_{\mathbf{B}}} F_k(G_{kl}E_l - B_{kl}F_l) - E_k(G_{kl}F_l + B_{kl}E_l),$$

which describe the sum of power flows at node k. Note that we dropped the index t to improve readability.

In the following, we abbreviate the equations (5) and (6) with the help of the following "symbolic" equation:

$$AC(E^t, F^t, P^t, Q^t) = 0. \tag{7}$$

Besides the power flow equations (7), one has to take into account an equality constraint for the slack bus, given by $F_1^t = 0$ for all t, and for generators $k, l \in \mathbf{DC}$ modelling a DC line $\overline{kl}$:

$$P^t_{dc+,k} = -\nu_{DC} P^t_{dc-,l} \text{ and } P^t_{dc+,l} = -\nu_{DC} P^t_{dc-,k} \,. \tag{8}$$

Here, line losses are taken into account by means of the factor $\nu_{DC} \in (0, 1)$.

Moreover, an additional set of equality constraints has to be imposed for modelling the state of charge, SOC_i^t, of storage facilities. These constraints are given by

$$SOC_i^t = SOC_i^{t-1} + \tau(\nu_l P^t_{sl,i} - \nu_g^{-1} P^t_{sg,i}), \quad \text{for all } i \in \mathbf{S} \tag{9}$$

with factors $\nu_l, \nu_g \in (0, 1)$ describing the efficiency of the loading process $P^t_{sl,i}$ and generation process $P^t_{sg,i}$, respectively.

Secondly, we take a look at the inequality constraints. Due to technical restrictions arising from power plants, renewable energy sources, storage facilities and DC transmission lines, it's necessary to impose lower and upper bounds for active and reactive power injections:

$$P^t_{min} \le P^t \le P^t_{max} \quad \text{and} \quad Q_{min} \le Q^t \le Q_{max} \,. \tag{10}$$

In our application the only bounds, which are time dependent, correspond to the power injected by renewable energy sources, i.e. P_r^t. Also the node voltages and the active power flow over transmission lines, denoted by p_{kl}, must be constrained:

$$U_{min}^2 \leq (E^t)^2 + (F^t)^2 \leq U_{max}^2 \tag{11}$$

$$p_{kl}(E_k^t, E_l^t, F_k^t, F_l^t) \leq Pf_{max,kl} \text{ for all } (k,l) \in \mathbf{E}\,. \tag{12}$$

Finally, we impose ramping constraints for conventional power plants to bound the rate of change for injected power between consecutive time steps by

$$|P_g^t - P_g^{t-1}| \leq \tau P_{max}^t \alpha \;\text{ for all } t \in \mathbf{T} \setminus \{1\}\,. \tag{13}$$

α is the ramping factor in percentage per unit time.

2.1.2 Cost Function

The cost function f accounts for expenditure and income related to active power processes for the complete time interval $[0, T]$ and is composed of contributions from each time step $t \in \mathbf{T}$ [5]:

$$f(P) = \sum_{t=1}^{N_T} \tau f_t(P^t) \tag{14}$$

with

$$\begin{aligned} f_t(P^t) = &\sum_{i\in\mathbf{C}} (a_{\mathbf{C},i,2}(P_{g,i}^t)^2 + a_{\mathbf{C},i,1} P_{g,i}^t) \\ &+ \sum_{i\in\mathbf{R}} (a_{\mathbf{R},i,2}(P_{r,i}^t)^2 + a_{\mathbf{R},i,1} P_{r,i}^t) \\ &+ \sum_{i\in\mathbf{V}} (a_{ex,i,2}(P_{ex,i}^t)^2 + a_{ex,i,1} P_{ex,i}^t) \\ &+ \sum_{i\in\mathbf{V}} (a_{im,i,2}(P_{im,i}^t)^2 + a_{im,i,1} P_{im,i}^t) \\ &+ \sum_{i\in\mathbf{LS}} (a_{\mathbf{LS},i,2}(P_{l,i}^t)^2 + a_{\mathbf{LS},i,1} P_{l,i}^t) \end{aligned} \tag{15}$$

and coefficients $a_{*,*,*} \in \mathbb{R}$.

2.1.3 Optimisation Problem

With the definitions of the previous sections at hand, the DOPF problem can be stated in an abstract way:

$$\begin{aligned}
\min_x \ & f(x) \text{ s.t.} \\
g_I^t(x^t) &= 0, && \text{for all } t \in \mathbf{T} \\
g_S^t(x^t, x^{t-1}) &= 0, && \text{for all } t \in \mathbf{T} \setminus \{1\} \\
h_I^t(x^t) &\leq 0, && \text{for all } t \in \mathbf{T} \\
h_R^t(x^t, x^{t-1}) &\leq 0, && \text{for all } t \in \mathbf{T} \setminus \{1\}
\end{aligned} \tag{16}$$

where the vector of optimisation variables is given by

$$x = [x^t]_{t \in \mathbf{T}} \in \mathbb{R}^{n^x} \text{ and } x^t = \begin{pmatrix} E^t & F^t & P^t & Q^t \end{pmatrix} \in \mathbb{R}^{n^{x,t}} . \tag{17}$$

Moreover, we used and will use the following definitions: $n^{x,t} := 2N_{\mathbf{B}} + N_P + N_Q$ and $n^x := N_T \, n^{x,t}$ and, as of now, let $n^{\lambda,t}$ be defined as the number of equality constraints corresponding to time step t, i.e. $\lambda^t \in \mathbb{R}^{n^{\lambda,t}}$ with $\lambda^t = (\lambda_I^t, \lambda_S^t)$ denoting the Lagrangian multipliers corresponding to g_I^t and g_S^t, respectively. Analogously, $\mu^t = (\mu_I^t, \mu_R^t) \in \mathbb{R}^{n^{\mu,t}}$ denotes the Lagrangian multipliers for the inequality constraints h_I^t and h_R^t. Consequently, we define $n^\lambda := N_T n^{\lambda,t}$ and $n^\mu := N_T n^{\mu,t}$.

Note that the ramping constraints h_R given by (13) and the storage constraints g_S given by (9) establish the only couplings between the variables corresponding to different time steps. Without them, a solution to (16) could be obtained by solving N_T independent OPF problems. The optimisation problem (16) is non-linear due to the AC equations and due to the voltage and line flow inequality constraints.[1] Moreover, the latter ones and the AC equations are the sources of non-convexity in (16).

To simplify the notation in the following sections, we abbreviate the optimisation problem (16) to

$$\min_x f(x) \quad \text{s.t.} \ \ g(x) = 0, \ \ h(x) \leq 0 \tag{18}$$

with quadratic functions

$$f\colon \mathbb{R}^{n^x} \to \mathbb{R}, \ g\colon \mathbb{R}^{n^x} \to \mathbb{R}^{n^\lambda}, \ h\colon \mathbb{R}^{n^x} \to \mathbb{R}^{n^\mu} . \tag{19}$$

[1]We have not explicitly stated an equation for the power flow p_{kl} through the transmission line (k, l). Please refer to [13].

3 Primal Dual Interior Point Method for DOPF

In this section, we briefly describe the application of the Primal Dual Interior Point Method (*PDIPM*) to a DOPF problem. The description is in line with the implementation found in the MATLAB package MATPOWER [21].

3.1 The PDIPM Algorithm

For the general optimisation problem (18), the corresponding Lagrangian function is

$$\begin{aligned} L\colon \mathbb{R}^{n^x} \times \mathbb{R}^{n^\lambda} \times \mathbb{R}^{n^\mu} &\to \mathbb{R} \\ (x, \lambda, \mu) &\mapsto f(x) + \lambda^T g(x) + \mu^T h(x). \end{aligned} \tag{20}$$

Assuming additional constraint qualifications, e.g., the linear independence constraint qualification (*LICQ*) [14], for every local minimum x^* of (18), there exist corresponding Lagrangian multipliers λ^*, μ^* such that (x^*, λ^*, μ^*) are in accord with the Karush-Kuhn-Tucker (*KKT*) conditions [5]:

$$\mathbf{F}_0(x, s, \lambda, \mu) := \begin{pmatrix} \nabla_x L(x, \lambda, \mu) \\ g(x) \\ h(x) + s \\ S\mu \end{pmatrix} = 0, \;\; s, \mu \geq 0, \tag{21}$$

where $s \in \mathbb{R}^{n^\mu}$ is a vector containing slack variables and $S \in \mathbb{R}^{n^\mu \times n^\mu}$ denotes a diagonal matrix whose diagonal elements are $S_{kk} = s_k$.

The PDIPM is an algorithm to find solutions to Eq. (21). Mathematically it solves a slightly modified version of this equation by applying to it Newton's method. The modification consists in adding a "term", which decreases with iteration index k. This term keeps the slack variables s away from zero and, thus, the inequality constraints inactive. This means that solutions $(x^{(k)}, s^{(k)}, \lambda^{(k)}, \mu^{(k)})$ to this modified version of Eq. (21) are inside the feasible region and *not* on its surface, hence, the method is called interior point method. More rigorously formulated, we find

$$\begin{aligned} \mathbf{F}_\gamma(x, s, \lambda, \mu) := \mathbf{F}_0(x, s, \lambda, \mu) - \begin{pmatrix} 0 \; 0 \; 0 \; \gamma e \end{pmatrix}^T &= 0, \\ s, \mu &\geq 0, \end{aligned} \tag{22}$$

for a sequence of *barrier parameters*

$$\gamma_k := \sigma \frac{(\mu^{(k-1)})^T s^{(k-1)}}{n^\mu}$$

where $\sigma = 0.1$ is the *centering parameter* and $(\mu^{(k-1)})^T s^{(k-1)}$ is called *complementary gap*. Additionally, $e = (1, \dots, 1) \in \mathbb{R}^{n^\mu}$. For details we refer the inclined reader to [5, 8] and [21].

Remark A very basic PDIPM is applied, i.e. there is no globalisation strategy. Since we focus on parallel linear algebra techniques for DOPF problems, a locally convergent algorithm is sufficient to our purpose.

3.2 KKT Matrix in PDIPM

The application of Newton's method to Eq. (22) yields iterations in which the following linear systems of equations must be solved:

$$\nabla \mathbf{F}_{\gamma_k}(x^{(k)}, s^{(k)}, \lambda^{(k)}, \mu^{(k)}) \Delta^{(k)} = -\mathbf{F}_{\gamma_k}(x^{(k)}, s^{(k)}, \lambda^{(k)}, \mu^{(k)}) . \tag{23}$$

Henceforth we omit the iteration index k. Assuming that $s, \mu > 0$, the Newton system (23) can be transformed into a reduced, symmetric saddle point form

$$\underbrace{\begin{pmatrix} \nabla^2_{xx} L + (\nabla h)^T \Sigma (\nabla h) & (\nabla g)^T \\ \nabla g & 0 \end{pmatrix}}_{=:\, A(x,s,\lambda,\mu)} \underbrace{\begin{pmatrix} \Delta x \\ \Delta \lambda \end{pmatrix}}_{=:\Delta_R} = -\underbrace{\begin{pmatrix} r_x \\ r_\lambda \end{pmatrix}}_{=:b} \tag{24}$$

with diagonal matrix $\Sigma = \operatorname{diag}\left(\frac{\mu_1}{s_1}, \dots, \frac{\mu_{n^\mu}}{s_{n^\mu}}\right)$ and

$$r_x = \nabla_x L + (\nabla h(x))^T Z^{-1} \left(\gamma e + \operatorname{diag}(\mu) h(x)\right) ,$$
$$r_\lambda = g(x) .$$

The KKT matrices A, arising in every PDIPM iteration k, may become more and more ill-conditioned when a KKT point is approached. This is caused by the sequence of scaling matrices $\Sigma^{(k)}$. If strict complementary holds at the KKT point x^*, if $x^{(k)} \to x^*$ and if $h_i(x^*)$ is an active inequality, one can show that $\Sigma^{(k)}_{ii} \to \infty$. On the other hand, $\Sigma^{(k)}_{ii} \to 0$, if $h_i(x^*)$ is inactive [14].[2] For this reason, many IPM software packages (like IPOPT [7]) use direct methods such as

2

LDL^T-factorizations to solve the appearing linear systems. These methods are less sensitive to ill-conditioned matrices.

In contrast, iterative linear solvers like GMRES [15] are very sensitive to ill-conditioned matrices. A good preconditioner is needed to lower the matrices' condition numbers. In exchange, they offer a higher potential of parallelization and allow the use of inexact Interior Point methods, see Sect. 3.3.

The distribution of the spectrum of a matrix K, defined as

$$\sigma(K) := \{\lambda \in \mathbb{C},\ \lambda \text{ is an eigenvalue of } K\},$$

is an indicator of the convergence behaviour of GMRES applied to $Kv = b$. By rule of thumb, a spectrum which is clustered away from 0 is beneficial to the rate of convergence of GMRES. [11, Th. 4.62]

For general non-linear optimisation problems with the corresponding KKT matrix

$$K = \begin{pmatrix} H & B^T \\ B & 0 \end{pmatrix},\ H \in \mathbb{R}^{n^x \times n^x},\ B \in \mathbb{R}^{n^\lambda \times n^x},\ n^\lambda \leq n^x, \tag{25}$$

the so-called *constraint preconditioner* is an example of a preconditioner. It is given by

$$\tilde{K} = \begin{pmatrix} \tilde{H} & B^T \\ B & 0 \end{pmatrix}, \tag{26}$$

with $\tilde{H}$ being an approximation to H, that is easy to factorize, e.g., $\tilde{H} = \text{diag}(H)$ [14].

One can show that $\sigma(\tilde{K}^{-1}K) = \{1\} \cup \tilde{\sigma}$ with $|\tilde{\sigma}| = n^x - n^\lambda$. Therefore, at most $n^x - n^\lambda$ eigenvalues of $\tilde{K}^{-1}K$ are not equal to 1. The distribution of these remaining eigenvalues depends on how well $\tilde{H}$ approximates H on the nullspace of B [10].[3]

In Sect. 4, we describe how to take advantage of the special structure given by (16) to construct a parallel preconditioner. Furthermore, we will see that there is a close relation between our proposed preconditioner and the constraint preconditioner defined above, see Sect. 4.4.

Proof *(sketch)* The KKT conditions require that $\mu_i^*(h_i(x^*) + s_i^*) = 0$. If $h_i(x^*)$ is active, then $h_i(x^*) = 0$ and $s_i^* = 0$. Upon strict complementarity follows that $\mu_i^* \neq 0$. Finally, if $x^{(k)} \to x^*$, then $\mu_i^{(k)} \to \mu_i^*$, $s_i^{(k)} \to s_i^*$ and $\Sigma_{ii}^{(k)} = \frac{\mu_i^{(k)}}{s_i^{(k)}} \to \infty$. □

[3]Let Z denote the nullspace of B, i.e. $BZ = 0$. Then $\tilde{H}$ approximates B on its nullspace, if $(Z^T \tilde{H} Z)^{-1}(Z^T H Z) \approx 1$.

3.3 Inexact PDIPM

In contrast to direct methods, iterative solvers allow solving linear systems with prescribed accuracy. One can exploit this fact by means of inexact Interior Point methods. Within these methods, the linear systems corresponding to the first PDIPM iterations are solved with a rather low accuracy and therefore a low number of iterations. As the PDIPM iterate approaches the exact solution, the accuracy of the linear solver is increased. We use the approach from [3] for determining the accuracy in each PDIPM iteration. Here, (23) has to be solved with residual $\tilde{r}^{(k)}$, i.e,

$$\begin{aligned}\nabla \mathbf{F}_{\gamma_k}(x^{(k)}, s^{(k)}, \lambda^{(k)}, \mu^{(k)})\Delta^{(k)} =& - \mathbf{F}_{\gamma_k}(x^{(k)}, s^{(k)}, \lambda^{(k)}, \mu^{(k)}) \\ &+ \tilde{r}^{(k)},\end{aligned}$$

that satisfies $\|\tilde{r}^{(k)}\|_2 \leq \tilde{\eta}_k \frac{(\mu^{(k)})^T s^{(k)}}{n^\mu}$ with forcing sequence $\tilde{\eta}_k \in (0, 1)$. With this choice, the update step $\Delta^{(k)}$ satisfies

$$\begin{aligned}\nabla \mathbf{F}_0(x^{(k)}, s^{(k)}, \lambda^{(k)}, \mu^{(k)})\Delta^{(k)} =& - \mathbf{F}_0(x^{(k)}, s^{(k)}, \lambda^{(k)}, \mu^{(k)}) \\ &+ r^{(k)}\end{aligned}$$

with $\|r^{(k)}\|_2 \leq (\gamma_k + \tilde{\eta}_k)\|\mathbf{F}_0(x^{(k)}, s^{(k)}, \lambda^{(k)}, \mu^{(k)})\|_2$. Therefore, "$\gamma_k + \tilde{\eta}_k$ can be viewed as forcing sequence of inexact Newton methods" [3]. In our numerical experiments, we set $\tilde{\eta}_k = 0.1$ for all k and computed the relative tolerance for the iterative solver as follows

$$\text{rtol}^{(k)} = 0.1 \frac{\mu^{(k)^T} s^{(k)}}{n^\mu \|\mathbf{F}_{\gamma_k}(x^{(k)}, s^{(k)}, \lambda^{(k)}, \mu^{(k)})\|}\,.$$

We stopped to decrease the relative tolerance when it reached a value smaller than 10^{-10}.

3.4 Termination Criteria for PDIPM

We use the following criteria for monitoring the progress of PDIPM [21]:

$$c_{feas}(x, s, \lambda, \mu) := \frac{\max\{\|g(x)\|_\infty, \max_i\{h_i(x)\}\}}{1 + \max\{\|x\|_\infty, \|s\|_\infty\}} \tag{27}$$

$$c_{grad}(x, s, \lambda, \mu) := \frac{\|\nabla_x L(x, \lambda, \mu)\|_\infty}{1 + \max\{\|\lambda\|_\infty, \|\mu\|_\infty\}} \tag{28}$$

$$c_{comp}(x, s, \lambda, \mu) := \frac{(\mu^T s)}{1 + \|x\|_\infty} \tag{29}$$

$$c_{cost}(x, s, \lambda, \mu) := \frac{|f(x) - f(x^-)|}{1 + |f(x^-)|} \tag{30}$$

with x^- denoting the previous iterate.

4 Schwarz Domain Decomposition Method for DOPF

The key idea behind Schwarz domain decomposition methods applied to DOPF problems is the decomposition of the set of time steps $\mathbf{T}$ into several smaller subsets. Due to this decomposition, it is possible to define corresponding submatrices of the global KKT matrix defined in (24) that are smaller in size and therefore faster to factorize. Furthermore, the factorization of these submatrices can be done in parallel. After computing subsolutions corresponding to these submatrices, one can reconstruct an approximate solution to the global system (24), which is used as preconditioner within a Krylov-type iterative solver.

In Sect. 4.1 we introduce the mathematics, which describe the additive Schwarz Method (*ASM*) in the context of DOPF problems. In Sect. 4.2 we briefly describe some issues related to its implementation. Section 4.3 is intended to provide a deeper insight into the subproblems that have to be solved when the ASM is applied. Finally, we describe the relation between the ASM and the above mentioned constraint preconditioner in Sect. 4.4.

4.1 Mathematical Formulation of the Additive Schwarz Method

For the application of the ASM as a preconditioner for the KKT matrix $A(x, s, \lambda, \mu) \in \mathbb{R}^{(n^x+n^\lambda)\times(n^x+n^\lambda)}$ defined by (24), it is needed to decompose the set of time steps $\mathbf{T} = \{1, \ldots, N_T\}$ into q non-overlapping subdomains:

$$\mathbf{T} = \bigcup_{l=1}^{q} \tilde{\mathbf{T}}_l,$$
$$\text{with } \tilde{\mathbf{T}}_l \cap \tilde{\mathbf{T}}_k = \emptyset \text{ for } k \neq l \text{ and } \tilde{\mathbf{T}}_l := \{\tilde{t}_l^-, \tilde{t}_l^- + 1, \ldots, \tilde{t}_l^+\}\,.$$

Afterward, each subdomain $\tilde{\mathbf{T}}_l$ is expanded by additional s_{ol} time steps on both ends, yielding an overlapping decomposition of $\mathbf{T}$ (see Fig. 1):

$$\mathbf{T} = \bigcup_{l=1}^{q} \mathbf{T}_l,\ \ \mathbf{T}_l := \{t_l^-, t_l^- + 1, \ldots, t_l^+\},$$

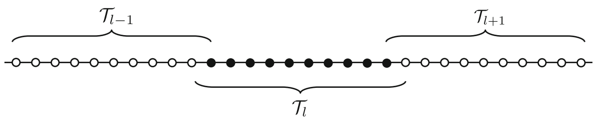

Fig. 1 Decomposition of time steps with overlap $s_{ol} = 1$

with

$$t_l^- = \begin{cases} \tilde{t}_l^- - s_{ol}, & l > 1 \\ \tilde{t}_l^-, & l = 1 \end{cases}, \quad t_l^+ = \begin{cases} \tilde{t}_l^+ + s_{ol}, & l < q \\ \tilde{t}_l^+, & l = q \end{cases}$$

Typically, $s_{ol} \in \{1, 2, 3\}$.

Furthermore, we use the definitions of Sect. 2.1.3 to introduce

$$\begin{aligned} n_l^x &:= \sum_{t \in \mathbf{T}_l} n^{x,t}, \\ n_l^\lambda &:= \sum_{t \in \mathbf{T}_l} n^{\lambda,t} \\ n_l &:= n_l^x + n_l^\lambda, \\ n &:= n^x + n^\lambda . \end{aligned}$$

Henceforward, all expressions involving $\tilde{}$ refer to the non-overlapping subdomains $\tilde{\mathbf{T}}_l$.

When restricting the optimisation variables to the components, which belong to $\mathbf{T}_l$, we write

$$x_{[l]} = [x^t]_{t \in \mathbf{T}_l} \text{ and} \tag{31}$$

$$\lambda_{[l]} = \left[\lambda^t\right]_{t \in \mathbf{T}_l} = \left[\begin{pmatrix} \lambda_I^t \\ \lambda_S^t \end{pmatrix}\right]_{t \in \mathbf{T}_l} . \tag{32}$$

The combination of $x_{[l]}$ and $\lambda_{[l]}$ yields the following definition:

$$\Delta_{R[l]} := \begin{pmatrix} \Delta x \\ \Delta \lambda \end{pmatrix}_{[l]} := \begin{pmatrix} \Delta x_{[l]} \\ \Delta \lambda_{[l]} \end{pmatrix} \in \mathbb{R}^{n_l} . \tag{33}$$

This vector contains the components of the solution vector of (24) belonging to the time steps in $\mathbf{T}_l$. Moreover, let $R_l \in \{0, 1\}^{n_l \times n}$ be a *restriction matrix* corresponding to the subset $\mathbf{T}_l$ defined by its action:

$$R_l \begin{pmatrix} \Delta x \\ \Delta\lambda \end{pmatrix} = \begin{pmatrix} \Delta x \\ \Delta\lambda \end{pmatrix}_{[l]}, \text{ where } R_l = \begin{pmatrix} R_l^x & 0 \\ 0 & R_l^\lambda \end{pmatrix}. \tag{34}$$

The matrices R_l^x and R_l^λ are composed of either zero or identity blocks. As an example, consider the case of $l \neq 1$ and $l \neq q$:

$$\begin{aligned} R_l^x &= \begin{pmatrix} 0 & I_{n_l^x} & 0 \end{pmatrix} \in \{0,1\}^{n_l^x \times n^x}, \\ R_l^\lambda &= \begin{pmatrix} 0 & I_{n_l^\lambda} & 0 \end{pmatrix} \in \{0,1\}^{n_l^\lambda \times n^\lambda}. \end{aligned} \tag{35}$$

Note that the size of the zero matrices varies with each restriction matrix and each l. Analogously, we define $\tilde{R}_l$ as restriction operator corresponding to the non-overlapping subdomain $\tilde{\mathbf{T}}_l$.

With these definitions at hand, it is possible to extract submatrices A_l from the KKT matrix A by multiplying it with R_l:

$$A_l := R_l A R_l^T. \tag{36}$$

In Sect. 4.3, we explore the structure of these submatrices.

On the important assumption, that all submatrices A_l are *non-singular*, we are able to formulate the multiplicative Schwarz Method (*MSM*) for approximately solving $A\Delta_R = b$. We do this in the form of an algorithm [18]:

Algorithm 1 Multiplicative Schwarz method

```
Set $\Delta_R^0 = 0$
for $l = 1, \dots, q$ do
    $\Delta_R^l = \Delta_R^{l-1} + R_l^T A_l^{-1} R_l \left(b - A\Delta_R^{l-1}\right) = \Delta_R^{l-1} + \delta_l$
end for
Return $\Delta_R^q$
```

In every iteration l, the current error with respect to the exact solution Δ_R, given by

$$e^{l-1} = \Delta_R - \Delta_R^{l-1},$$

satisfies

$$Ae^{l-1} = r^{l-1} \text{ with residual vector } r^{l-1} = b - A\Delta_R^{l-1}.$$

By restricting r^{l-1} to subdomain $\mathbf{T}_l$ and solving with A_l^{-1}, one obtains an approximation

$$\hat{e}^l := A_l^{-1} R_l r^{l-1}$$

to the exact error $R_l e^{l-1}$ on subdomain $\mathbf{T}_l$. Prolongation with R_l^T yields a correction $\delta_l = R_l^T \hat{e}^l$ to the current approximation Δ_R^{l-1}. Thus, the multiplicative Schwarz method can be seen as “defect correction algorithm”.[4]

Unfortunately, the the MSM is a sequential algorithm. In order to parallelize it, one omits residual updates in each iteration, yielding the ASM [18]:

Algorithm 2 Additive Schwarz method $\Delta_R^q = \mathrm{ASM}(b)$

```
Set $\Delta_R^0 = 0$
for $l = 1, \dots, q$ do
    $\Delta_R^l = \Delta_R^{l-1} + R_l^T A_l^{-1} R_l b$
end for
Return $\Delta_R^q$
```

which can be written as

$$\Delta_R^q = M_{ASM} b \ \text{ with } M_{ASM} := \sum_{l=1}^{q} R_l^T A_l^{-1} R_l \,.$$

The right preconditioned linear system for solving $A\Delta_R = b$ is then given by

$$A M_{ASM} u = b, \ \Delta_R = M_{ASM} u \,. \tag{37}$$

Since M_{ASM} is symmetric but in general not positive definite, we use the Generalized Minimal Residual (*GMRES*) method for solving the non-symmetric system (37). In general, the ASM computes a less accurate approximation to the solution of $A\Delta_R = b$ and therefore needs an increased number of GMRES iterations compared to the MSM given by Algorithm 1. However, the ASM offers a higher potential of parallelization, which results usually in better parallel scaling.

[4]In contrast to “classical defect correction” algorithms, there is no converging sequence $\left(\Delta_R^l\right)_{l\in\mathbb{N}} \to \Delta^R$. Nonetheless, the algorithm’s construction implies that $\Delta_R^q \approx \Delta^R$.

4.2 Implementation

In our implementation, we use one processor per subdomain $\mathbf{T}_l$ and distribute A such that every processor stores $\tilde{R}_l A$ in its local memory. $\tilde{R}_l A$ contains the non-overlapping portion of rows of $A_l = R_l A R_l^T$.

To set up M_{ASM} once, every processor first has to form its local submatrix A_l, i.e., in this step the overlapping part of A_l has to be communicated to processor l by processor $l-1$ and $l+1$. Afterward, each processor computes an LU-factorization of A_l, i.e., $A_l = L_l U_l$. This step doesn't involve any inter-processor communication and can be done in parallel.

The employment of the ASM as a preconditioner inside the GMRES method, requires to compute $M_{ASM} v^{(k)}$ in each iteration k. $v^{(k)}$ denotes the k-th basis vector of the Krylov subspace $\mathscr{K}_k(AM_{ASM}, b)$. On the assumption that the basis vector's data layout is the same as the matrix's one, i.e. every process stores $v_l^{(k)} := \tilde{R}_l v^{(k)}$, communication with process $l-1$ and $l+1$ is required to multiply M_{ASM} with $v^{(k)}$.[5] The computation of $A_l^{-1} v_l^{(k)}$ is done by one forward substitution with L_l and a backward substitution with U, respectively. This step does not involve any communication. After this, the local solution $A_l^{-1} v_l^{(k)}$ is prolonged back to obtain the global result of the matrix vector product $M_{ASM} v^{(k)}$. This again requires communicating the overlapping part of $v_l^{(k)}$ to process $l-1$ and $l+1$. Finally, it's necessary to multiply A with $M_{ASM} v^{(k)}$. Due to A's data layout and its off-diagonal elements, corresponding to intertemporal couplings, additional communication is required.

One can further improve the performance of ASM by using the so-called *restricted* version of ASM [4], which is given by

$$M_{rASM} := \sum_{l=1}^{p} \tilde{R}_l^T A_l^{-1} R_l \, . \tag{38}$$

For this preconditioner, just the non-overlapping part of the local solution $v_l^{(k)}$ is prolonged instead of the entire (overlapping) vector. Experiments show a beneficial behaviour in terms of GMRES iterations compared to standard ASM [4]. Furthermore, prolongation by $\tilde{R}_l$ doesn't involve any communication.

To further stabilise the LU-factorization, a small regularisation parameter $\epsilon = 10^{-8}$ is added to the KKT matrix A, i.e. $A + \epsilon I$.

[5]The reader may wonder why we have to make an assumption about the basis vectors' data layout: Our GMRES implementation is part of the external library PETSc and a quick glance at their source code did not reveal to us how they manage the according data.

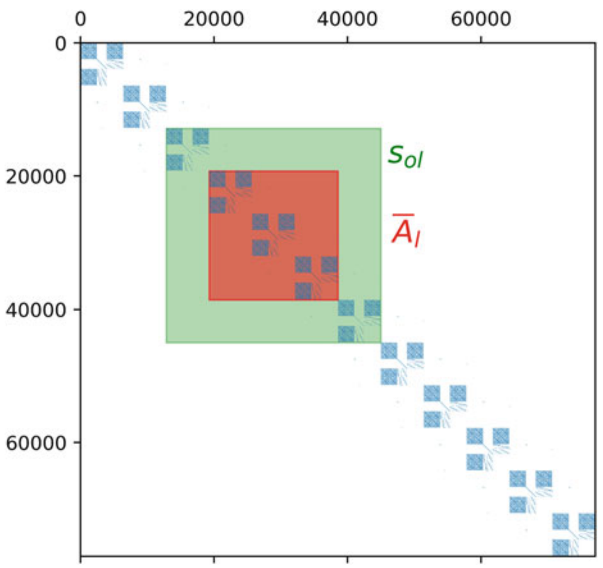

Fig. 2 Permuted KKT matrix A with $N_T = 12$, $s_{ol} = 1$, $q = 4$ and $\bar{A}_l = P_l A_l {P_l}^T$

4.3 Structure of Local KKT Matrices

In this section, we investigate the structure of the local submatrices A_l defined in Eq. (36). A good way to get an idea of their structure is to permute the KKT matrix A in such a way that all rows and columns belonging to *one* time step are grouped together. This yields a block diagonal matrix with some off-diagonal elements. Every block represents one time step and the off-diagonal elements arise due to the time-dependent constraints. Extracting the time steps belonging to the subdomain $\mathbf{T}_l$ results in a permuted version of the submatrix A_l.[6] Its structure is exactly the same as the one of the original KKT matrix A. If the off-diagonal elements are taken into account as “boundary conditions”, it is possible to interpret the submatrix A_l as representing an optimisation problem in its own, i.e. the original dynamical OPF problem is reduced to the subdomain $\mathbf{T}_l$. Figure 2 tries to demonstrate this. As a consequence the submatrices A_l tend to be ill-conditioned.

[6]Note that this extraction cannot be done by multiplying A with R_l and ${R_l}^T$.

4.4 Relationship Between the ASM and the Constraint Preconditioner

To see a relation between the additive Schwarz method and the constraint preconditioner, as introduced in Sect. 3.2, it is necessary to consider a dynamical OPF problem without time-dependent equality constraints. In our case, we had to drop the storage facilities g_S^t to do this. Furthermore, it is important to consider a non-overlapping decomposition of the time domain, i.e. $s_{ol} = 0$. In this case, the corresponding ASM preconditioner,

$$\tilde{M}_{ASM} = \sum_{l=1}^{q} \tilde{R}_l^T \tilde{A}_l^{-1} \tilde{R}_l, \quad (\text{where } \tilde{A}_l := \tilde{R}_l A \tilde{R}_l^T \in \mathbb{R}^{\tilde{n}_l \times \tilde{n}_l}) \tag{39}$$

reduces to a block Jacobi method with omitted couplings between variables belonging to $\tilde{t}_l^+$ and $\tilde{t}_{l+1}^-$ for each subdomain l. Since there are only time-dependent inequality constraints left, the coupling between the variables belonging to different time steps has to arise in the $(1, 1)$-block of the KKT matrix, namely in the $(\nabla h)^T \Sigma (\nabla h)$ part. This means that multiplying $\tilde{M}_{ASM}$ with b is like solving a modified KKT system, or equivalently,

$$\tilde{M}_{ASM} = \begin{pmatrix} \tilde{H} & (\nabla g)^T \\ \nabla g & 0 \end{pmatrix}^{-1} . \tag{40}$$

$\tilde{H}$ denotes the KKT matrix's modified $(1, 1)$-block. Thus, $\tilde{M}_{ASM}$ has the form of a constraint preconditioner.

At this point, it also becomes clear that the ASM can only be interpreted as a constraint preconditioner, if there are no time-dependent equality constraints. If there existed time-dependent equality constraints, the $(2, 1)$- and $(1, 2)$-block of $\tilde{M}_{ASM}^{-1}$ would have changed too.

As pointed out in Sect. 3.2, 1 is an eigenvalue of $\tilde{M}_{ASM}A$ of multiplicity $2n^\lambda$ and the remaining $n^x - n^\lambda$ eigenvalues are solutions of a generalized eigenvalue problem of the following form [10]:

$$Z^T \underbrace{(\nabla_{xx}^2 \mathbf{L} + (\nabla h)^T \Sigma (\nabla h))}_{=:H} Z v = \lambda Z^T \tilde{H} Z v\,. \tag{41}$$

For a low number of subdomains q, one can expect $\tilde{H}$ to be a good approximation to H, leading to a clustered spectrum $\sigma(\tilde{M}_{ASM}A)$ close to one. However, the more subdomains, the more neglected couplings and the less accurate does $\tilde{H}$ approximate H. This results in a more scattered eigenvalue distribution of $\tilde{M}_{ASM}A$, but still a large number of eigenvalues are equal to 1.

5 Numerical Experiments

In this section, we present some results for our proposed method applied to a DOPF problem. In its first part, we discuss the parallel speedup of our solver. In the second one, we investigate the way in which the KKT matrix's spectrum $\sigma(A)$ changes with the iterations of the interior point method. The test grid, which we used for our experiments, represents a possible variant of the German Transmission Grid in the year 2023 and consists of 1215 nodes, 2247 AC lines, 8 DC lines, 181 conventional power plants, 355 renewable energy sources and 67 storage facilities. For more details, we refer the reader to [12].

For solving (16), we use the PDIPM algorithm mips which is written in MATLAB code and part of MATPOWER [21]. In this algorithm, we replace the standard MATLAB backslash operator \ for solving linear systems by our own linear solver. Our solver applies a right preconditioned GMRES method. We use the ASM as the preconditioner. For computing the LU-factorizations of the local systems A_l, we use the software package Intel® MKL PARDISO. Our solver is written in C++ and makes use of the KSPGMRES and PCASM methods provided by PETSc [2], which is compiled in Release mode with the Intel compiler 17.0. The computation of the eigenvalues of the KKT matrices and the preconditioned KKT matrices have been done with SLEPc [9] and MATLAB. Inter-process communication is realised by means of the Message Passing Interface (MPI). The numerical experiments were carried out on the BwForCluster MLS&WISO Production at Heidelberg University with two Intel Xeon E5-2630v3 processors i.e. 16 CPU cores at 2.4 GHz per node, and 64 GB working memory.

5.1 Parallel Speedup

In our numerical experiments, we concatenated the 48 h load profile, which was part of the described test grid, to obtain a test case comprising 48 days with a temporal resolution of 1 h, i.e. $N_T = 960$ time steps. This causes the linear system (24), which has to be solved in every PDIPM iteration, to have the dimension $n^x + n^\lambda \approx 6.168 \cdot 10^6$.

Furthermore, we set $\alpha = 0.4$, which led to ramping constraints limiting the change of power generation of conventional power plants by 40%/h of their respective maximum power generation.

We set $\epsilon_{feas} = \epsilon_{grad} = \epsilon_{cost} = \epsilon_{comp} = 10^{-5}$ as PDIPM termination criteria and limited the number of PDIPM iterations to 300. The number of GMRES iterations was restricted to 1500. The GMRES method was allowed to restart after 300 iterations. Its relative residual tolerance rtol was determined as described in Sect. 3.3.

The ASM method was not restricted and the overlap s_{ol} was set to 3 for all tests.

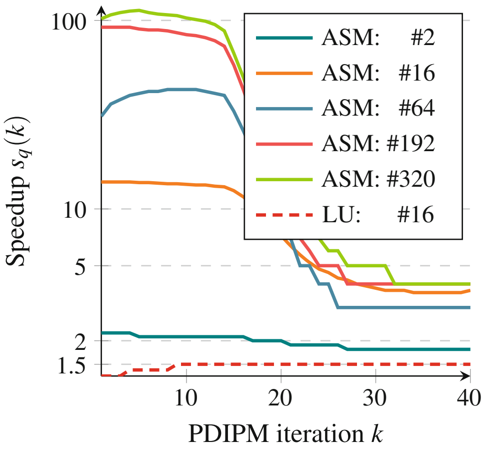

Fig. 3 Speedup up to the iteration on the x-axis for different numbers of processors

The parallel speedup on q processors for the first k PDIPM iterations is defined as $s_q(k) := \frac{T_1(k)}{T_q(k)}$. We computed the sequential reference time $T_1(k)$ by solving the linear systems (24) using a LU-factorization and adding up the times needed to solve them up to the k-th PDIPM iteration. MKL PARDISO's sequential LU-factorization algorithm was applied.

Analogously, $T_q(k)$ was obtained by adding up the times needed to solve the same linear systems, but this time iteratively, i.e. the GMRES method and the ASM were used. In Fig. 3 we plotted the results of our speedup computations for different numbers of processors. The dashed graph represents the speedup obtained when the KKT systems (24) are solved directly with MKL PARDISO's parallel implementation of the LU-factorization using 16 processors. For this amount of processors we observed the best speedup.

Besides, we scaled the y-axis logarithmically to ease drawing a comparison with the LU-solver and to emphasise the large speedups during the first iterations.[7] We observed for our proposed iterative solver, no matter the number of processors q, a clear speedup in comparison with the LU-solver. The parallelization, due to the ASM, shows its power during the first PDIPM iterations. Unfortunately, this initial speedup decreases rapidly. After 40 iterations we end up with a total speedup of five. This decrease is accompanied by an increase in GMRES iterations. To demonstrate this, we plotted in Fig. 4 the number of GMRES iterations needed to solve the KKT system at the k-th PDIPM iteration. The first thing to note is, that the number of GMRES iterations does not only change with PDIPM iterations, but also with the number of processors used to solve the KKT systems. It's likely that the increase in GMRES iterations with PDIPM iterations is caused by a change in the spectrum of the original KKT matrix A. The increase with the number of

[7]We remark that the PDIPM needed in most of the tested cases more than 40 iterations to converge, yet, it converged after at most 45 iterations.

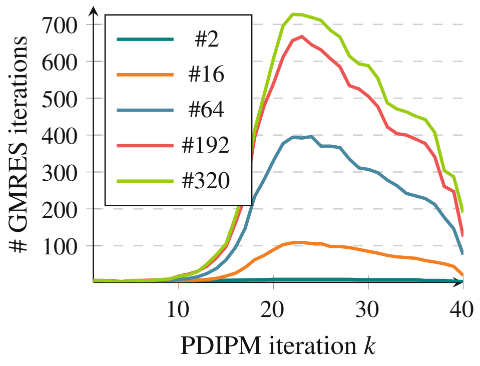

Fig. 4 The number of GMRES iterations needed to solve the KKT system corresponding to the PDIPM iteration on the x-axis

processors can be explained as follows: The more proccesors we use, the smaller the subproblems, described by A_l, become and, hence, the more intertemporal couplings are neglected, which reduces the "approximity" of M_{ASM} to A^{-1}. And the closer M_{ASM} is to A^{-1}, the more the spectrum of the preconditioned operator $\sigma(AM_{ASM})$ is clustered around 1.

Before we start to actual investigate the spectrum of the matrices in the next part of this section, we would like to summarise our findings. Even though we observed a rapid decrease in the speedup, it was possible to obtain a moderate speedup with only 16 processors and in comparison to a standard parallel LU-factorization also using 16 processors, our solver was more than a factor 2 faster.

5.2 *Eigenvalue Distribution*

Since the GMRES method converges faster if the eigenvalues of matrix A are clustered around 1, i.e. it needs less iterations, we would like to investigate how much the ASM, applied as preconditioner, improves the eigenvalues' distribution. Therefore, we calculated the eigenvalues with the largest and smallest magnitude of the KKT matrix A and the preconditioned KKT matrix AM_{ASM}, respectively. We write $\lambda_{max}^A = \max\{|\lambda| : \lambda \in \sigma(A)\}$ and $\lambda_{min}^A = \min\{|\lambda| : \lambda \in \sigma(A)\}$. Seeing that it takes very long to compute, or that it is even not possible to compute the eigenvalues for the complete $N_T = 960$ time steps, we decided to calculate them for $N_T = 96$ time steps and for each PDIPM iteration. We set the overlap to $s_{ol} = 1$ and used $q = 32$ subdomains. The results are presented in Fig. 5. The red lines depict the development of λ_{max}^A and λ_{min}^A of the KKT matrix, respectively. Thus, they form the limits for all possible magnitudes of eigenvalues corresponding to A. The latter is indicated by the red shaded area. Analogously, the blue lines and the blue area represent the spectrum of the preconditioned KKT matrix.

The spectrum of the preconditioned KKT matrix, is clustered around 1 in every interior point method iteration. Looking at the magnified part of the graph shows

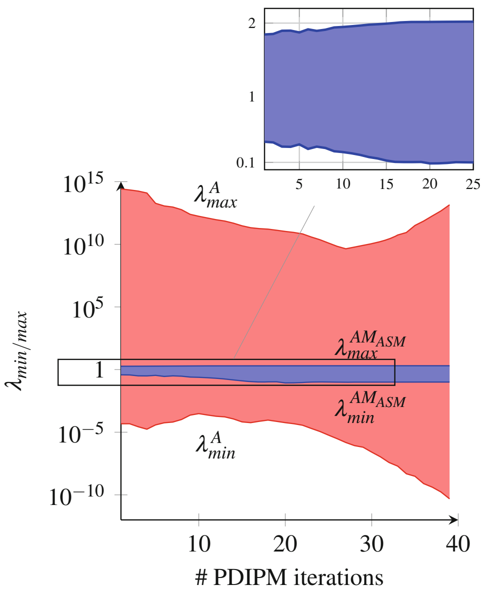

Fig. 5 Variation of the spectrum of the original KKT matrix and the preconditioned one

that $\lambda^{AM_{ASM}}_{min}$ becomes smaller between the 10-th and the 20-th iteration. This may explain the increase in GMRES iterations observed in our numerical experiments and plotted in Fig. 4.

6 Conclusion

In this work we proposed a way of solving linear systems arising from Dynamic Optimal Power Flow (DOPF) problems in parallel by means of an overlapping Schwarz domain decomposition method, namely the additive Schwarz method. It was shown how to apply this method in the context of DOPF problems and that the obtained submatrices correspond to localised formulations of a DOPF problem with additional boundary conditions. Numerical tests on one large-scale DOPF problem showed that the combination of the Generalized Minimal Residual (GMRES) method and the Additive Schwarz Method (ASM) is able to solve DOPF problems. Furthermore, we were able to show that this combination yields good parallel speedup for some of the PDIPM iterations. And, in a small example, we demonstrated that the ASM is a good preconditioner in the sense, that it clusters the eigenvalues around 1.

Unfortunately, the proposed solver had problems with some of the KKT systems arising in the large-scale DOPF problem. Therefore, it is necessary to improve it and this will be one of our main goals in future work.

Acknowledgements This work was carried out with the financial support of the German Research Foundation (DFG) within the project 242471205. Moreover, the authors acknowledge support by the state of Baden-Württemberg through bwHPC and the German Research Foundation (DFG) through grant INST 35/1134-1 FUGG.

References

1. N. Alguacil, A.J. Conejo, Multiperiod optimal power flow using benders decomposition. IEEE Trans. Power Syst. **15**(1), 196–201 (2000)
2. S. Balay, W.D. Gropp, L. Curfman McInnes, B.F. Smith, Efficient management of parallelism in object oriented numerical software libraries, in *Modern software tools in scientific computing*, ed. by E. Arge, A.M. Bruaset, H.P. Langtangen (Birkhäuser Press, Boston 1997), pp. 163–202
3. S. Bellavia, Inexact interior-point method. J. Optim. Theory Appl. **96**(1), 109–121 (1998)
4. X.-C. Cai, M. Sarkis, A restricted additive Schwarz preconditioner for general sparse linear systems. SIAM J. Sci. Comput. **21**(2), 792–797 (1999)
5. F. Capitanescu, M. Glavic, D. Ernst, L. Wehenkel, Interior-point based algorithms for the solution of optimal power flow problems. Electr. Power Syst. Res. **77**(5–6), 508–517 (2007)
6. C.Y. Chung, W. Yan, F. Liu, Decomposed predictor-corrector interior point method for dynamic optimal power flow. IEEE Trans. Power Syst. **26**(3), 1030–1039 (2011)
7. F.E. Curtis, J. Huber, O. Schenk, A. Wächter, A note on the implementation of an interior-point algorithm for nonlinear optimization with inexact step computations. Math. Program. **136**(1), 209–227 (2012)
8. P. Gerstner, V. Heuveline, M. Schick, A multilevel domain decomposition approach for solving time constrained optimal power flow problems. Preprint Ser. Eng. Math. Comput. Lab (EMCL) **4**, 1–30 (2015)
9. V. Hernandez, J.E. Roman, V. Vidal, SLEPc: a scalable and flexible toolkit for the solution of eigenvalue problems. ACM Trans. Math. Softw. **31**(3), 351–362 (2005)
10. C. Keller, N.I.M. Gould, A.J. Wathen, Constraint preconditioning for indefinite linear systems. SIAM J. Matrix Anal. Appl. **21**(4), 1300–1317 (2000)
11. A. Meister, *Numerik linearer Gleichungssysteme. Eine Einführung in moderne Verfahren. Mit MATLAB-Implementierungen von C. Vömel*, 5th revised edn. (Springer Spektrum, Heidelberg, 2015)
12. N. Meyer-Hübner, M. Suriyah, T. Leibfried, V. Slednev, V. Bertsch, W. Fichtner, P. Gerstner, M. Schick, V. Heuveline, Time constrained optimal power flow calculations on the German power grid, in *International ETG Congress 2015; Die Energiewende – Blueprints for the new energy age*, Bonn, 2015, pp. 1–7
13. N. Meyer-Hübner, F. Gielnik, M. Suriyah, T. Leibfried, Dynamic optimal power flow in AC networks with multi-terminal HVDC and energy storage, in *2016 IEEE Innovative Smart Grid Technologies – Asia (ISGT-Asia)*, Nov 2016
14. J. Nocedal, S.J. Wright, *Numerical optimization*. Springer Series in Operations Research and Financial Engineering (Springer, Berlin, 2006)
15. Y. Saad, M.H. Schultz, GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. SIAM J. Sci. Stat. Comput. **7**(3), 856–869 (1986)
16. M. Schönfelder, A. Eßer-Frey, M. Schick, W. Fichtner, V. Heuveline, T. Leibfried, New developments in modeling network constraints in techno-economic energy system expansion planning models. Zeitschrift für Energiewirtschaft **36**(1), 27–35 (2012)
17. H.A. Schwarz, *Ueber einen Grenzübergang durch alternirendes Verfahren*. Vierteljahrsschrift der Naturforschenden Gesellschaft in Zürich. Zürcher u. Furrer, 1870
18. B.F. Smith, P.E. Bjørstad, W.D. Gropp, *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations* (Cambridge University Press, Cambridge, 1996)

19. E.M. Soler, V.A. de Sousa, G.R.M. da Costa, A modified primal-dual logarithmic-barrier method for solving the optimal power flow problem with discrete and continuous control variables. Eur. J. Oper. Res. **222**(3), 616–622 (2012)
20. K. Xie, Y.H. Song, Dynamic optimal power flow by interior point methods. IEE Proc. Gener. Transm. Distrib. **148**(1), 76–84 (2001)
21. R.D. Zimmerman, C.E. Murillo-Sánchez, R.J. Thomas, Matpower: steady-state operations, planning, and analysis tools for power systems research and education. IEEE Trans. Power Syst. **26**(1), 12–19 (2011)
22. J. Zhu, *Optimization of Power System Operation*, vol. 47 (John Wiley & Sons, Hoboken, 2015)