A Multilevel Domain Decomposition approach for solving time constrained Optimal Power Flow problems

Philipp Gerstner, Vincent Heuveline, Michael Schick

Preprint No. 2015-04

$\frac{\partial u}{\partial t} - \nu\Delta u$
$+ (u \cdot \nabla) u + \frac{1}{\rho}\nabla p$
$\frac{\partial u}{\partial t}$

Preprint Series of the Engineering Mathematics and Computing Lab (EMCL)
ISSN 2191–0693
Preprint No. 2015-04

Affiliation of the Authors

Philipp Gerstner[a,b,1], Vincent Heuveline[a,b], Michael Schick[a,b]

[a] *Engineering Mathematics and Computing Lab (EMCL), Interdisciplinary Center for Scientific Computing (IWR), Heidelberg University, Germany*
[b] *Data Mining and Uncertainty Quantification Group (DMQ), Heidelberg Institute for Theoretical Studies (HITS), Germany*
[1] *Corresponding Author: Philipp Gerstner, philipp.gerstner@uni-heidelberg.de*

Impressum

Heidelberg University
Interdisciplinary Center for Scientific Computing (IWR)
Engineering Mathematics and Computing Lab (EMCL)

Speyerer Str. 6,
69115 Heidelberg
Germany

A Multilevel Domain Decomposition approach for solving time constrained Optimal Power Flow problems

Philipp Gerstner, Vincent Heuveline, Michael Schick

September 14, 2015

Abstract

Solving Time Constrained Optimal Power Flow problems (TCOPF) is a major task for determining optimal extensions of a given power grid. When employing any gradient based optimization algorithm such as Interior Point Method or Sequential Quadratic Programming for TCOPF, the main computational effort lies in the solution of large and coupled linear systems. Even for medium-sized electrical networks and time periods in the range of a few days, these systems can contain several millions of equations. The corresponding matrix is block tri-diagonal with non-diagonal blocks corresponding to intertemporal couplings. In our work, we exploit this fact by using Schwarz preconditioning techniques in combination with iterative Krylov subspace methods such as GMRES for solving linear systems in parallel. We propose a way of applying these domain decomposition methods in context of TCOPF problems and present numerical experiments that illustrate their behaviour on two benchmark problems.

1 Introduction

In many countries the energy sector continues to undergo substantial changes. For instance, the expansion of decentralized renewable energy sources requires to extend existing transmission grids which have been designed for centralized and controllable power production in conventional power plants. Determining such an optimal grid extension leads to a large-scale mixed-integer optimization problem and includes finding optimal operating states for given power grids over a given time period.

The problem of finding an optimal operating state over a specific time period is known as *dynamic Optimal Power Flow* or *time constrained Optimal Power Flow* (TCOPF) [20]. Since TCOPF itself is a large-scale nonlinear optimization problem, solving it on a parallel computer architecture is of crucial importance.

In this context, *Interior Point Methods* (IPM) have proven to be one of the most powerful optimization algorithms, because their number of iterations to obtain convergence is rather insensitive to the problem size. Moreover, the main computational effort when applying IPM lies in the solution of linear systems of equations which is a suitable task to be carried out on many parallel processors.

There exist a few works on parallel solution of linear systems that arise from IPM applied to TCOPF. One such approach is based on parallel LU-factorizations by means of Schur complement techniques [8], [20]. Other strategies use Benders Decomposition to decompose the complete optimization problem into smaller ones [1].

In this work, we propose the use of *Schwarz Domain Decomposition Methods* as a preconditioner for Krylov-type iterative methods for solving linear systems in parallel. We apply these techniques by decomposing the time period into several smaller sub-domains, allowing the use of multiple computing processors. All modifications are done on an algebraic level and do not disturb the convergence behaviour of the underlying IPM algorithm. Furthermore, iterative linear solvers allow the use of *Inexact Interior Point Methods* which may significantly reduce the overall computational effort.

On the basis of two benchmark problems, we show that our method can achieve a significant speedup compared to sequential and parallel direct solvers. However, the standard Schwarz method suffers from the well-known increase in Krylov iterations for an increasing number of processors. To remedy this issue, we propose a two-level method and numerical experiments indicate that this method is capable of conserving a stable number of iterations.

This paper is structured in the following way: In section 2 we state the TCOPF problem and the application of a suitable IPM. Further, we briefly describe the structure of the arising linear systems. In section 3.1 we propose a Schwarz method for solving these linear systems in parallel. In section 3.2 and 3.3 we formulate an extension of the previously defined method, leading to the concept of *Multilevel Methods*. In section 4, we investigate the numerical behaviour of our methods based on two test cases. We summarise our results in section 5.

2 Time Constrained Optimal Power Flow

In this section we begin with formulating the TCOPF problem we want so solve. Afterwards a common way of solving this nonlinear optimization problem by means of IPM is presented. In the last subsection we give a short description of the linear systems arising in every IPM-iteration.

2.1 TCOPF Formulation

We consider a power grid of $N_\mathcal{B}$ nodes denoted by $\mathcal{B} := \{1, \dots, N_\mathcal{B}\}$ and $N_\mathcal{E}$ transmission lines denoted by $\mathcal{E} \subset \mathcal{B} \times \mathcal{B}$ with corresponding admittance matrix

$$Y = G + jB \in \mathbb{C}^{N_\mathcal{B} \times N_\mathcal{B}} \text{ with } G, B \in \mathbb{R}^{N_\mathcal{B} \times N_\mathcal{B}} \text{ and } j = \sqrt{-1}.$$

Note that $Y = Y^T$ and $Y_{kl} \neq 0$ if and only if there is a transmission line connecting node k and l, i.e., $kl \in \mathcal{E}$. Therefore, Y is a sparse matrix for most real world power grids.

The complex voltage at every node $k \in \mathcal{B}$ is given by

$$V_k = U_k \exp j\Theta_k,$$

with voltage amplitude U_k and voltage phase angle difference Θ_k between node k and the reference node 1, i.e., $\Theta_1 = 0$.

Furthermore, let $\mathcal{P} := \{1, \dots, N_\mathcal{P}\}$ and $\mathcal{Q} := \{1, \dots, N_\mathcal{Q}\}$ denote the set of active and reactive power injection processes, respectively. For every $l \in \mathcal{P}$, $P_{G,l}$ is the variable of active power injection and for every $l \in \mathcal{Q}$, $Q_{G,l}$ is the variable of reactive power injection. Every power injection process is assigned to a specific node by means of power injection matrices $C_P \in \mathbb{R}^{N_\mathcal{B} \times N_\mathcal{P}}$ and $C_Q \in \mathbb{R}^{N_\mathcal{B} \times N_\mathcal{Q}}$ with

$$(C_P)_{kl} = \begin{cases} 1, & \text{active power of process } l \text{ is injected at node } k \\ 0, & \text{else} \end{cases}$$

$$(C_Q)_{kl} = \begin{cases} 1, & \text{reactive power of process } l \text{ is injected at node } k \\ 0, & \text{else.} \end{cases}$$

Denoting the active and reactive power load at node $k \in \mathcal{B}$ with $P_{D,k}$ and $Q_{D,k}$ respectively, the AC power flow equations read [18]

$$C_P P_G - P_D - P_F(U, \Theta) = 0 \in \mathbb{R}^{N_\mathcal{B}}, \tag{2.1}$$

$$C_Q Q_G - Q_D - Q_F(U, \Theta) = 0 \in \mathbb{R}^{N_\mathcal{B}}, \tag{2.2}$$

with

$$P_{F,k}(U, \Theta) = \sum_{l=1}^{N_\mathcal{B}} U_k U_l (G_{kl} \cos \Theta_{kl} + B_{kl} \sin \Theta_{kl}), \ k \in \mathcal{B},$$

$$Q_{F,k}(U, \Theta) = \sum_{l=1}^{N_\mathcal{B}} U_k U_l (G_{kl} \sin \Theta_{kl} - B_{kl} \cos \Theta_{kl}), \ k \in \mathcal{B},$$

where $\Theta_{kl} = \Theta_k - \Theta_l$. $P_{F,k}$ and $Q_{F,k}$ can also be written in terms of active and reactive power flow p_{kl}, q_{kl} over all lines incident to node k:

$$\begin{aligned} P_{F,k}(U,\Theta) &= \sum_{l\neq k, Y_{kl}\neq 0} p_{kl}(U_k, U_l, \Theta_k, \Theta_l), \\ Q_{F,k}(U,\Theta) &= \sum_{l\neq k, Y_{kl}\neq 0} q_{kl}(U_k, U_l, \Theta_k, \Theta_l). \end{aligned}$$

Equation (2.1) can be seen as balance equation where the difference $(C_P P_G - P_D)_k$ of injected and extracted power at node k equals the power flow $P_{F,k}$ to all nodes adjacent to k. This is a direct consequence of Kirchhoff's circuit law. The same holds for (2.2).

In the following we rewrite equations (2.1), (2.2) as

$$AC(\Theta, U, P_G, Q_G, P_D, Q_D) = 0. \tag{2.3}$$

With this definitions at hand, we can formulate the *Optimal Power Flow* (OPF) problem [7]:

$$(OPF) \begin{cases} \min\limits_{\Theta,U,P_G,Q_G} f(\Theta, U, P_G, Q_G) \text{ s.t.} \\ AC(\Theta, U, P_G, Q_G, P_D, Q_D) = 0 \\ \Theta_1 = 0 \\ p_{kl}^2(U,\Theta) + q_{kl}^2(U,\Theta) \leq S_{kl}, \qquad \forall k \neq l \in \mathcal{B}, Y_{kl} \neq 0 \\ U_{\min} \leq U \leq U_{\max} \\ P_{G,\min} \leq P_G \leq P_{G,\max} \\ Q_{G,\min} \leq Q_G \leq Q_{G,\max}. \end{cases}$$

Here, $\leq$ is understood componentwise and $U_{\min}, U_{\max}, P_{G,\min}, P_{G,\max}, Q_{G,\min}, Q_{G,\max}$ are lower and upper bounds for node voltages and injected power, respectively. S_{kl} denotes the upper limit for the power flow over transmission line kl.

The cost function f accounts for costs of generated active power (e.g., in \$ per MWh) and has the form [7]

$$f(U, \Theta, P_G, Q_G) = \sum_{l \in \mathcal{P}} a_{l2} P_{G,l}^2 + a_{l1} P_{G,l}, \quad a_{l2}, a_{l1} \geq 0, \ a_{l2} + a_{l1} > 0.$$

The optimization problem (OPF) can be interpreted as finding the operation state of minimal generating costs for a given power grid described by the admittance matrix Y at a single point in time with current power load P_D, Q_D.

In contrast to OPF, TCOPF states the problem of finding an optimal operation state over a given time period instead at a single point in time. To be more precise, let $0 = T_1 < T_2 < \ldots < T_{N_T} = T$ denote a uniform partition of a time period of interest $[0, T]$ with constant step size $\tau = T_t - T_{t-1}$. To every time step T_t one can assign a vector of corresponding optimization variables $x^t \in \mathbb{R}^{n^{x,t}}$ with

$$x^t = \begin{pmatrix} \Theta^t \\ U^t \\ P_G^t \\ Q_G^t \end{pmatrix},$$

where $\Theta^t, U^t \in \mathbb{R}^{N_\mathcal{B}}, P_G^t \in \mathbb{R}^{N_\mathcal{P}}, Q_G^t \in \mathbb{R}^{N_\mathcal{Q}}$ correspond to the optimization variables in (OPF). Consequently, $n^{x,t} = 2N_\mathcal{B} + N_\mathcal{P} + N_\mathcal{Q}$.

Furthermore, the power demand may vary over time, i.e., for every time step t there are vectors P_D^t, Q_D^t describing the power demand at every node k. When dealing with renewable energy sources, varying upper and lower bounds for power generation $P_{G,\min}^t, P_{G,\max}^t, Q_{G,\min}^t, Q_{G,\max}^t$ are possible as well. Additionally the change of active power generation between consecutive time steps is bounded from above, i.e., one has to impose ramp constraints of the form

$$|P_{G,l}^{t+1} - P_{G,l}^t| \leq \tau R_l, \ \forall l \in \mathcal{P}, \ t = 1, \ldots, N_T - 1.$$

Finally, our TCOPF problem is given as [20]

$$(\mathit{TCOPF})\left\{\begin{array}{rll} \min\limits_{(\Theta^t,U^t,P_G^t,Q_G^t)_t} & \sum\limits_{t=1}^{N_T} \tau f(\Theta^t,U^t,P_G^t,Q_G^t) \text{ s.t.} & \\ & AC(\Theta^t,U^t,P_G^t,Q_G^t,P_D^t,Q_D^t)=0, & t=1,\dots,N_T \\ & \Theta_1^t = 0, & t=1,\dots,N_T \\ & p_{kl}^2(U^t,\Theta^t)+q_{kl}^2(U^t,\Theta^t) \le S_{kl}, & t=1,\dots,N_T \\ & U_{\min} \le U^t \le U_{\max}, & t=1,\dots,N_T \\ & P_{G,\min}^t \le P_G^t \le P_{G,\max}^t, & t=1,\dots,N_T \\ & Q_{G,\min}^t \le Q_G^t \le Q_{G,\max}^t, & t=1,\dots,N_T \\ & P_G^{t+1}-P_G^t \le \tau R, & t=1,\dots,N_T-1 \\ & P_G^t-P_G^{t+1} \le \tau R, & t=1,\dots,N_T-1. \end{array}\right.$$

One can rewrite this optimization problem in a more compact form as

$$\left\{\begin{array}{rll} \min\limits_{(x^t)_t} & \sum\limits_{t=1}^{N_T} \tau f(x^t) \text{ s.t.} & \\ & g^t(x^t)=0, & t=1,\dots,N_T \\ & h_I^t(x^t)\le 0, & t=1,\dots,N_T \\ & h_R^t(x^{t+1},x^t)\le 0, & t=1,\dots,N_T-1. \end{array}\right. \tag{2.4}$$

or

$$\left\{\begin{array}{l} \min\limits_x F(x) \text{ s.t.} \\ \quad g(x)=0 \\ \quad h(x)\le 0 \end{array}\right.$$

with $x = \begin{pmatrix} x^1 & \dots & x^{N_T}\end{pmatrix} \in \mathbb{R}^{n^x}$, $n^x = \sum_{t=1}^{N_T} n^{x,t}$ and twice continuously differentiable functions

$$\begin{aligned} F &: \mathbb{R}^{n^x} \to \mathbb{R} \\ g &: \mathbb{R}^{n^x} \to \mathbb{R}^{N_T(2N_{\mathcal{B}}+1)} \\ h &: \mathbb{R}^{n^x} \to \mathbb{R}^{2N_T(N_{\mathcal{E}}+N_{\mathcal{B}}+N_{\mathcal{P}}+N_{\mathcal{Q}})+2(N_T-1)N_{\mathcal{P}}}. \end{aligned}$$

Note that the ramp constraints given by h_R establish the only couplings between variables of different time steps. Without them, a solution to (TCOPF) could be computed by solving N_T independent (OPF) problems.

There exist other possible formulations of (TCOPF) as well. For example, one could include changes in active power generation in the cost function f or add additional intertemporal constraints corresponding to energy storage systems or power generation contracts (e.g., [20]).

2.2 Primal Dual Interior Point Method for TCOPF

Due to the nonlinearity of the AC equations (2.3), (TCOPF) is a nonlinear and non-convex optimization problem where the number of optimization variables x is proportional to the grid size $2\mathcal{B}+\mathcal{P}+\mathcal{Q}$ and to the number of considered time steps N_T. When considering real world scenarios involving national transmission grids and time horizons of several days, typical grid sizes are of order 10^4 and number of time steps are of order 10^2. For such kind of large-scale optimization problems, the use of Interior Point Methods is a common approach. We briefly describe the application of a *Primal Dual Interior Point Method* (PDIPM) for our problem (e.g., [7]).

For (TCOPF) we define the corresponding Lagrangian function by

$$\mathcal{L}\colon \mathbb{R}^{n^x}\times\mathbb{R}^{n^\lambda}\times\mathbb{R}^{n^\mu}\to\mathbb{R},\ (x,\lambda,\mu)\mapsto F(x)+\lambda^T g(x)+\mu^T h(x),$$

with $n^\lambda = N_T(2N_\mathcal{B}+1)$ and $n^\mu = 2N_T(N_\mathcal{E}+N_\mathcal{B}+N_\mathcal{P}+N_\mathcal{Q}) + 2(N_T-1)N_\mathcal{P}$.

Assuming additional constraint qualifications like (LICQ), for every local minimum x^* of $(TCOPF)$ there exist corresponding Lagrangian multipliers λ^*, μ^* such that (x^*, λ^*, μ^*) solves the following KKT-conditions [7]:

$$(KKT)\begin{cases} \nabla_x \mathcal{L}(x,\lambda,\mu) = \nabla F(x)^T + \nabla g(x)^T\lambda + \nabla h(x)^T\mu = 0 & \text{(stationarity)}\\ \nabla_\lambda \mathcal{L}(x,\lambda,\mu) = g(x) = 0 & \text{(primal feasibility)}\\ \nabla_\mu \mathcal{L}(x,\lambda,\mu) = h(x) \le 0 & \text{(primal feasibility)}\\ \forall i = 1,\dots,n^\mu: \ \mu_i h_i(x) = 0 & \text{(complementary slackness)}\\ \mu \ge 0. & \text{(dual feasibility)} \end{cases}$$

The idea of PDIPM is based on solving the system of nonlinear equations and inequalities (KKT) by means of a perturbed Newton's method. In a first step, one removes nonlinear inequalities by introducing additional slack variables $s \in \mathbb{R}^{n^\mu}$, resulting in an equivalent system:

$$(KKT_0)\begin{cases} \nabla F(x)^T + \nabla g(x)^T\lambda + \nabla h(x)^T\mu = 0\\ g(x) = 0\\ h(x) + s = 0\\ \forall i = 1,\dots,n^\mu: \ \mu_i s_i = 0\\ s \ge 0\\ \mu \ge 0. \end{cases}$$

At first glance, one might think of solving (KKT_0) with Newton's method by starting at a feasible point $(s^{(0)}, \mu^{(0)}) \in \Omega := \{(s,\mu) \ge 0\}$ and restricting the step size for each Newton update $(\Delta x, \Delta s, \Delta\lambda, \Delta\mu)$ to ensure $(s^{(l)}, \mu^{(l)}) \in \Omega$ for all $l \ge 0$. Due to $\mu_i s_i = 0$, the exact solution of (KKT_0) lies at the boundary of the feasible region Ω. Thus, the Newton iterates $(s^{(l)}, \mu^{(l)})$ might be forced towards $\partial\Omega$ to achieve $\mu_i s_i = 0$ while other components of the corresponding nonlinear residual are far from being close to 0. This may lead to very short step sizes and therefore poor convergence.

To remedy this effect, the complementary slackness condition is relaxed to $\mu_i s_i = \gamma$ for some $\gamma > 0$, yielding the following perturbed KKT system:

$$(KKT_\gamma)\begin{cases} \nabla F(x)^T + \nabla g(x)^T\lambda + \nabla h(x)^T\mu = 0\\ g(x) = 0\\ h(x) + s = 0\\ \forall i = 1,\dots,n^\mu: \ \mu_i s_i = \gamma\\ s \ge 0\\ \mu \ge 0. \end{cases}$$

(KKT_γ) can be written in a more compact form as

$$\mathcal{F}_\gamma(x,s,\lambda,\mu) := \begin{pmatrix} \nabla F(x) + \nabla g(x)^T\lambda + \nabla h(x)^T\mu \\ g(x) \\ h(x) + s \\ S\mu - \gamma e \end{pmatrix} = 0, \ s, \mu \ge 0, \tag{2.5}$$

with $S = \operatorname{diag}(s_1, \dots, s_{n^\mu})$ and $e = (1,\dots,1) \in \mathbb{R}^{n^\mu}$. Solving $\mathcal{F}_\gamma = 0$, $\gamma > 0$ by applying Newton's method is in general much faster then using Newton's method for $\mathcal{F}_0 = 0$.

Finally, the PDIPM is obtained by solving a sequence of nonlinear systems of equations $F_{\gamma_k} = 0$ with Newton's method, where γ_k is chosen such that $\gamma_k \to 0$. An additional step size control ensures that $(s^{(k,l)}, \mu^{(k,l)}) \in \Omega$ for all obtained iterates. Under certain assumptions (including second order sufficient conditions, strict complementary condition and (LICQ)) one can show that for sufficiently small γ_k a locally unique solution $(x^{(k,*)}, s^{(k,*)}, \lambda^{(k,*)}, \mu^{(k,*)})$ of

$$F_{\gamma_k}(x,s,\lambda,\mu) = 0, \ s, \mu \ge 0 \tag{2.6}$$

exists. This sequence of points is also known as *central path* and it converges to the exact solution $(x^*, s^*, \lambda^*, \mu^*)$ of (KKT_0) if $\gamma_k \to 0$. Furthermore, these assumptions imply that $\nabla \mathcal{F}_{\gamma_k}(x, s, \lambda, \mu)$ is nonsingular if (x, s, λ, μ) is sufficiently close to $(x^*, s^*, \lambda^*, \mu^*)$ and if γ_k is sufficiently close to 0 (e.g., chapter 19 in [14]).

In practical implementations the sequence of nonlinear problems (2.6) is not solved exactly. Instead, each system (2.6) is solved with a termination criterion of the form $\|F_{\gamma_k}(x^{(k,l)}, s^{(k,l)}, \lambda^{(k,l)}, \mu^{(k,l)})\| \le \epsilon_{\gamma_k}$ and accordingly defined tolerance ϵ_{γ_k} (e.g., [5]).

We use the approach of updating γ_k after each Newton iteration via

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

with *centering parameter* $\sigma = 0.1$ and *complementary gap* $(\mu^{(k)})^T s^{(k)}$ [7], yielding the following algorithm [21]:

Algorithm 1. *Primal Dual Interior Point Method for TCOPF*

```
Set  $\lambda^{(0)} = 0,\ \mu^{(0)} = e,\ \gamma_0 = 1,\ k = 0$
Choose  $x^{(0)},\ s^{(0)}$
Compute  $r_0 := F_{\gamma_0}(x^{(0)}, s^{(0)}, \lambda^{(0)}, \mu^{(0)})$
Do until convergence
    Compute  $\bar{A}_k := \nabla F_{\gamma_k}(x^{(k)}, s^{(k)}, \lambda^{(k)}, \mu^{(k)})$
    Solve  $\bar{A}_k \Delta_k = -r_k,\ \Delta_k = (\Delta x^{(k)}, \Delta s^{(k)}, \Delta \lambda^{(k)}, \Delta \mu^{(k)})$
    Compute primal and dual step size $\alpha_p, \alpha_d$
    Set  $(x^{(k+1)}, s^{(k+1)}) = (x^{(k)}, s^{(k)}) + \alpha_p(\Delta x^{(k)}, \Delta s^{(k)})$
    Set  $(\lambda^{(k+1)}, \mu^{(k+1)}) = (\lambda^{(k)}, \mu^{(k)}) + \alpha_d(\Delta \lambda^{(k)}, \Delta \mu^{(k)})$
    Set  $\gamma_{k+1} = \sigma \frac{(\mu^{(k+1)})^T s^{(k+1)}}{n^\mu}$
    Compute  $r_{k+1} := F_{\gamma_{k+1}}(x^{(k+1)}, s^{(k+1)}, \lambda^{(k+1)}, \mu^{(k+1)})$
    Check convergence criteria
    Set  $k \leftarrow k + 1$
```

Here, we choose $x^{(0)}$ by setting

$$\Theta^t = 0,\ U^t = \frac{1}{2}(U_{\max} + U_{\min}),\ P^t = \frac{1}{2}(P^t_{\max} + P^t_{\min}),\ Q^t = \frac{1}{2}(Q^t_{\max} + Q^t_{\min}),\ t = 1, \ldots, N_T$$

and $s^{(0)}$ by

$$s_i^{(0)} = \begin{cases} -h_i(x^{(0)}), & h_i(x^{(0)}) < -1 \\ 1, & \text{else} \end{cases}.$$

The primal and dual step sizes are obtained by the *fraction to the boundary rule* [14] in order to prevent $s^{(k)}$ and $\mu^{(k)}$ from becoming negative:

$$\alpha_p = \max\{\alpha \in (0, 1],\ s + \alpha \Delta s \ge (1 - \rho)s\}$$
$$\alpha_d = \max\{\alpha \in (0, 1],\ \mu + \alpha \Delta \mu \ge (1 - \rho)\mu\}$$

with $\rho = 0.99995$. The iteration terminates if all of the following conditions are satisfied [21]:

$$\frac{\max\{\|g(x)\|_\infty, \max_i\{h_i(x)\}\}}{1 + \max\{\|x\|_\infty, \|z\|_\infty\}} < \epsilon_{feas} \text{ feasibility condition}$$

$$\frac{\|\nabla_x \mathcal{L}(x, \lambda, \mu)\|_\infty}{1 + \max\{\|\lambda\|_\infty, \|\mu\|_\infty\}} < \epsilon_{grad} \text{ gradient condition}$$

$$\frac{(\mu^T s)}{1 + \|x\|_\infty} < \epsilon_{comp} \text{ complementary condition}$$

$$\frac{|F(x) - F(x^-)|}{1 + |F(x^-)|} < \epsilon_{cost} \text{ cost condition}$$

where we omitted the iteration index $k+1$ and denote x^k by x^-.

Remark Algorithm 1 is a very basic PDIPM implementation without any globalization strategy. There exist more sophisticated implementations using line-search and trust-region methods for ensuring global convergence (e.g., [19]). Since we focus on parallel linear algebra techniques for TCOPF problems, for our purpose a locally convergent algorithm is sufficient.

2.3 KKT Matrix in PDIPM

When applying Interior Point Methods, the main computational effort lies in the solution of arising linear systems. In our case we have to solve the system

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

in every step of the PDIPM algorithm 1. In the following, we omit the iteration index k. The Newton matrix is given by

$$\bar{A} := \bar{A}(x, s, \lambda, \mu) = \nabla \mathcal{F}_{\gamma}(x, s, \lambda, \mu) = \begin{pmatrix} \nabla^2_{xx}\mathcal{L}(x,\lambda,\mu) & 0 & (\nabla g(x))^T & (\nabla h(x))^T \\ 0 & M & 0 & S \\ \nabla g(x) & 0 & 0 & 0 \\ \nabla h(x) & I & 0 & 0 \end{pmatrix} \tag{2.7}$$

with diagonal matrices $M = \text{diag}(\mu_1, \ldots, \mu_{n^\mu})$, $S = \text{diag}(s_1, \ldots, s_{n^\mu})$ and identity matrix $I \in \mathbb{R}^{n^\mu \times n^\mu}$. Eliminating the second and fourth row yields a reduced matrix (see appendix),

$$A := A(x, s, \lambda, \mu) := \begin{pmatrix} \nabla^2_{xx}\mathcal{L}(x,\lambda,\mu) + (\nabla h(x))^T\Sigma(\nabla h(x)) & (\nabla g(x))^T \\ \nabla g(x) & 0 \end{pmatrix} \tag{2.8}$$

with diagonal matrix $\Sigma = \text{diag}(\frac{\mu_1}{s_1}, \ldots, \frac{\mu_{n^\mu}}{s_{n^\mu}})$.

Note that A is a symmetric saddle point matrix and thus indefinite. Matrices of this structure are also labelled as *KKT matrix*.

The dimension of A is $n^x + n^\lambda = N_T(4N_\mathcal{B} + N_\mathcal{P} + N_\mathcal{Q} + 1)$. In contrast, $\bar{A}$ is of dimension $n^x + n^\lambda + 2n^\mu = N_T(8N_\mathcal{B} + 7N_\mathcal{P} + 5N_\mathcal{Q} + 4N_\mathcal{E} + 1) - 2N_\mathcal{P}$. Since $N_\mathcal{E} \gtrapprox 2N_\mathcal{B}$ for real world transmission grids, $\dim(\bar{A}) \gtrapprox 3\dim(A)$. Additionally, the sparsity structure of $(\nabla h(x))^T\Sigma\nabla h(x)$ is almost a subset of the sparsity structure of $\nabla^2_{xx}\mathcal{L}(x,\lambda,\mu)$. There are just few additional non-zeros due to ramp constraints h_R and lower / upper bounds on P, Q, U. This is due to the fact, that the sparsity structure of all Jacobian matrices of nonlinear constraints equals the sparsity structure of the admittance matrix Y.

To summarise, the dimension of A is reduced by a factor of ≈ 3 compared to $\bar{A}$ and the $(1,1)$-block of A has almost the same sparsity structure than the $(1,1)$-block of $\bar{A}$. In the following, we work with matrix A.

When dealing with linear systems arising from IPM, one generally has to face the problem of ill-conditioning. To see this, assume that strict complementary holds at the exact solution $(x^*, s^*, \lambda^*, \mu^*)$ of (KKT_0), i.e.,

$$s_i^*\mu_i^* = 0,\ s_i^* + \mu_i^* > 0,\ s_i^*, \mu_i^* \geq 0 \text{ for all } i.$$

If the inequality constraint i is active at x^*, i.e., $s_i^* = h_i(x^*) = 0$, then $s_i^{(k)} \to 0$ and $\mu_i^{(k)} \to \mu^* > 0$ for $k \to \infty$. Consequently, the corresponding entry in the diagonal matrix Σ becomes very large as the IPM iterate approaches the exact solution:

$$\sigma_i^{(k)} = (\Sigma^{(k)})_{ii} = \frac{\mu_i^{(k)}}{s_i^{(k)}} \to \infty \text{ for } k \to \infty,$$

which yields an increasing condition number $\kappa(A) = \|A\|\|A^{-1}\|$. For this reason, many IPM software packages (like IPOPT [9]) use direct methods such as LDL^T-factorizations to solve the arising linear systems. In contrast, iterative linear solvers like GMRES [15] are very sensitive to ill-conditioned matrices, unless a good preconditioner is used. In exchange, they offer a higher potential of parallelization and allow the use of inexact Interior Point methods [3].

The distribution of the spectrum $\mathcal{S}(K) = \{\lambda \in \mathbb{C},\ \lambda \text{ is eigenvalue of } K\}$ of a matrix K is an indicator for the convergence behaviour of GMRES applied to $Kv = b$. If $K \in \mathbb{R}^{N\times N}$ is diagonalizable, i.e., $K = V\Gamma V^{-1}$ with $\Gamma = \operatorname{diag}(\lambda_1, \dots, \lambda_N)$, $V \in \mathbb{R}^{N\times N}$, then it can be shown that the residual $r_k = b - Kv_k$ in the k-th step of GMRES satisfies

$$\frac{\|r_k\|_2}{\|r_0\|_2} \le \|V\|_2 \|V^{-1}\|_2 \min_{p\in\mathcal{P}_k, p(0)=1} \max_{\lambda\in\mathcal{S}(K)} |p(\lambda)|$$

with $\mathcal{P}_k$ being the space of polynomials of degree less then or equal to k. This estimate indicates that a spectrum which is clustered away from 0 is beneficial for the speed of convergence of GMRES. In particular, GMRES terminates with the exact solution after at most $|\mathcal{S}(K)|$ steps.

For general nonlinear optimization problems with corresponding KKT matrix

$$K = \begin{pmatrix} H & B^T \\ B & 0 \end{pmatrix},\ H \in \mathbb{R}^{n\times n},\ B \in \mathbb{R}^{m\times n},\ m \le n$$

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

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

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

One can show that $\mathcal{S}(\tilde{K}^{-1}K) = \{1\} \cup \{\lambda,\ \exists u\ Z^T H Z u = \lambda Z^T \tilde{H} Z u\}$ with Z being a basis of the nullspace null$B$ of B [12]. Since K is assumed to be nonsingular, it holds rank$B = m$ and dim(null$B$) $= n - m$ according to Lemma 3.2. Therefore, at most $n - m$ 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. For $(TCOPF)$ it holds

$$n - m = n^x - n^\lambda = N_T(N_\mathcal{P} + N_\mathcal{Q} - 1).$$

For many optimization problems, the constraint preconditioner shows a good performance when combined with GMRES. However, factorizing $\tilde{K}$ in parallel might be difficult.

In section 3 we describe how to exploit the special structure given by $(TCOPF)$ to construct a parallelizable preconditioner. To get further insight into this structure, note that the linear system to be solved is of the following form (compare (2.8)):

$$\begin{pmatrix} \nabla^2_{xx}\mathcal{L} + (\nabla h)^T \Sigma (\nabla h) & (\nabla g)^T \\ \nabla g & 0 \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta\lambda \end{pmatrix} = \begin{pmatrix} r_x \\ r_\lambda \end{pmatrix}. \tag{2.9}$$

One can construct a permutation matrix $P \in \{0,1\}^{(n^x+n^\lambda)\times(n^x+n^\lambda)}$ such that

$$P\begin{pmatrix} \Delta x \\ \Delta\lambda \end{pmatrix} = P\begin{pmatrix} \Delta x^1 \\ \vdots \\ \Delta x^{N_T} \\ \Delta\lambda^1 \\ \vdots \\ \Delta\lambda^{N_T} \end{pmatrix} = \begin{pmatrix} \Delta x^1 \\ \Delta\lambda^1 \\ \vdots \\ \Delta x^{N_T} \\ \Delta\lambda^{N_T} \end{pmatrix},$$

where x^t is defined in section 2.1 and λ^t denotes the Lagrangian multipliers corresponding to the equality constraints $g^t(x^t) = 0$. Applying this permutation to A yields a block tri-diagonal matrix

$$\hat{A} := PAP^T = \begin{pmatrix} A_{11} & A_{12} & 0 & 0 & \dots & \dots & 0 \\ A_{12}^T & A_{22} & A_{23} & 0 & \dots & \dots & 0 \\ 0 & A_{23}^T & A_{33} & A_{34} & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & A_{N_{T-2}N_{T-1}}^T & A_{N_{T-1}N_{T-1}} & A_{N_{T-1}N_T} & 0 \\ 0 & \dots & \dots & 0 & A_{N_{T-2}N_{T-1}}^T & A_{N_{T-1}N_{T-1}} & A_{N_{T-1}N_T} \\ 0 & \dots & \dots & \dots & 0 & A_{N_{T-1}N_T}^T & A_{N_T N_T} \end{pmatrix}. \tag{2.10}$$

Each diagonal block A_{tt} is of saddle point structure and can be interpreted as KKT matrix for a single (OPF) problem at time step t. The off-diagonal blocks A_{tt+1} describe the couplings between consecutive time steps t and $t+1$. For $(TCOPF)$ presented in section 2.1, these couplings are given by the ramp constraints $|P^{t+1} - P^t| \leq \tau R$. Since there aren't any couplings between variables x^t, x^s with $|t-s| > 1$ in our TCOPF formulation, the corresponding off-diagonal blocks A_{ts} vanish.

3 Multilevel Domain Decomposition for TCOPF

We describe the construction of a parallelizable preconditioner that allows the use of iterative linear solvers inside the PDIPM algorithm 1. We make use of domain decomposition techniques that are well established for solving partial differential equations (PDEs) in parallel. The *additive Schwarz Method* (ASM) is such kind of method. In section 3.1 we present the application of ASM in context of TCOPF problems. In section 3.2 and 3.3 we propose an extension to ASM, involving the concepts of *Coarse Grid Correction* and *Multilevel ASM*.

3.1 Additive Schwarz Method

The original Schwarz method was formulated in 1870 as theoretical tool for proofing existence of elliptic PDEs on complicated domains [16]. 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 methods in context of PDEs [17]. In this work we use the additive variant of Schwarz Methods as preconditioner for GMRES in order to solve linear systems given by (2.9).

3.1.1 Mathematical Formulation

To apply ASM as preconditioner for the KKT matrix $A = A(x, s, \lambda, \mu) \in \mathbb{R}^{n \times n}$ defined by (2.9), we decompose the set of time steps $\mathcal{T} = \{1, \ldots, N_T\}$ into q nonoverlapping sub-domains:

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

Afterwards, each sub-domain $\tilde{\mathcal{T}}_l$ is augmented by additional s_{ol} time steps on both ends, yielding an overlapping decomposition of $\mathcal{T}$ (see Figure 1):

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

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

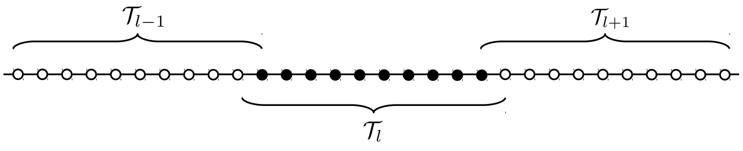

Figure 1: Decomposition of time steps with overlap $s_{ol} = 1$.

We fix the following notation:

$$\begin{aligned} n^{x,t} &: \text{ dimension of } x^t \\ n^{\lambda,t} &: \text{ dimension of } \lambda^t \\ n_l^x &= \sum_{t\in\mathcal{T}_l} n^{x,t}, \quad n^x = \sum_{t=1}^{N_T} n^{x,t}, \\ n_l^\lambda &= \sum_{t\in\mathcal{T}_l} n^{\lambda,t}, \quad n^\lambda = \sum_{t=1}^{N_T} n^{\lambda,t}, \\ n_l &= n_l^x + n_l^\lambda, \quad n = n^x + n^\lambda. \end{aligned}$$

When restricting optimization variables to their components contained in $\mathcal{T}_l$, we write

$$\begin{aligned} x_{[l]} &= [x^t]_{t\in\mathcal{T}_l}, \quad \lambda_{[l]} = [\lambda^t]_{t\in\mathcal{T}_l}, \\ \mu_{I,[l]} &= [\mu_I^t]_{t\in\mathcal{T}_l}, \quad s_{I,[l]} = [s_I^t]_{t\in\mathcal{T}_l}, \\ \mu_{R,[l]} &= [\mu_R^t]_{t\in\mathcal{T}_l\cup\{t_l^- -1\}}, \quad s_{R,[l]} = [s_R^t]_{t\in\mathcal{T}_l\cup\{t_l^- -1\}}, \end{aligned}$$

with primal variables x, multipliers λ corresponding to equality constraints, slack variables s and multipliers μ corresponding to inequality constraints. The latter ones are split into components assigned to h_I and h_R, see (2.4). For all expressions involving $\tilde{}$, $\mathcal{T}_l$ is replaced by $\tilde{\mathcal{T}}_l$. The constraint functions g and h are restricted in the same way:

$$\begin{aligned} g_{[l]}(x_{[l]}) &= [g^t(x^t)]_{t\in\mathcal{T}_l}, \\ h_{I,[l]}(x_{[l]}) &= [h_I^t(x^t)]_{t\in\mathcal{T}_l}, \\ h_{R,[l]}(x_{[l]}, x^{t_l^- -1}, x^{t_l^+ +1}) &= [h_R^t(x^{t+1}, x^t)]_{t\in\mathcal{T}_l\cup\{t_l^- -1\}}, \\ h_{[l]}(x_{[l]}, x^{t_l^- -1}, x^{t_l^+ +1}) &= \begin{pmatrix} h_{I,[l]}(x_{[l]}) \\ h_{R,[l]}(x_{[l]}, x^{t_l^- -1}, x^{t_l^+ +1}) \end{pmatrix} \end{aligned}$$

and

$$\begin{aligned} \Sigma_{I,[l]} &= \operatorname{diag}\left(\frac{\mu_{I,[l],1}}{s_{I,[l],1}}, \dots, \frac{\mu_{I,[l],|\mu_{I,[l]}|}}{s_{I,[l],|\mu_{I,[l]}|}}\right), \\ \Sigma_{R,[l]} &= \operatorname{diag}\left(\frac{\mu_{R,[l],1}}{s_{R,[l],1}}, \dots, \frac{\mu_{R,[l],|\mu_{R,[l]}|}}{s_{R,[l],|\mu_{R,[l]}|}}\right), \\ \Sigma_{[l]} &= \begin{pmatrix} \Sigma_{I,[l]} & 0 \\ 0 & \Sigma_{R,[l]} \end{pmatrix}. \end{aligned}$$

For $l = 1, \dots, q$ we define by

$$\begin{pmatrix} \Delta x \\ \Delta\lambda \end{pmatrix}_{[l]} := \begin{pmatrix} \Delta x_{[l]} \\ \Delta\lambda_{[l]} \end{pmatrix} \in \mathbb{R}^{n_l}, \quad \begin{pmatrix} \Delta x \\ \Delta\lambda \end{pmatrix}_{[\tilde{l}]} := \begin{pmatrix} \Delta x_{[\tilde{l}]} \\ \Delta\lambda_{[\tilde{l}]} \end{pmatrix} \in \mathbb{R}^{\tilde{n}_l},$$

the components of the solution vector of (2.9) that can be assigned to time steps in $\mathcal{T}_l$ and $\tilde{\mathcal{T}}_l$, respectively. Let further $R_l \in \{0,1\}^{n_l\times n}$, $\tilde{R}_l \in \{0,1\}^{\tilde{n}_l\times n}$ be restriction matrices corresponding to the subsets $\mathcal{T}_l$ and $\tilde{\mathcal{T}}_l$ such that

$$R_l \begin{pmatrix} \Delta x \\ \Delta\lambda \end{pmatrix} = \begin{pmatrix} \Delta x \\ \Delta\lambda \end{pmatrix}_{[l]} \quad \text{and} \quad \tilde{R}_l \begin{pmatrix} \Delta x \\ \Delta\lambda \end{pmatrix} = \begin{pmatrix} \Delta x \\ \Delta\lambda \end{pmatrix}_{[\tilde{l}]} \tag{3.1}$$

holds. The restriction matrices are given as

$$R_l = \begin{pmatrix} R_l^x & 0 \\ 0 & R_l^\lambda \end{pmatrix} \quad \text{and} \quad \tilde{R}_l = \begin{pmatrix} \tilde{R}_l^x & 0 \\ 0 & \tilde{R}_l^\lambda \end{pmatrix} \tag{3.2}$$

with sub-restriction matrices of the following form

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

Here, I_m denotes the identity matrix in $\mathbb{R}^m$ and the size of the zero matrices may vary for each restriction matrix and each l.

With this definitions at hand, one can define local sub-matrices of A by

$$A_l := R_l A R_l^T. \tag{3.3}$$

In section 3.1.3 we analyse the structure and invertibility of these sub-matrices.

In the following, we assume that all sub-matrices A_l are nonsingular. Then the *multiplicative Schwarz Method* for (approximately) solving $Av = b$ is given by [17]:

Algorithm 2. *Multiplicative Schwarz Method* $v = \mathrm{MSM}(b)$

```
Set  $v = 0$
For  $l = 1, \dots, q$
    $v \leftarrow v + R_l^T A_l^{-1} R_l (b - Av)$
```

In every iteration l, the current residual $b - Av$ is restricted to sub-domain $\mathcal{T}_l$. Solving with A_l^{-1} gives a local approximation to the error and prolongating the error back with R_l^T yields a correction on $\mathcal{T}_l$. Thus, the multiplicative Schwarz method can be seen as sequential defect correction algorithm. Omitting residual updates in each iteration yields the parallelizable ASM:

Algorithm 3. *Additive Schwarz Method* $v = \mathrm{ASM}(b)$

```
Set  $v = 0$
For  $l = 1, \dots, q$
    $v \leftarrow v + R_l^T A_l^{-1} R_l b$
```

which can be written as

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

The right preconditioned linear system for solving $Av = b$ is then given by

$$A M_{ASM} u = b, \; v = M_{ASM} u. \tag{3.4}$$

Since M_{ASM} is symmetric but in general not positive definite, we use GMRES for solving the unsymmetric system (3.4).

Remark By means of coloring techniques, it is possible to parallelize the multiplicative Schwarz method up to a certain degree (e.g., [17]).

3.1.2 Implementation

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

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

Applying ASM as preconditioner of an iterative method requires computation of $v_k = M_{ASM} b_k$ in each iteration k. For this step, each process l first restricts b to its overlapping part $b_l := R_l b$ which requires communication with process $l-1$ and $l+1$. The computation of $A_l^{-1} b_l$ is done by one forward and backward solve with L_l and U_l. This step doesn't involve any communication. As final step, the local solution $v_l := A_l^{-1} b_l$ is prolongated back to update the global solution vector v. This step again requires communicating the overlapping part of v_l to process $l-1$ and $l+1$.

One can further improve the performance of ASM by using the so called *restricted* version of ASM [6] which is given by

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

For this preconditioner just the nonoverlapping part of the local solution v_l is prolongated instead of the entire (overlapping) vector. Experiments show a beneficial behaviour in terms of GMRES iterations compared to standard ASM [6]. Furthermore, prolongation by $\tilde{R}_l$ doesn't involve any communication.

3.1.3 Analysis of Local KKT Matrices

In this section, we investigate the structure of the local sub-matrices A_l. For convenience, let $1 < l < q$. For $l = 1$ and $l = q$ the same results hold with minor modifications.

By definition,

$$\begin{aligned} A_l(x, s, \lambda, \mu) &= R_l A(x, s, \lambda, \mu) R_l^T \\ &= \begin{pmatrix} R_l^x & 0 \\ 0 & R_l^\lambda \end{pmatrix} \begin{pmatrix} \nabla^2_{xx}\mathcal{L} + (\nabla h)^T \Sigma (\nabla h) & (\nabla g)^T \\ \nabla g & 0 \end{pmatrix} \begin{pmatrix} (R_l^x)^T & 0 \\ 0 & (R_l^\lambda)^T \end{pmatrix} \\ &= \begin{pmatrix} R_l^x \left(\nabla^2_{xx}\mathcal{L} + (\nabla h)^T \Sigma (\nabla h)\right) (R_l^x)^T & R_l^x (\nabla g)^T (R_l^\lambda)^T \\ R_l^\lambda (\nabla g) (R_l^x)^T & 0 \end{pmatrix}. \end{aligned}$$

Since there are no temporal couplings in the equality constraints g, ∇g is of block diagonal form:

$$\nabla g(x) = \begin{pmatrix} \nabla_{x^1} g^1(x^1) & 0 & 0 & \dots & 0 \\ 0 & \nabla_{x^2} g^2(x^2) & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & \nabla_{x^{N_T-1}} g^{N_T-1}(x^{N_T-1}) & 0 \\ 0 & \dots & 0 & 0 & \nabla_{x^{N_T}} g^{N_T}(x^{N_T}) \end{pmatrix} \tag{3.5}$$

and

$$R_l^\lambda \left(\nabla g(x)\right) (R_l^x)^T = \nabla g_{[l]}(x_{[l]}). \tag{3.6}$$

Further,

$$\begin{aligned} R_l^x (\nabla h)^T \Sigma (\nabla h) (R_l^x)^T &= R_l^x \left(\sum_{t=1}^{N_T} (\nabla h_I^t(x^t))^T \Sigma_I^t (\nabla h_I^t(x^t)) \right) (R_l^x)^T \\ &\quad + R_l^x \left(\sum_{t=1}^{N_T-1} (\nabla h_R^t(x^{t+1}, x^t))^T \Sigma_R^t (\nabla h_R^t(x^{t+1}, x^t)) \right) (R_l^x)^T \\ &= \sum_{t \in \mathcal{T}_l} R_l^x (\nabla h_I^t(x^t))^T \Sigma_I^t (\nabla h_I^t(x^t)) (R_l^x)^T \\ &\quad + \sum_{t=t_l^- -1}^{t_l^+} R_l^x (\nabla h_R^t(x^{t+1}, x^t))^T \Sigma_R^t (\nabla h_R^t(x^{t+1}, x^t)) (R_l^x)^T \\ &= (\nabla_{x_{[l]}} h_{I,[l]}(x_{[l]}))^T \Sigma_{I,[l]} (\nabla_{x_{[l]}} h_{I,[l]}(x_{[l]})) \\ &\quad + (\nabla_{x_{[l]}} h_{R,[l]}(x_{[l]}, x^{t_l^- -1}, x^{t_l^+ +1}))^T \Sigma_{R,[l]} (\nabla_{x_{[l]}} h_{R,[l]}(x_{[l]}, x^{t_l^- -1}, x^{t_l^+ +1})) \\ &= (\nabla_{x_{[l]}} h_{[l]}(x_{[l]}, x^{t_l^- -1}, x^{t_l^+ +1}))^T \Sigma_{[l]} (\nabla_{x_{[l]}} h_{[l]}(x_{[l]}, x^{t_l^- -1}, x^{t_l^+ +1})) \end{aligned}$$

where we used that ∇h_I is of similar form as ∇g.

Note that $h_{R,[l]}$ is a function of x^t for $t \in \mathcal{T}_l \cup \{t_l^- - 1\} \cup \{t_l^+ + 1\}$, whereas $\nabla_{x_{[l]}} h_{R,[l]}(x)$ is a constant matrix. In the following, we consider $h_{R,[l]}$ as a function of $x_{[l]}$ with some fixed vectors v^-, v^+ in place of $x^{t_l^- - 1}$ and $x^{t_l^+ + 1}$.

Define the local Lagrangian function

$$\mathcal{L}_{[l]}(x_{[l]}, \lambda_{[l]}, \mu_{[l]}) = \sum_{t \in \mathcal{T}_l} \tau f(x^t) + \lambda_{[l]}^T g_{[l]}(x_{[l]}) + \mu_{[l]}^T h_{[l]}(x^{[l]}).$$

Since all temporal couplings are linear, $\nabla^2_{xx}\mathcal{L}$ is block diagonal and it holds

$$R_l^x \, \nabla^2_{xx}\mathcal{L}(x, \lambda, \mu)(R_l^x)^T = \nabla^2_{x_{[l]}x_{[l]}}\mathcal{L}(x, \lambda, \mu) = \nabla^2_{x_{[l]}x_{[l]}}\mathcal{L}_{[l]}(x_{[l]}, \lambda_{[l]}, \mu_{[l]}).$$

Thus, A_l can be written as

$$A_l(x, s, \lambda, \mu) = \begin{pmatrix} H_{[l]}(x_{[l]}, \lambda_{[l]}, \mu_{[l]}) & (\nabla g_{[l]}(x_{[l]}))^T \\ \nabla g_{[l]}(x_{[l]}) & 0 \end{pmatrix} \tag{3.7}$$

with $H_{[l]}(x_{[l]}, \lambda_{[l]}, \mu_{[l]}) = \nabla^2_{x_{[l]}x_{[l]}}\mathcal{L}_{[l]}(x_{[l]}, \lambda_{[l]}, \mu_{[l]}) + (\nabla h_{[l]}(x_{[l]}))^T \Sigma_{[l]} (\nabla h_{[l]}(x_{[l]}))$.

Therefore, A_l is the KKT matrix obtained when applying algorithm 1 to the optimization problem corresponding to the Lagrangian function $\mathcal{L}_{[l]}$. This problem has the form of $(TCOPF)$ with $\mathcal{T}$ replaced by $\mathcal{T}_l$ and additional "boundary conditions" v^-, v^+:

$$(TCOPF_l(v^-, v^+)) \begin{cases} \min\limits_{(x^t)_{t \in \mathcal{T}_l}} \sum\limits_{t \in \mathcal{T}_l} \tau f(x^t) \text{ s.t.} \\ \qquad g^t(x^t) = 0, \; t \in \mathcal{T}_l \\ \qquad h_I^t(x^t) \le 0, \; t \in \mathcal{T}_l \\ \qquad h_R^t(x^{t+1}, x^t) \le 0, \; t \in \mathcal{T}_l \setminus \{t_l^+\} \\ \qquad h_R^{t_l^- - 1}(x^{t_l^-}, v^-) \le 0 \\ \qquad h_R^{t_l^+}(v^+, x^{t_l^+}) \le 0. \end{cases}$$

Using the results above, one can show that the restriction of global KKT points are KKT points of the local problem:

Lemma 3.1. *Let $(x^*, \lambda^*, \mu_I^*, \mu_R^*)$ be a KKT point of $(TCOPF)$. Then $(x^*_{[l]}, \lambda^*_{[l]}, \mu^*_{I,[l]}, \mu^*_{R,[l]})$ is a KKT point of $(TCOPF_l(x^{*,t_l^- - 1}, x^{*,t_l^+ + 1}))$.*

Proof The validity of primal feasibility, dual feasibility and complementary slackness condition in the KKT conditions for $(TCOPF_l(x^{*,t_l^- - 1}, x^{*,t_l^+ + 1}))$ follows directly from the corresponding conditions in (KKT). The stationarity condition for $(TCOPF_l(x^{*,t_l^- - 1}, x^{*,t_l^+ + 1}))$ can be obtained by restricting the stationarity condition in (KKT) with R_l^x. $\square$

In order to analyse the invertibility of A_l we use the following result which is variant of Theorem 3.2 in [4]:

Lemma 3.2. *Let*

$$K = \begin{pmatrix} H & B^T \\ B & 0 \end{pmatrix} \text{ with } H \in \mathbb{R}^{n \times n}, B \in \mathbb{R}^{m \times n}, \; m \le n$$

and $Z \in \mathbb{R}^{n \times l}$ denote a basis of null$B$. *Then it holds:*

i) If K is invertible then rank$B = m$.

ii) If rank$B = m$ *and $Z^T H Z$ is invertible then K is invertible.*

Proof See Appendix.

Assuming nonsingularity of the global KKT matrix A, Lemma 3.2 (i) yields that the Jacobian of equality constraints ∇g has full row rank. Due to the block diagonal structure of ∇g, see (3.5), $\nabla g_{[l]}$ has full row rank as well. Thus, to ensure nonsingularity of A_l, by means of Lemma 3.2 (ii) it is sufficient to ensure nonsingularity of $Z_{[l]}^T H_{[l]} Z_{[l]}$, with $Z_{[l]}$ being a basis of null$\nabla g_{[l]}$.

By choosing a basis of orthonormal vectors $Z_{[l]}$ and adding a multiple of the identity matrix $\epsilon I, \epsilon > 0$ to $H_{[l]}$, $Z_{[l]}^T H_{[l]} Z_{[l]}$ is modified to $Z_{[l]}^T H_{[l]} Z_{[l]} + \epsilon I$. Therefore, nonsingularity of $Z_{[l]}^T H_{[l]} Z_{[l]}$ can be obtained for ϵ being large enough. In a practical implementation this can be done if the LU-factorization of A_l without modification fails. Since A_l is just used as preconditioner, this modification only influences the convergence of the linear solver, but not the result of solving $Av = b$.

3.1.4 Relationship between ASM and Constraint Preconditioner

Let

$$\tilde{A}_l := \tilde{R}_l A \tilde{R}_l^T \in \mathbb{R}^{\tilde{n}_l \times \tilde{n}_l}$$

be the restriction of A to the nonoverlapping sub-domain $\tilde{\mathcal{T}}_l = \{\tilde{t}_l^-, \ldots, \tilde{t}_l^+\}$. In this case, the corresponding ASM

$$\tilde{M}_{ASM} = \sum_{l=1}^{q} \tilde{R}_l^T \tilde{A}_l^{-1} \tilde{R}_l$$

reduces to a block Jacobi method with omitted couplings between variables assigned to $\tilde{t}_l^+$ and $\tilde{t}_{l+1}^-$ for each sub-domain l. Permuting $\tilde{M}_{ASM}$ with P defined by (2.10), yields a block diagonal matrix which is the inverse of the permuted KKT matrix $\hat{A}$ with neglected off-diagonal blocks $A_{t_l^+ t_{l+1}^-}$ for $l = 1, \ldots, q-1$.

Since these omitted couplings only arise in the $(1,1)$-block of A, namely in $(\nabla h)^T \Sigma (\nabla h)$, $\tilde{M}_{ASM}$ has the form of constraint preconditioner for A:

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

As pointed out in section 2.3, 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:

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

For a low number of sub-domains q, one might expect $\tilde{H}$ to be a good approximation to H, leading to a clustered spectrum $\mathcal{S}(\tilde{M}_{ASM} A)$ close to one. However, the more sub-domains, 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. The corresponding behaviour of GMRES preconditioned by $\tilde{M}_{ASM}$ is illustrated by the numerical example in section 4.1.

3.2 Coarse Grid Correction

The main computational effort remains in factorizing A_l and applying M_{ASM}. Obviously, increasing the number of sub-domains q results in smaller sub-matrices A_l and therefore significant reduction of computing time for the factorization step. However, since information about the solution on different sub-domains is only exchanged between neighbouring sub-domains, an increasing number of sub-domains generally leads to an increasing number of GMRES iterations. To remedy this effect, a common approach consists of augmenting ASM with a mechanism that ensures the exchange of global information across all sub-domains. This leads to the concept of *Coarse Grid Correction* [17].

3.2.1 Abstract Coarse Grid Correction

To augment M_{ASM} with a coarse grid correction, we need a restriction operator similar to R_l in Section 3.1.1:

$$R_0 := \begin{pmatrix} R_0^x & 0 \\ 0 & R_0^\lambda \end{pmatrix} \in \mathbb{R}^{n_0 \times n} \text{ with } R_0^x \in \mathbb{R}^{n_0^x \times n^x},\ R_0^\lambda \in \mathbb{R}^{n_0^\lambda \times n^\lambda}. \tag{3.8}$$

In contrast to R_l^x, R_0^x extracts components of x that are distributed over the whole vector instead of a locally clustered part. The same holds for R_0^λ. In the next section, we describe R_0^x and R_0^λ in more detail.

Similar to (3.3) we define a coarse system matrix by restricting the global matrix A:

$$A_0 = R_0 A R_0^T.$$

Now, a two-level ASM is given by the following algorithm [17]:

Algorithm 4. *Two-Level Additive Schwarz Method* $v = \text{ASM2}(b)$

```
Set  $v = 0$
For  $l = 1, \dots, q$
    $v \leftarrow v + R_l^T A_l^{-1} R_l b$
Compute  $r = b - Av$
Set  $v \leftarrow v + R_0^T A_0^{-1} R_0 r$
```

or equivalently,

$$v = M_{ASM} b + \underbrace{R_0^T A_0^{-1} R_0}_{=:M_0}(b - A M_{ASM} b) = (M_0 + (I - M_0 A) M_{ASM}) b =: M_{ASM2} b.$$

Thus, the standard ASM algorithm 3 is extended by an additional defect correction step. ASM2 is an hybrid algorithm with parallelized sub-domain approximation and coarse grid correction being performed in sequential.

Defining the error $e_{ASM} = v_* - v_{ASM}$ w.r.t. the exact solution v_* and the ASM approximation v_{ASM}, it holds

$$r = b - Av_{ASM} = Av_* - Av_{ASM} = A(v_* - v_{ASM}) = Ae_{ASM}.$$

Assuming that local error components are rather small due to the exact sub-domain solutions by A_l^{-1},

$$e_0 = R_0^T A_0^{-1} R_0 r$$

should yield a good approximation to e_{ASM}. Therefore, by

$$v = v_{ASM} + e_0 \approx v_{ASM} + e_{ASM} = v_*$$

one might obtain an improved approximation to the exact solution.

3.2.2 Construction of a Coarse Restriction Operator

In this section we describe the construction of our coarse restriction operator R_0 in more detail. This approach is motivated by the usage of coarse grid corrections in context of numerical solution of PDEs. In this area, coarse systems are usually obtained by discretizing the arising differential operators on the complete domain but with lower resolution [17]. Therefore, our coarse grid $\mathcal{T}_0$ should also cover the whole range of time steps $\mathcal{T}$ but with lower temporal resolution. To this end, define the coarse grid

$$\mathcal{T}_{0,c} := \{t = cs + 1,\ t \in \mathcal{T},\ s \in \mathbb{N}_0\} \cup \{N_T\} \text{ with } c \in \mathbb{N}, c \geq 2.$$

$\mathcal{T}_{0,c}$ contains every c-th time step in $\mathcal{T}$ and both end points. In the following, set $c = 2$ and write $\mathcal{T}_0 = \mathcal{T}_{0,2}$. As in section 3.1.1, we define restrictions of vectors $x \in \mathbb{R}^{n^x}$, $\lambda \in \mathbb{R}^{n^\lambda}$ by

$$x_{[0]} = [x_{[0]}^t]_{t \in \mathcal{T}_0} = [x^t]_{t \in \mathcal{T}_0} \in \mathbb{R}^{n_0^x}$$
$$\lambda_{[0]} = [\lambda_{[0]}^t]_{t \in \mathcal{T}_0} = [\lambda^t]_{t \in \mathcal{T}_0} \in \mathbb{R}^{n_0^\lambda}$$
$$n_0^x = \sum_{t \in \mathcal{T}_0} n^{x,t},\ n_0^\lambda = \sum_{t \in \mathcal{T}_0} n^{\lambda,t},\ n_0 = n_0^x + n_0^\lambda.$$

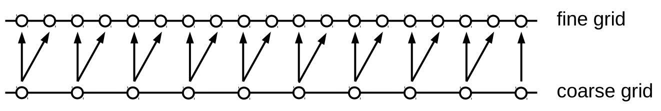

Figure 2: One side extrapolation of variables from coarse to fine space

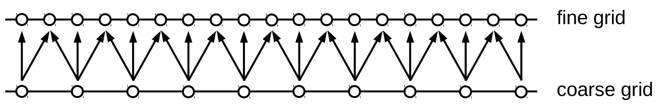

Figure 3: Two side interpolation of variables from coarse to fine space

Furthermore, we define for $t \in \mathcal{T}$ the nearest points in $\mathcal{T}_0$ by

$$\mathcal{I}_-(t) := \max\{t_0 \in \mathcal{T}_0,\ t_0 \leq t\} \text{ and } \mathcal{I}_+(t) := \min\{t_0 \in \mathcal{T}_0,\ t_0 \geq t\}.$$

We construct the restriction operator R_0 by giving a form of its transpose R_0^T, i.e., the prolongation from coarse to fine space variables. Thus, we have to describe how to obtain values for those variables $x^t,\ \lambda^t,\ t \notin \mathcal{T}_0$ that are not part of the coarse space.

Note that in the optimization problem $(TCOPF)$ the equality constraints $g^t(x^t) = 0$ are completely decoupled. In order to preserve this structure, the prolongation operator for multipliers $(R_0^\lambda)^T$ should not introduce any couplings between $\lambda^t_{[0]},\ \lambda^s_{[0]}$ for $t \neq s$. Therefore, we use a one side extrapolation to obtain values for $\lambda^t,\ t \notin \mathcal{T}_0$.

The prolongation operator for multipliers $(R_0^\lambda)^T : \mathbb{R}^{n_0^\lambda} \to \mathbb{R}^{n^\lambda}$ is now given such that a coarse multiplier $\tilde{\lambda} = [\tilde{\lambda}^t]_{t \in \mathcal{T}_0} \in \mathbb{R}^{n_0^\lambda}$ is mapped to the space of fine multipliers by

$$(R_0^\lambda)^T \tilde{\lambda} = \begin{pmatrix} \lambda^1 \\ \vdots \\ \lambda^{N_T} \end{pmatrix} \text{ with } \lambda^t = \tilde{\lambda}^{\mathcal{I}_-(t)}.$$

For primal variables $x^t = \begin{pmatrix} \Theta^t & U^t & P_G^t & Q_G^t \end{pmatrix}^T$ the situation is slightly different. Since there is no coupling between $\Theta^t,\ \Theta^s$ for $t \neq s$ in $(TCOPF)$, we use the same type of prolongation as for dual variables λ. The same holds for U^t and Q_G^t. In contrast, $P_G^t,\ P_G^{t+1}$ are coupled via h_R^t. This type of variables are prolongated by interpolation. To sum up, the prolongation operator for primal variables $(R_0^x)^T : \mathbb{R}^{n_0^x} \to \mathbb{R}^{n^x}$ maps a coarse primal variable $\tilde{x} = [\tilde{x}^t]_{t \in \mathcal{T}_0} \in \mathbb{R}^{n_0^x}$, $\tilde{x}^t = \begin{pmatrix} \tilde{\Theta}^t & \tilde{U}^t & \tilde{P}_G^t & \tilde{Q}_G^t \end{pmatrix}^T$ into the fine space via

$$\begin{aligned}
(R_0^x)^T \tilde{x} &= \begin{pmatrix} x^1 \\ \vdots \\ x^{N_T} \end{pmatrix},\ x^t = \begin{pmatrix} \Theta^t \\ U^t \\ P_G^t \\ Q_G^t \end{pmatrix} \text{ such that} \\
\Theta^t &= \tilde{\Theta}^{\mathcal{I}_-(t)} \\
U^t &= \tilde{U}^{\mathcal{I}_-(t)} \\
Q_G^t &= \tilde{Q}_G^{\mathcal{I}_-(t)} \\
P_G^t &= \frac{t - \mathcal{I}_-(t)}{\mathcal{I}_+(t) - \mathcal{I}_-(t)} \tilde{P}_G^{\mathcal{I}_-(t)} + \frac{\mathcal{I}_+(t) - t}{\mathcal{I}_+(t) - \mathcal{I}_-(t)} \tilde{P}_G^{\mathcal{I}_+(t)}
\end{aligned}$$

Both kind of prolongation operators are illustrated in Figure 2 and 3.

3.2.3 Implementation

The setup of M_{ASM2} consists of setting up M_{ASM} and two additional steps, namely constructing and factorizing the coarse system matrix A_0. These steps can be done either in sequential or in parallel.

In the sequential case, construction of A_0 requires communication between one master processor (computes and stores A_0 in its local memory) and all other other processors l (each storing $\tilde{R}_l A$ in its local memory). Afterwards, the restriction operator R_0 is applied by the master process to obtain A_0 and a LU-factorization is computed.

In the parallel case, each processor l stores the local part $\tilde{R}_l R_0^T$ of R_0 in its local memory and construction of A_0 is done in parallel which involves communication with processors $l-1$ and $l+1$. To compute an LU-factorization of A_0 one can use a parallel direct solver. Since the parallel efficiency for these kind of solvers often deteriorates with increasing number of processors, it might be advantageous to use just a subset of all available processors.

When applying M_{ASM2}, the residual $r = b - Av_{ASM}$ has to be computed after the application of M_{ASM}. This computation is done in parallel by all processors that were used to compute $v_{ASM} = M_{ASM}b$. Here, communication is necessary for distributing v_{ASM} over all processors. The computation of $R_0^T A_0^{-1} R_0 r$ is done by those processors that were previously used to set up A_0. Each such processors extracts its local part of the residual r and performs one forward and backward solve. Finally, the result of $A_0^{-1} R_0 r$ is prolongated to the fine space and added to the global vector v. Each of these steps involves global communication.

3.3 Multilevel ASM

The numerical experiments in section 4 will show that the previously defined coarse grid correction can substantially reduce the number of iterations of ASM-preconditioned GMRES. However, the dimension of the corresponding coarse system matrix A_0 is only divided by a factor of 2 compared to the dimension of the original matrix A. So A_0 might still be too large to be factorized in reasonable time. Fortunately, there exists a (quite straightforward) solution for this problem: Instead of factorizing A_0 and computing an exact solution of $e = A_0^{-1} R_0 r$ in algorithm 4, we solve the system $A_0 e = R_0 r$ approximately by applying again the two-level ASM algorithm 4. This can be done recursively, leading to the concept of *Multilevel Methods* [11].

3.3.1 Mathematical Formulation

In the following, we consider a hierarchy of $K+1$ grids of different temporal resolution given by

$$\begin{aligned} \mathcal{T}^K &= \mathcal{T} \\ \mathcal{T}^k &= \{t = 2s+1,\ t \in \mathcal{T}^{k+1},\ s \in \mathbb{N}_0\} \cup \{N_T\},\ k = K-1, \dots, 0 \end{aligned}$$

Each grid, except of the coarsest, is decomposed into q overlapping sub-grids, see section 3.1.1,

$$\mathcal{T}^k = \bigcup_{l=1}^{q} \mathcal{T}_l^k,\ k = 1, \dots, K$$

and for each sub-grid $\mathcal{T}_l^k$ let $R_l^k \in \{0,1\}^{n_l^k \times n^k}$ denote the corresponding restriction operator as defined in (3.1) and (3.2). Here,

$$n_l^k = \sum_{t \in \mathcal{T}_l^k} n^{x,t} + n^{\lambda,t},\ n^k = \sum_{t \in \mathcal{T}^k} n^{x,t} + n^{\lambda,t}.$$

With R_0^k we denote the restriction operator from $\mathcal{T}^k$ to $\mathcal{T}^{k-1}$, see (3.8). For each level, we define the corresponding global system matrix by

$$\begin{aligned} A^K &= A \\ A^k &= R_0^{k+1} A^{k+1} (R_0^{k+1})^T,\ k = K-1, \dots, 0 \end{aligned}$$

and sub-matrices by

$$A_l^k = R_l^k A^k (R_l^k)^T,\ l = 1, \dots, q,\ k = 1, \dots, K.$$

The ASM preconditioner on each level $k \geq 1$ is then obtained by

$$M_{ASM}^k b := \sum_{l=1}^{q} (R_l^k)^T (A_l^k)^{-1} R_l^k b.$$

Our multilevel additive Schwarz method (MLASM) is now given as V-cycle in the standard multilevel framework with an ASM-preconditioned *Richardson Iteration* as pre- and post-smoother, see chapter 4 in [11]:

Algorithm 5. *Multilevel Additive Schwarz Method* $v = \mathrm{MLASM}(r, k)$

```
If  $k = 0$
    Compute  $v = (A^0)^{-1}r$          (0) exact coarse solution
    Return  $v$
Else
    $v^k = S^k(0, r, \nu_1)$            (1) pre smoothing
    $r^k = r - A^k v^k$                 (2) residual computation
    $r^{k-1} = R_0^k r^k$               (3) restriction
    $e^{k-1} = MLASM(r^{k-1}, k-1)$     (4) approximate error
    $e^k = (R_0^k)^T e^{k-1}$           (5) prolongation
    $v^k \leftarrow v^k + e^k$          (6) defect correction
    $v = v^k + S^k(v^k, r, \nu_2)$      (7) post smoothing
    Return  $v$
```

The preconditioned Richardson iteration at level k is defined by

Algorithm 6. *Preconditioned Richardson Iteration* $v = S^k(x, r, \nu)$

```
Set  $v = x$
For  $i = 1, \dots, \nu$
    $v \leftarrow v + M^k_{ASM}(r - A^k v)$
```

where ν denotes the number of smoothing steps. Typical values for ν_i are in the range of 1 to 5. We use the Richardson iteration, because it is one of the simplest iterative methods for solving linear systems. It only consists of applying ASM ν times with a residual update after each iteration.

3.3.2 Implementation

Like in the previous cases, we use MLASM as right preconditioner for GMRES applied to $Av = b$, i.e., we solve

$$Bu = b,\ v = \mathrm{MLASM}(u, K)$$

with the operator

$$B\colon \mathbb{R}^n \to \mathbb{R}^n,\ u \mapsto A\ \mathrm{MLASM}(u, K).$$

For the parallel implementation of algorithm 5 on q processors we combine the data structurs presented in section 3.1.2 and 3.2.3, i.e., every processor l stores $\tilde{R}_l^k A^k,\ k = 1, \dots, K$ in its local memory. The coarse system matrix A^0 is distributed over a subset of q_0 processors, where q_0 depends on the dimension of A^0. As mentioned in section 3.2.3, using $q_0 < q$ might be computationally more efficient if $\mathcal{T}^0$ just covers a few time steps. Setting up M^k_{ASM} and $L^0U^0 = A^0$ is done successively for $k = 1, \dots, K$ but parallel within each level k.

The costs for the setup phase are dominated by LU-factorizations of A_l^k. Assume that the number of floating point operations for computing $LU = A_l^k$ is approximately given by $C(T_l^k) \approx c(T_l^k)^\alpha$ with $T_l^k = |\mathcal{T}_l^k|$, $\alpha \geq 1$ and c being independent of k, l. For simplicity, assume further that $T_l^k = \frac{1}{q}\frac{N_T}{2^{K-k}}$, i.e., we ignore the overlap of sub-domains s_{ol}. Then the number of floating point operations C_l for computing factorizations of A_l^k for all levels $k = 1, \dots, K$ on each processor l can approximately be computed by

$$C_l = \sum_{k=1}^K C(T_l^k) = c\sum_{k=1}^K \left(\frac{N_T}{q2^{K-k}}\right)^\alpha = cN_T^\alpha \frac{1}{q^\alpha}\sum_{k=1}^K \left(\frac{1}{2^\alpha}\right)^{K-k} = \bar{C}\frac{1}{q^\alpha}\frac{1-\frac{1}{2^{\alpha K}}}{1-\frac{1}{2^\alpha}}.$$

If the factorization of A^0 is computed in sequential, the total number of floating point operations on each processor l is given by

$$\bar{C}_l = \bar{C}\left(\frac{1}{q^\alpha}\frac{1-\frac{1}{2^{\alpha K}}}{1-\frac{1}{2^\alpha}} + \frac{1}{2^{\alpha K}}\right) =: \bar{C}\,\rho_\alpha(q,K)$$

with $\bar{C}$ denoting the number of operations for computing $LU = A$. Thus, the maximal possible speedup that can be obtained by GMRES + MLASM compared to a sequential LU-factorization is bounded from above by $\rho_\alpha(q,K)$.

When applying MLASM, the operations (1-7) within level $k \geq 1$ are done consecutively, but every single operation is parallelized. Communication between processors assigned to neighbouring sub-domains $\mathcal{T}_l^k$ is necessary in steps 1,4 and 7. The final coarse grid correction MLASM$(r^0, 0)$ is done as described in section 3.2.3.

4 Numerical Experiments

In this section, we present some results for our previously proposed methods applied to two different $(TCOPF)$ problems. For both test cases we use the grid data case3120sp provided by MATPOWER [21]. This grid consists of 3120 nodes, 248 generators and 3693 transmission lines. We consider a time period $\mathcal{T} = \{1, \dots, 64\}$ with step size $\tau = 1$ hour and constant upper and lower bounds $P^t_{G,\max}, P^t_{G,\min}, Q^t_{G,\max}, Q^t_{G,\min}$ with values provided by case3120sp. In order to obtain temporal varying demand data P^t_D, Q^t_D, we use a scaling function $\delta : [0, 24] \to [0.65, 1]$ that roughly models a typical demand curve for one day. We set

$$P^t_D = \delta(t \bmod 24)\bar{P}_D,\; Q^t_D = \delta(t \bmod 24)\bar{Q}_D,$$

with reference demands $\bar{P}_D, \bar{Q}_D$ contained in case3120sp.

The ramp constraints $|P^{t+1}_G - P^t_G| \leq \tau R$ are modelled in two different ways. For the first test case we set $R = R_I = 0.2P_{G,\max}$ and for the second one we set $R = R_{II} = 0.8\Delta_{P,\max}$. Here, $\Delta_{P,\max,i} := \max_{1\leq t\leq 63} |P^{t+1}_{G,i} - P^t_{G,i}|$ with P_G being the result of solving $(TCOPF)$ with the configuration defined above but without any ramp constraints. It holds $R_{II,i} = \beta_i P_{G,\max,i}$ with $\beta_i \in (0, 0.5)$.

For solving $(TCOPF)$ we use the PDIPM algorithm mips which is written in Matlab code and part of MATPOWER. In this algorithm we replace the standard Matlab backslash operator $\backslash$ for solving linear systems by our own linear solver. This solver consists of GMRES with right preconditioner given by the restricted versions of ASM, ASM2 and MLASM, respectively. For computing LU-factorizations of local systems A_l and the coarse system A_0 we use SuperLU_DIST [13]. Our solver is written in C++ and makes use of the KSPFGMRES, PCASM and PCMG methods provided by PETSc [2] which is compiled in Release mode. All tests are performed on a Linux workstation with Intel Core i7-4770 CPU @ 3.40GHz x 8 processor and 31,4 GB of RAM.

We set the PDIPM termination criteria $\epsilon_{feas} = \epsilon_{grad} = \epsilon_{comp} = \epsilon_{cost} = 10^{-6}$ and solve the arising linear systems with relative residual tolerance $\frac{\|b-Av\|}{\|b\|} \leq 10^{-10}$, where $\|\cdot\|$ denotes the Euclidean norm. When solving linear systems with this accuracy, mips needs as many iterations to converge as when applied with direct solver $\backslash$ for our test cases. GMRES is used without restarting and the maximum number of iterations is set 200. The overlap s_{ol} is set to 1 for all tests.

4.1 Test Case 1

In a first step, we apply the one-level ASM to test case 1 and compare its performance with a block Jacobi preconditioner. As showed in section 3.1.4, the block Jacobi preconditioner has the form of a constraint preconditioner (CP).

Figure 4 shows the progress of PDIPM termination criteria $\epsilon_{feas}, \epsilon_{grad}, \epsilon_{comp}, \epsilon_{cost}$ obtained for $q = 2$ sub-domains. One can observe that convergence towards a feasible point is rather fast, whereas it takes much longer to fulfill the optimality criteria $\epsilon_{grad} < 10^{-6}$ and $\epsilon_{comp} < 10^{-6}$.

In Figure 5 the number of GMRES iterations is plotted over the number IPM iteration index k. For $k \leq 70$ one can observe a slight increase in GMRES+ASM iterations for increasing q. This increase in iterations is typical for one-level domain decomposition methods and is due to the fact, that the local subspace correction operators $R_l^T A_l^{-1} R_l$ allow exchange of information only between neighbouring sub-domains. The more sub-domains, the more iterations it takes to propagate information across all sub-domains. This effect becomes more intense as the PDIPM iterate converges to the exact solution because the entries Σ_{ii} corresponding to active constraints tend towards ∞, leading to an increasingly ill-conditioned matrix A.

CP given by block Jacobi shows a similar behaviour as ASM in the first 30 IPM steps. Afterwards, only CP with two sub-domains can conserve a stable number of iterations, whereas the performance of CP-4 and CP-8 deteriorates.

Figure 6 shows the progress of speedup su(k) for the first k IPM iterations of GMRES+ASM and SuperLU_DIST with $q = 2, 4, 8$ processors compared to the reference solution time obtained by sequential SuperLU_DIST:

$$\mathrm{su}(k) = \frac{\sum_{i=1}^{k} C_1(i)}{\sum_{i=1}^{k} C(i)}$$

with $C_1(i)$ denoting the solution time of sequential SuperLU_DIST in iteration i and $C(i)$ denoting the solution time in iteration i of GMRES+ASM and parallel SuperLU_DIST, respectively. Obviously, the increase in GMRES iteration leads to a decreasing speedup for ASM. Nevertheless, the speedup obtained by GMRES+ASM is significantly higher than the one obtained by parallel SuperLU_DIST.

$N_{\mathcal{B}}$	3120
$N_{\mathcal{E}}$	3693
$N_{\mathcal{P}}$	248
$N_{\mathcal{Q}}$	248
N_T	64
τ	1
$n^{x,t}$	6736
$n^{\lambda,t}$	6241
n^x	431,104
n^λ	399,424
n	830,528

Table 1: Dimensions of test case 1

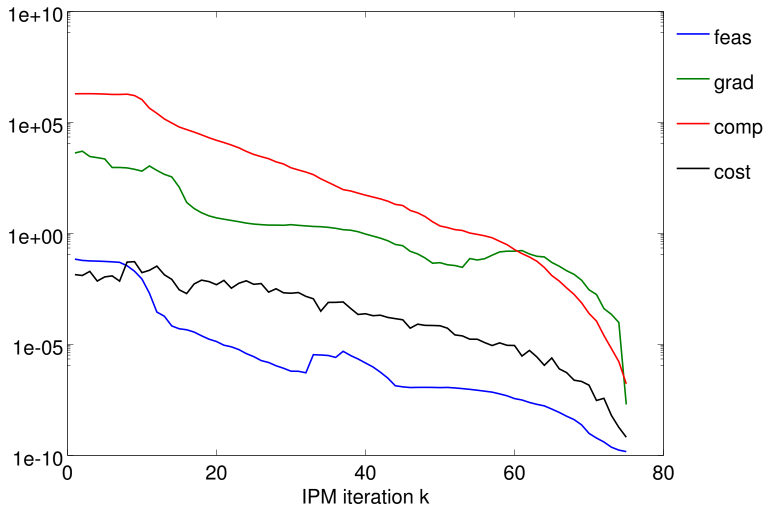

Figure 4: Test case 1: Progress of $\epsilon_{feas}, \epsilon_{grad}, \epsilon_{comp}, \epsilon_{cost}$

4.2 Test Case 2

When applying one-level ASM to test case 2, the number of GMRES iterations changes drastically. In this case, one can observe a large increase in iterations for both increasing number of sub-domains q and increasing IPM iteration index k, see Figure 8. The critical index k until which the number of GMRES iterations is stable, decreases from ≈ 70 in case 1 to ≈ 11 in case 2. This different convergence of GMRES preconditioned by ASM is a result of tighter ramp constraints h_R, leading to corresponding entries Σ_{ii} that grow much faster than in case 1. Therefore, the algebraic coupling in A between variables corresponding to different sub-domains $\mathcal{T}_l$ is stronger. The higher the number of sub-domains q, the more of this strong couplings are ignored by the preconditioner M_{ASM} and the less accurately does it approximate the exact inverse A^{-1}.

Using a coarse grid correction can significantly reduce this effect. As one can observe in Figure 9, the number of GMRES iterations just slightly increases for an increasing number of sub-domains q and it is even lower than in case 1. Moreover, the convergence of GMRES preconditioned by two-level ASM is rather insensitive to k, except of the final 10 iterations. Note that the increase in the average number

method	av. GMRES it.
ASM-2	10.6
ASM-4	18.6
ASM-8	19.6
CP-2	23.7
CP-4	42.4
CP-8	56.6

Table 2: Average number of GMRES iterations

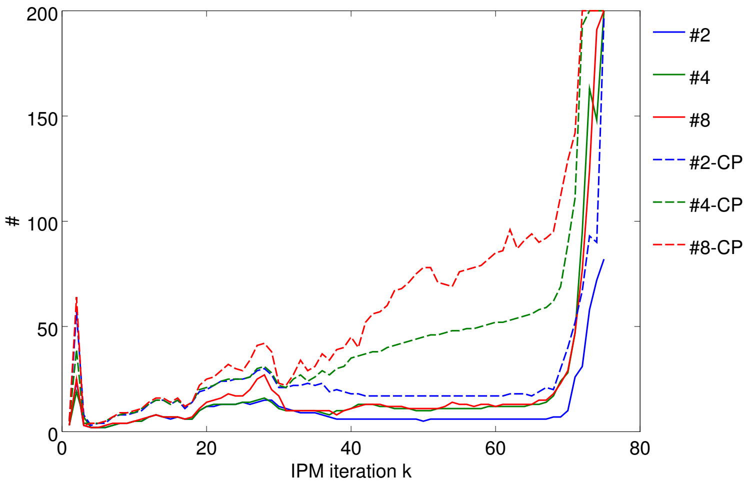

Figure 5: Test case 1: Number of GMRES+(ASM/CP) iterations per PDIPM step for $q = 2, 4, 8$

method	av. comp. time (s)
ASM-2	12.0
ASM-4	10.5
ASM-8	7.2
LU-1	17.5
LU-2	14.4
LU-4	14.5
LU-8	16.2

Table 3: Average solution time

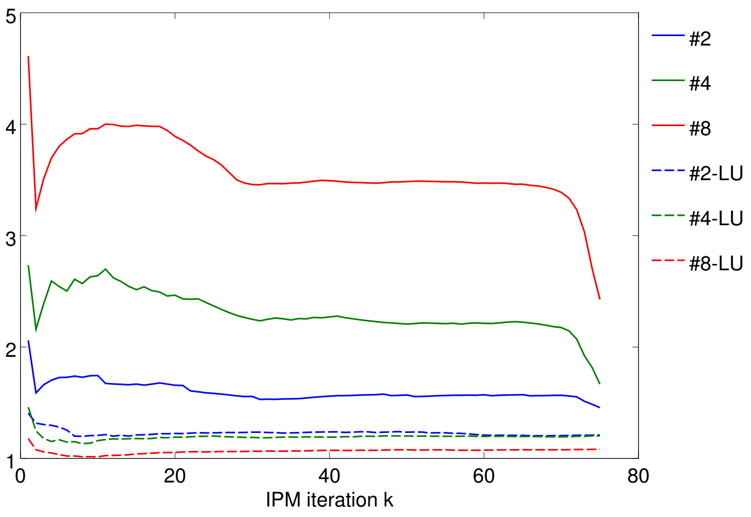

Figure 6: Test case 1: Progress of speedup for GMRES+ASM and SuperLU for $q = 2, 4, 8$

of GMRES iterations from 9.0 for $q = 8$ to 14.8 for $q = 16$ is mainly due to the final IPM iterations. Considering only the first, say, 70 IPM iterations, the average number of GMRES iterations reduces to

MLASM_1-2	MLASM_1-4	MLASM_1-8	MLASM_1-16
4.3	4.6	4.7	6.2

Concerning a practical algorithm, one might think of reducing the number of sub-domains for the final stage of IPM.

Finally, we apply multilevel ASM to case 2 with different number of levels $K = 1, 2, 3$, where MLASM with $K = 1$ reduces to two-level ASM. As MLASM can be interpreted as two-level ASM with inexact coarse grid correction, one might expect less accuracy in approximating A^{-1} for an increasing number of levels K, resulting in a higher number of GMRES iterations. This is exactly what can be observed in Figure 10. However, this increase is rather moderate in our example.

$N_{\mathcal{B}}$	3120
$N_{\mathcal{E}}$	3693
$N_{\mathcal{P}}$	248
$N_{\mathcal{Q}}$	248
N_T	64
τ	1
$n^{x,t}$	6736
$n^{\lambda,t}$	6241
n^x	431,104
n^{λ}	399,424
n	830,528

Table 4: Dimensions of test case 2

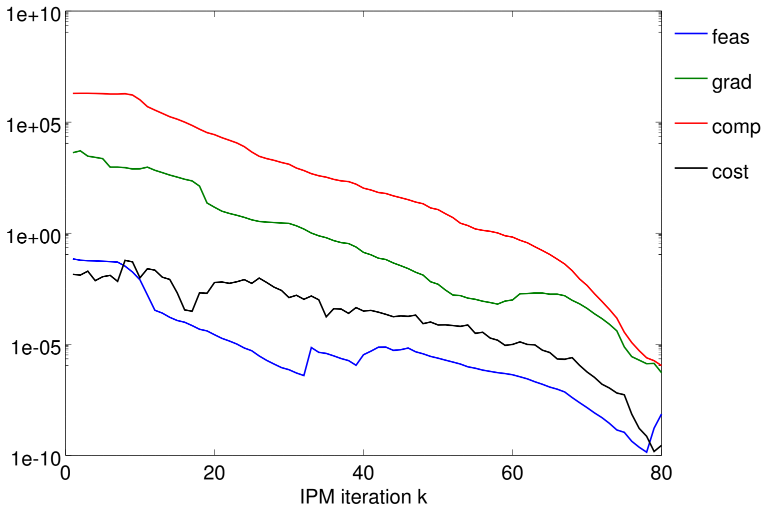

Figure 7: Test case 2: Progress of $\epsilon_{feas}, \epsilon_{grad}, \epsilon_{comp}, \epsilon_{cost}$

5 Conclusions

In this work we propose a way of solving linear systems arising from TCOPF problems in parallel by means of overlapping Schwarz domain decomposition methods. It was shown how to apply these methods in the context of TCOPF and that the local sub-matrices correspond to localized formulations of TCOPF with additional boundary conditions. Numerical tests showed that ASM can lead to a speedup of order 3-4 for 8 processors compared to sequential and parallel direct solvers. However, for tight intertemporal constraints the performance of ASM deteriorates with an increasing number of sub-domains. We augmented ASM with a coarse grid correction and numerical tests indicated that this combination is rather insensitive to the number of sub-domains. Further, we applied the previously developed ASM and coarse grid correction in a multilevel framework.

For our test cases we observed an almost linear complexity of SuperLU_DIST for computing LU-factorization of the KKT matrix, leading to a suboptimal speedup of GMRES+ASM.

In our future work, we will apply our methodology to TCOPF problems with a higher number of intertemporal constraints. For such kind of problems we expect an increasing complexity of direct solvers and therefore larger speedup of our method. Furthermore, we will employ inexact Interior Point methods in order to take advantage of using iterative instead of direct linear solvers. The use of inexact local solvers could lead to additional reduction of computational effort.

From a theoretical point of view, we will investigate whether the abstract Schwarz theory for indefinite matrices [10] is applicable in order to obtain results concerning convergence of GMRES+ASM. The use of nonoverlapping domain decomposition methods for TCOPF could be of further interest.

Acknowledgements

This work was carried out with the support of the German Research Foundation (DFG) within the project HE 4760 / 8-1 in collaboration with the Institute for Industrial Production and the Institute of Electric Energy Systems and High-Voltage Technology at Karlsruhe Institute of Technology.

Further, we want to thank the Heidelberg Institute for Theoretical Studies (HITS) for their support.

method	av. GMRES it.
ASM-2	27.8
ASM-4	40.8
ASM-8	61.5
ASM-16	70.4

Table 5: Average number of GMRES iterations

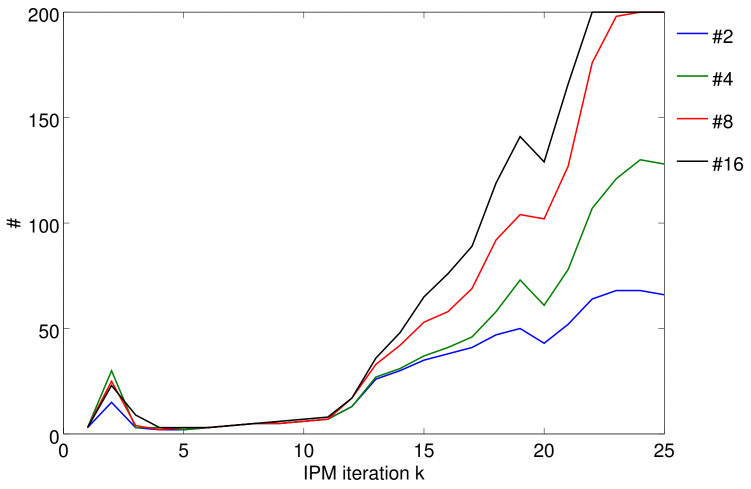

Figure 8: Test case 2: Number of GMRES+ASM iterations per PDIPM step for $q = 2, 4, 8, 16$

method	av. GMRES it.
MLASM_1-2	8.4
MLASM_1-4	8.7
MLASM_1-8	9.0
MLASM_1-16	14.8

Table 6: Average number of GMRES iterations

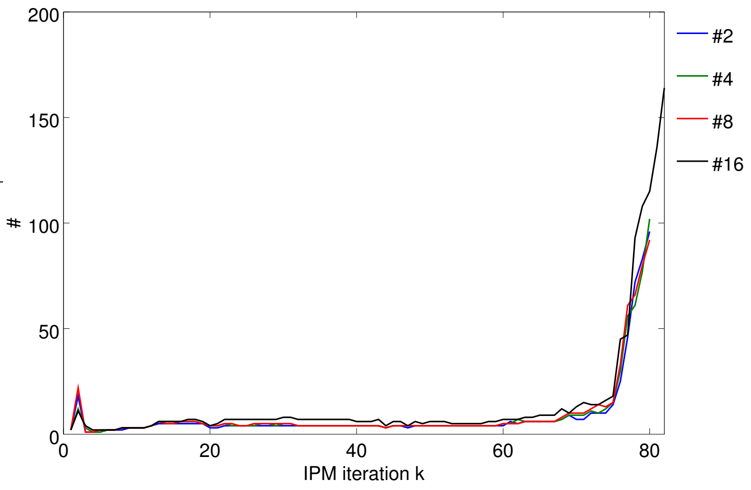

Figure 9: Test case 2: Number of GMRES+MLASM iterations per PDIPM step for $q = 2, 4, 8, 16,\ K = 1$

method	av. GMRES it.
MLASM_1-8	9.0
MLASM_2-8	9.6
MLASM_3-8	13.6

Table 7: Average number of GMRES iterations

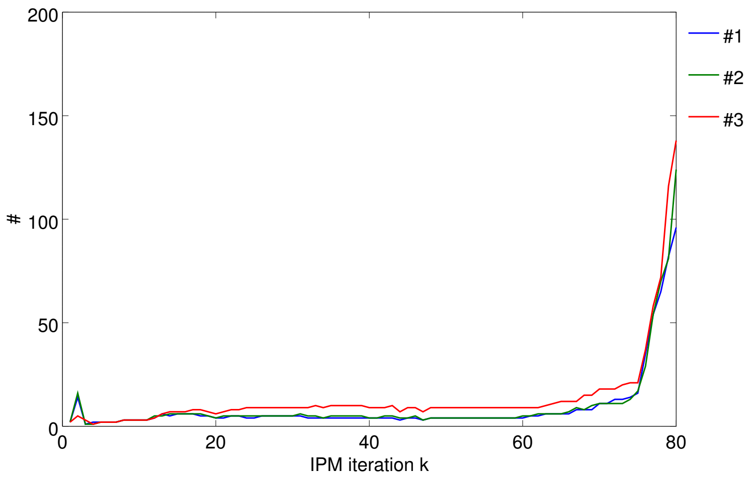

Figure 10: Test case 2: Number of GMRES+MLASM iterations per PDIPM step for $q = 8,\ K = 1, 2, 3$

Appendix

Obtaining a Reduced Form of $\bar{A}$

Consider a linear system with matrix $\bar{A}$ defined in (2.7) as

$$\begin{pmatrix} H & 0 & J_g^T & J_h^T \\ 0 & M & 0 & S \\ J_g & 0 & 0 & 0 \\ J_h & I & 0 & 0 \end{pmatrix} \begin{pmatrix} x \\ s \\ \lambda \\ \mu \end{pmatrix} = \begin{pmatrix} r_x \\ r_s \\ r_\lambda \\ r_\mu \end{pmatrix}.$$

Note that M and S are diagonal matrices with positive entries. Multiplying the second equation by S^{-1} yields

$$\begin{pmatrix} H & 0 & J_g^T & J_h^T \\ 0 & \Sigma & 0 & I \\ J_g & 0 & 0 & 0 \\ J_h & I & 0 & 0 \end{pmatrix} \begin{pmatrix} x \\ s \\ \lambda \\ \mu \end{pmatrix} = \begin{pmatrix} r_x \\ S^{-1} r_s \\ r_\lambda \\ r_\mu \end{pmatrix}.$$

Eliminating $s = -\Sigma^{-1}\mu + M^{-1} r_s$ given by the second equation yields

$$\begin{pmatrix} H & J_g^T & J_h^T \\ J_g & 0 & 0 \\ J_h & 0 & -\Sigma^{-1} \end{pmatrix} \begin{pmatrix} x \\ \lambda \\ \mu \end{pmatrix} = \begin{pmatrix} r_x \\ r_\lambda \\ r_\mu - M^{-1} r_s \end{pmatrix}$$

with $\Sigma = S^{-1}M$. Eliminating $\mu = \Sigma J_h x - \Sigma(r_\mu + M^{-1} r_s) = \Sigma J_h x - \Sigma r_\mu - S^{-1} r_s$ finally yields the reduced KKT system

$$\begin{pmatrix} H + J_h^T \Sigma J_h & J_g^T \\ J_g & 0 \end{pmatrix} \begin{pmatrix} x \\ \lambda \end{pmatrix} = \begin{pmatrix} r_x + J_h^T \Sigma (r_\mu + M^{-1} r_s) \\ r_\lambda \end{pmatrix}.$$

Proof of Lemma 3.2

i) Assume $\operatorname{rank} B < m$. Then the columns of $B^T \in \mathbb{R}^{n \times m}$ are linearly dependent, so there is a $\lambda \in \mathbb{R}^m \setminus \{0\}$ with $B^T \lambda = 0$. Setting $v = \begin{pmatrix} 0 \\ \lambda \end{pmatrix} \neq 0$ yields $Kv = 0$ which is a contradiction to K being nonsingular.

ii) Let $Z \in \mathbb{R}^{n \times l}$ be a matrix with columns forming a basis of $\operatorname{null} B$. Since $\operatorname{rank} B = m$, $\dim(\operatorname{null} B) = n - \dim(\operatorname{range} B) = n - m$. Further, there is a matrix $Y \in \mathbb{R}^{n \times m}$ such that the columns of $[Z\ Y] \in \mathbb{R}^{n \times n}$ form a basis of $\mathbb{R}^n$ and $BY \in \mathbb{R}^{m \times m}$ is nonsingular.

Let $v = \begin{pmatrix} x \\ \lambda \end{pmatrix} \in \mathbb{R}^{n+m}$ such that $Kv = 0$. Then $Bx = 0$ and therefore $x = Zz$ for some $z \in \mathbb{R}^{n-m}$. Multiplying the first equation

$$HZz + B^T \lambda = 0$$

by Z^T yields

$$0 = Z^T HZz + (BZ)^T \lambda = Z^T HZz.$$

Thus, $z = 0$ and $x = Zz = 0$. Multiplying the first equation again by Y^T gives

$$(BY)^T \lambda = 0$$

and therefore $\lambda = 0$. $\square$

References

[1] N. Alguacil and A.J. Conejo. Multiperiod optimal power flow using benders decomposition. *Power Systems, IEEE Transactions on*, 15(1):196–201, Feb 2000.

[2] Satish Balay, William D. Gropp, Lois Curfman McInnes, and Barry F. Smith. Efficient management of parallelism in object oriented numerical software libraries. In E. Arge, A. M. Bruaset, and H. P. Langtangen, editors, *Modern Software Tools in Scientific Computing*, pages 163–202. Birkhäuser Press, 1997.

[3] S. Bellavia. Inexact interior-point method. *Journal of Optimization Theory and Applications*, 96(1):109–121, 1998.

[4] M. Benzi, G.H. Golub, and J. Liesen. Numerical solution of saddle point problems. *Acta Numerica*, 14:1–137, 5 2005.

[5] Richard H. Byrd, Guanghui Liu, and Jorge Nocedal. On the local behavior of an interior point method for nonlinear programming. In *Numerical Analysis 1997*, pages 37–56. Addison Wesley Longman, 1998.

[6] Xiao-Chuan Cai and Marcus Sarkis. A restricted additive schwarz preconditioner for general sparse linear systems. *SIAM Journal on Scientific Computing*, 21(2):792–797, 1999.

[7] Florin Capitanescu, Mevludin Glavic, Damien Ernst, and Louis Wehenkel. Interior-point based algorithms for the solution of optimal power flow problems. *Electric Power Systems Research*, 77(56):508 – 517, 2007.

[8] C.Y. Chung, Wei Yan, and Fang Liu. Decomposed predictor-corrector interior point method for dynamic optimal power flow. *Power Systems, IEEE Transactions on*, 26(3):1030–1039, Aug 2011.

[9] Frank E. Curtis, Johannes Huber, Olaf Schenk, and Andreas Wächter. A note on the implementation of an interior-point algorithm for nonlinear optimization with inexact step computations. *Mathematical Programming*, 136(1):209–227, 2012.

[10] X. Feng and C. Lorton. On Schwarz Methods for Nonsymmetric and Indefinite Problems. *ArXiv e-prints*, August 2013.

[11] W. Hackbusch. *Multi-Grid Methods and Applications*. Springer Series in Computational Mathematics. Springer Berlin Heidelberg, 2003.

[12] Carsten Keller, Nicholas I. M. Gould, and Andrew J. Wathen. Constraint preconditioning for indefinite linear systems. *SIAM Journal on Matrix Analysis and Applications*, 21(4):1300–1317, 2000.

[13] Xiaoye S. Li and James W. Demmel. SuperLU_DIST: A scalable distributed-memory sparse direct solver for unsymmetric linear systems. *ACM Trans. Mathematical Software*, 29(2):110–140, June 2003.

[14] Jorge Nocedal and Steve J. Wright. *Numerical optimization.* Springer Series in Operations Research and Financial Engineering. Springer, Berlin, 2006. NEOS guide http://www-fp.mcs.anl.gov/otc/Guide/.

[15] Youcef Saad and Martin H Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, 7(3):856–869, July 1986.

[16] H.A. Schwarz. *Ueber einen Grenzübergang durch alternirendes Verfahren.* Vierteljahrsschrift der Naturforschenden Gesellschaft in Zürich. Zürcher u. Furrer, 1870.

[17] Barry Francis Smith, Petter E. Bjrstad, and William D. Gropp. *Domain decomposition : parallel multilevel methods for elliptic partial differential equations.* Cambridge University Press, Cambridge, 1996. 1re dition en 1996 et 1re dition broche en 2004.

[18] Andréa A. Sousa, Geraldo L. Torres, and Claudio A. Canizares. Robust Optimal Power Flow Solution Using Trust Region and Interior-Point Methods. *IEEE Transactions on Power Systems*, PP(99), 2010.

[19] Andre L. Tits, Andreas Wächter, Sasan Bakhtiari, Thomas J. Urban, and Craig T. Lawrence. A primal-dual interior-point method for nonlinear programming with strong global and local convergence properties. *SIAM Journal on Optimization*, 14(1):173–199, 2003.

[20] K. Xie and Y.H. Song. Dynamic optimal power flow by interior point methods. *Generation, Transmission and Distribution, IEE Proceedings-*, 148(1):76–84, Jan 2001.

[21] R.D. Zimmerman, C.E. Murillo-Sanchez, and R.J. Thomas. Matpower: Steady-state operations, planning, and analysis tools for power systems research and education. *Power Systems, IEEE Transactions on*, 26(1):12–19, Feb 2011.

Preprint Series of the Engineering Mathematics and Computing Lab

recent issues

No. 2015-03	Martin Wlotzka, Vincent Heuveline: Block-asynchronous and Jacobi smoothers for a multigrid solver on GPU-accelerated HPC clusters
No. 2015-02	Nicolai Schoch, Fabian Kißler, Markus Stoll, Sandy Engelhardt, Raffaele de Simone, Ivo Wolf, Rolf Bendl, Vincent Heuveline: Comprehensive Pre- & Post-Processing for Numerical Simulations in Cardiac Surgery Assistance
No. 2015-01	Teresa Beck, Martin Baumann, Leonhard Scheck, Vincent Heuveline, Sarah Jones: Comparison of mesh-adaptation criteria for an idealized tropical cyclone problem
No. 2014-02	Christoph Paulus, Stefan Suwelack, Nicolai Schoch, Stefanie Speidel, Rüdiger Dillmann, Vincent Heuveline: Simulation of Complex Cuts in Soft Tissue with the Extended Finite Element Method (X-FEM)
No. 2014-01	Martin Wlotzka, Vincent Heuveline: A parallel solution scheme for multiphysics evolution problems using OpenPALM
No. 2013-02	Nicolai Schoch, Stefan Suwelack, Rüediger Dillmann, Vincent Heuveline: Simulation of Surgical Cutting in Soft Tissue using the Extended Finite Element Method (X-FEM)
No. 2013-01	Martin Schindewolf, Björn Rocker, Wolfgang Karl, Vincent Heuveline: Evaluation of two Formulations of the Conjugate Gradients Method with Transactional Memory
No. 2012-07	Andreas Helfrich-Schkarbanenko, Vincent Heuveline, Roman Reiner, Sebastian Ritterbusch: Bandwidth-Efficient Parallel Visualization for Mobile Devices
No. 2012-06	Thomas Henn, Vincent Heuveline, Mathias J. Krause, Sebastian Ritterbusch: Aortic Coarctation simulation based on the Lattice Boltzmann method: benchmark results
No. 2012-05	Vincent Heuveline, Eva Ketelaer, Staffan Ronnas, Mareike Schmidtobreick, Martin Wlotzka: Scalability Study of HiFlow^3 based on a Fluid Flow Channel Benchmark
No. 2012-04	Hartwig Anzt, Armen Beglarian, Suren Chilingaryan, Andrew Ferrone, Vincent Heuveline, Andreas Kopmann: A unified Energy Footprint for Simulation Software
No. 2012-03	Vincent Heuveline, Chandramowli Subramanian: The Coffee-table Book of Pseudospectra
No. 2012-02	Dominik P.J. Barz, Hendryk Bockelmann, Vincent Heuveline: Electrokinetic optimization of a micromixer for lab-on-chip applications
No. 2012-01	Sven Janko, Björn Rocker, Martin Schindewolf, Vincent Heuveline, Wolfgang Karl: Software Transactional Memory, OpenMP and Pthread implementations of the Conjugate Gradients Method - a Preliminary Evaluation
No. 2011-17	Hartwig Anzt, Jack Dongarra, Vincent Heuveline, Piotr Luszczek: GPU-Accelerated Asynchronous Error Correction for Mixed Precision Iterative Refinement
No. 2011-16	Vincent Heuveline, Sebastian Ritterbusch, Staffan Ronnås: Augmented Reality for Urban Simulation Visualization