ENGINEERING MATHEMATICS
AND COMPUTING LAB

UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

# A time step reduction method for Multi-Period Optimal Power Flow problems

Nils Schween, Nico Meyer-Hübner, Philipp Gerstner, Vincent Heuveline

Preprint No. 2019-02

Preprint Series of the Engineering Mathematics and Computing Lab (EMCL)

www.emcl.iwr.uni-heidelberg.de

Preprint Series of the Engineering Mathematics and Computing Lab (EMCL)
ISSN 2191–0693
Preprint No. 2019-02



## Affiliation of the Authors

Schween, N.[a,1], Meyer-Hübner, N.[b], Gerstner, P.[a], Heuveline, V.[a]

[a] *Engineering Mathematics and Computing Lab (EMCL), Interdisciplinary Center for Scientific Computing (IWR), Heidelberg University, Germany*
[b] *Institute of Electric Energy Systems and High-Voltage Technology (IEH), Karlsruhe Institute of Technology (KIT), Germany*
[1] *Corresponding Author: Nils Schween, nils.schween@uni-heidelberg.de*

## Impressum

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

Im Neuenheimer Feld 205,
69120 Heidelberg
Germany



# A time step reduction method for Multi-Period Optimal Power Flow problems

Nils Schween, Nico Meyer-Hübner, Philipp Gerstner, Vincent Heuveline

May 30, 2019

**Abstract**

The computation of the optimum of a dynamical or multi-period Optimal Power Flow problem assuming an Interior Point Method (IPM) leads to linear systems of equations whose size is proportional to the number of considered time steps. In this preprint we investigate a possibility to reduce the amount of time steps needed to be taken into account: Assuming that the power grid's dynamic is mainly determined by changes of the residual demand, we drop time steps in case it does not change much. Hence, the size of the linear systems can be reduced. We tested this method for the German Power Grid of the year 2023 and a synthetic 960 h profile. We were able to reduce the amount of time steps by 40% without changing the objective function's value significantly.

## 1 Introduction

A dynamical or multi-period Optimal Power Flow problem[1)] addresses the question how to improve the operation of a power grid with respect to a given objective in a given period of time. This question is answered by computing the dispatch of power generation among the available sources. From a mathematical point of view a multi-period Optimal Power Flow problem is a constrained optimisation problem with repeating constraints, see [4] and references therein.

The problem is treated discretely, i.e. the considered period of time, denoted $H$, called optimisation period and normally given in hours, is reduced to a set of time steps of equal length. The distance between time steps is called the temporal resolution and denoted $\Delta t_0$ . The time steps are addressed with the help of the index set $\mathcal{T}_0$. Hence, the cardinality of $\mathcal{T}_0$ equals the number of time steps considered, which we denote $N_T$.

The objective function $f_0$ of the resulting optimisation problem is defined by means of

$$f_0(x) = \sum_{i=1}^{N_T} \Delta t_0 f_t(x^{t_i})\,. \tag{1}$$

$f_t$ denotes an objective function, which models the costs per unit of time at any time. It includes, for example, the costs for a generator to provide a specified amount of power for a specified amount of time. $x = [x^{t_i}]_{i\in\mathcal{T}_0}$ is the complete vector of optimisation variables, which consists of the optimisation variables pertaining to the time steps $t_i$, $i \in \mathcal{T}_0$, i.e. $x^{t_i}$. It, for example, contains the power output of a generator at $t_i$, the voltage of a node at $t_i$ or the amount of energy stored at $t_i$ in a storage system.

Since a power grid's constraints, e.g. branch limits, voltage limits, and limits to power generation, should not be violated at any time, they are required to hold at every time step. Furthermore, there are some constraints, which may depend on the previous time step, e.g. constraints limiting the amount of energy to be stored in a storage system or constraints which avoid a too fast increase in power generation of conventional power plants [6].

[1)] A very concise and brief introduction to the Optimal Power Flow problem can be found in [2].

We summarise this in the following equations describing a general constrained optimisation problem:

$$\begin{aligned} \min_x \ & f(x) \text{ s.t.} \\ g^{t_i}(x^{t_i}) &= 0, && \text{for all } i \in \mathcal{T}_0\,, \\ g_{td}^{t_i}(x^{t_{i-1}}, x^{t_i}) &= 0, && \text{for all } i \in \mathcal{T}_0 \setminus \{1\}\,, \\ h^{t_i}(x^{t_i}) &\leq 0, && \text{for all } i \in \mathcal{T}_0\,, \\ h_{td}^{t_i}(x^{t_{i-1}}, x^{t_i}) &\leq 0, && \text{for all } i \in \mathcal{T}_0 \setminus \{1\}\,. \end{aligned} \tag{2}$$

The dimension of the vector of optimisation variables $x$ is $n^x$ and, furthermore, $x^{t_i} \in \mathbb{R}^{n^{x,t}}$. Thus, $n^x = N_T\, n^{x,t}$. The number of optimisation variables and the number of constraints are proportional to the number of time steps $N_T$.[3, 8]

Depending on the explicit form of the constraints and the objective function, there are many different algorithms to solve such an optimisation problem. We have chosen a Primal Dual Interior Point Method (*PDIPM*) [7]. First of all, it is an algorithm, which, in principal, is able to solve the most general class of optimisation problems, i.e. non-convex and non-linear optimisation problems. Secondly, its computational most intensive part is the solution of a linear system of equations, whose size is proportional to the number of considered time steps as we will show in the next paragraphs. The following description is in line with the implementation found in the MATLAB package MATPOWER [11].

As in the well-known case of unconstrained optimisation, there are necessary and sufficient conditions for a solution to be the optimum of a constrained optimisation problem. The necessary conditions are the Karush-Kuhn-Tucker conditions (*KKT* conditions), which we summarised in the following equation [3, 8]:[2)]

$$\mathbf{F}(x, s, \lambda, \mu) := \begin{pmatrix} \nabla_x \mathcal{L}_0(x, \lambda, \mu) \\ g(x) \\ h(x) + s \\ S\mu \end{pmatrix} = 0,\ s, \mu \geq 0\,. \tag{3}$$

$\mathcal{L}_0(x, \lambda, \mu)$ denotes the Lagrangian function, which is given by

$$\begin{aligned} \mathcal{L}_0 \colon \mathbb{R}^{n^x} \times \mathbb{R}^{n^\lambda} \times \mathbb{R}^{n^\mu} &\to \mathbb{R} \\ (x, \lambda, \mu) &\mapsto f(x) + \lambda^T g(x) + \mu^T h(x)\,, \end{aligned} \tag{5}$$

and $s \in \mathbb{R}^{n^\mu}$ is a vector containing slack variables and $S \in \mathbb{R}^{n^\mu \times n^\mu}$ denotes a diagonal matrix whose diagonal elements are $S_{kk} = s_k$. $\lambda$ and $\mu$ are called Lagrange multipliers. Their dimensions ($n^\lambda$ and $n^\mu$) equal the number of equality and inequality constraints respectively. Hence, they are proportional to the number of time steps considered, i.e. $n^\lambda = n^{\lambda,t} N_T$ and $n^\mu = n^{\mu,t} N_T$.

The PDIPM is an algorithm to find solutions to equation (3) and thus to find a point $(x^*, \lambda^*, \mu^*)$ which complies with the necessary conditions to be a local minimum of the multi-period Optimal Power Flow problem. 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 equation (3) are inside the feasible region and *not* on its surface, hence, the name Interior Point Method. More rigorously formulated, we find

$$\begin{aligned} \mathbf{F}_\gamma(x, s, \lambda, \mu) := \mathbf{F}(x, s, \lambda, \mu) - \begin{pmatrix} 0 & 0 & 0 & \gamma e \end{pmatrix}^T &= 0, \\ s, \mu &\geq 0, \end{aligned} \tag{7}$$

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, $e = (1, \ldots, 1) \in \mathbb{R}^{n^\mu}$. For details we refer the inclined reader to [7, 11].

[2)] For the sake of correctness, it is also necessary to assume an additional constraint qualification, e.g. the linear independence constraint qualification (*LICQ*) [7].

The application of Newton's method to equation (7) yields iterations in which the following linear systems of equations must be solved:

$$\nabla \mathbf{F}_{\gamma_k}(x^{(k)}, s^{(k)}, \lambda^{(k)}, \mu^{(k)}) \Delta^{(k)} = -\mathbf{F}_{\gamma_k}(x^{(k)}, s^{(k)}, \lambda^{(k)}, \mu^{(k)}) \,. \tag{8}$$

Henceforth we omit the iteration index $k$. Assuming that $s, \mu > 0$, the Newton system (8) can be transformed into a reduced, symmetric saddle point form

$$\underbrace{\begin{pmatrix} \nabla^2_{xx}\mathcal{L}_0(x,\lambda,\mu) + (\nabla_x h(x))^T \Sigma (\nabla_x h(x)) & (\nabla_x g(x))^T \\ \nabla_x g(x) & 0 \end{pmatrix}}_{=:\, A(x,s,\lambda,\mu)} \underbrace{\begin{pmatrix} \Delta x \\ \Delta \lambda \end{pmatrix}}_{=:\Delta_R} = \underbrace{-\begin{pmatrix} r_x \\ r_\lambda \end{pmatrix}}_{=:b} \tag{9}$$

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

$$\begin{aligned} r_x &= \nabla_x \mathcal{L}_0(x,\lambda,\mu) + (\nabla_x h(x))^T Z^{-1} \left(\gamma e + \operatorname{diag}(\mu) h(x)\right) \,, \\ r_\lambda &= g(x) \,. \end{aligned}$$

Please note the different meanings of the $\nabla$-operator. If it is applied to a scalar function, it is the gradient of this function. Applied to a vector function, it denotes the Jacobian of that function. Furthermore, if there are no variables specified in its subscript, the derivative of the (vector) function has to be computed with respect to all its arguments. $\nabla^2_{xx}\mathcal{L}_0$ denotes the Hessian matrix of the scalar function $\mathcal{L}_0$.

A look at the *KKT matrix* A, defined in equation (9), shows that its size is proportional to the number of time steps: First of all, the size of the Hessian matrix is by definition proportional to the number of optimisation variables $x$, which in turn are proportional to the number of time steps. Secondly, the number of rows of the Jacobian $\nabla_x g(x)$ equals the number of equality constraints and, consequently, it is also proportional to the number of time steps. Finally, $A \in \mathbb{R}^{n \times n}$ and $n = n^x + n^\lambda = (n^{x,t} + n^{\lambda,t}) N_T$. Since the computationally most intensive part of the PDIPM algorithm is the solution of equation (9), the reduction of its size by means of a time step reduction method may strongly improve its time to solution.

Before we start describing the proposed time step reduction method, we have to introduce the concept of a load profile [9]. The optimisation of the operation of a power grid heavily depends on the power demanded by different kinds of consumers. A load profile depicts the evolution of this demand with time. More importantly it is an integral part of the formulation of the constraints of the multi-period Optimal Power Flow problem: As in the case of a small electric circuit, in a power grid current and voltage are related by Kirchhoff's laws and Ohm's law. For example, if someone plugs in a hairdryer in Europe (i.e. assuming a nominal voltage of ∼230 V), a current depending on the hairdryer's resistance must flow. Thus, the physical restrictions described by the above laws depend on the demand, have to hold strictly and are not allowed to be violated at any $t_i$. In equation (2) they are represented by the function $g(x^{t_i})$ and in the literature they are referred to as Power Flow equations. An explicit formulation of these equations can be found in [2, 3, 10]. Please note that we use the terminology "load" and "demand" interchangeably.

# 2 Methodology

Our time step reduction method is based on the assumption that if the load's change is small, the change in the optimisation variables is almost linear and, thus, in this case it should be possible to drop time steps, i.e. to decrease the temporal resolution, and to compute the optimisation variables' values at the left out time steps by linear interpolation. In contrast, if the load changes rapidly, the non-linear character of the optimisation problem (2) becomes dominant and forbids linear interpolation and, hence, to skip time steps, i.e. a high temporal resolution is needed to map the power grid's dynamic.

First of all, we approximately compute the load's rate of change at node $k$ of the power grid and at the time steps $t_i$, $i \in \mathcal{T}_0 = \{1, \ldots, N_T\} \setminus \{N_T\}$, with the help of the difference quotient:

$$\dot{P}^{L,k}(t_i) = \left.\frac{dP^{L,k}}{dt}\right|_{t_i} \approx \frac{P^{L,k}_{t_{i+1}} - P^{L,k}_{t_i}}{t_{i+1} - t_i} \,. \tag{10}$$

If the rate of change is small, the load's change will be small, and, accordingly, we propose to drop some time steps. For this purpose, we segment the optimisation period $H$ accordingly and choose different temporal resolutions $\Delta t$ for the individual sections.

## 2.1 Measuring the change in residual load

In order to adjust the temporal resolution $\Delta t$ based on the load's rate of change, it is necessary to tell small rate changes apart from large ones. On a mathematical level, this can be done by defining a measure, i.e. a function which maps its arguments onto the interval $[0, 1]$. In a first step towards this definition, we compute the mean value of the rate of change at time step $t_i$:

$$\dot{P}^{L,MV}(t_i) = \frac{1}{N_B} \sum_{k=1}^{N_B} \dot{P}^{L,k}(t_i)\,. \tag{11}$$

$N_B$ denotes the number of nodes, which belong to the investigated power grid. In the second step, we divide these mean values by their maximum $\dot{P}^{L,MV}_{max}$ and, thus, obtain the definition of a measure of the rate of change at time step $t_i$:

$$\begin{aligned} \phi\colon \mathcal{T}_0 &\to [0,1] \\ i &\mapsto \frac{\dot{P}^{L,MV}(t_i)}{\dot{P}^{L,MV}_{max}}\,. \end{aligned} \tag{12}$$

With this definition at hand, we may say that the load's rate of change is big if the value of $\phi$ is close to one.

## 2.2 Different $\Delta t$ for different changes

Moreover, we need to relate the rate of change's measure to different temporal resolutions. To this end, we define an additional step function $\Phi$, which maps the rate of change's size of every time step $t_i$ to a temporal resolution $\Delta t$. The possible temporal resolutions are obtained by deciding on the maximal temporal resolution $\Delta t_{max}$. Since we are only dropping time steps, it is clear from the outset, that possible temporal resolutions can only be integer multiples of the load profile's original temporal resolution $\Delta t_0$. Hence, the number of possible temporal resolutions $N_{\Delta t}$ is $\Delta t_{max}/\Delta t_0$. Furthermore, the definition of $\Phi$ requires, that we split the interval $(0, 1]$ into $N_{\Delta t}$ parts:

$$(0,1] = \bigcup_{l=0}^{N_{\Delta t}-1} (l\frac{1}{N_{\Delta t}}, (l+1)\frac{1}{N_{\Delta t}}]\,.$$

We define the function $\Phi$ as

$$\begin{aligned} \Phi\colon [0,1] &\to \mathbb{R} \\ x &\mapsto (l+1)\Delta t_0 \;\text{ if } x \in (\frac{l}{N_{\Delta t}}, \frac{l+1}{N_{\Delta t}}]\,. \end{aligned} \tag{13}$$

In Fig. 1 we plotted one example $\Phi$ function to illustrate the way it relates different rates of change to different temporal resolutions $\Delta t$.

## 2.3 Segmenting the optimisation period

Due to the measure $\phi$ and the step function $\Phi$ we are able to assign a $\Delta t$ to every time step $t_i$ of the optimisation period $H$. Since the aggregated load does not change abruptly, i.e. the load profile is assumed to be continuously differentiable, it is likely that a time step $t_i$ is surrounded by a set of time steps whose values of $\phi$ do not differ much and to whom the step function $\Phi$ consequently assigns the same $\Delta t$. Thus, it is possible to segment the optimisation period $H$ into $N_A$ sections by grouping the time steps with an equal $\Delta t$. We write $H = \sum_{k=1}^{N_A} H_k$ and denote the temporal resolution, which corresponds to section $H_k$, $\Delta t_k$.

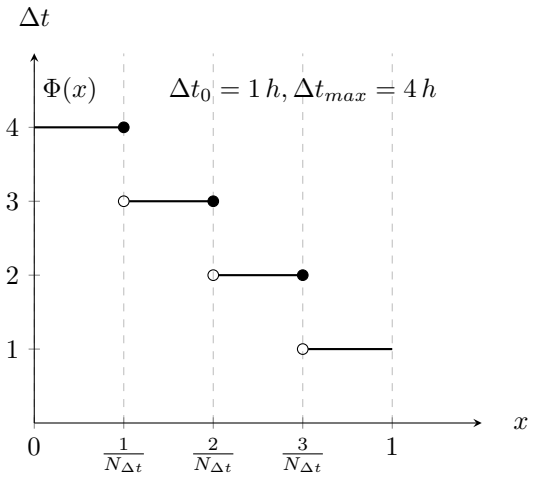



**Figure 1:** Step function $\Phi(x)$. Smaller changes $\phi(i) \lesssim 0.5$ are related to bigger temporal resolutions $\Delta t$.

The section's different temporal resolutions $\Delta t_k$ reduce the original set of indices $\mathcal{T}_0$, used to denote the time steps and to formulate the original optimisation problem, to the subset

$$\mathcal{T} = \bigcup_{k=1}^{N_A} \mathcal{T}_k \subset \mathcal{T}_0 \,.$$

Where $\mathcal{T}_k = \{i_k + m(\Delta t_k/\Delta t_0) \mid m \in \{0, \ldots, (H_k/\Delta t_k) - 1\}\}$ with

$$i_k = \sum_{l=1}^{k-1} \frac{H_l}{\Delta t_0} + \frac{\Delta t_k}{\Delta t_0} \text{ for } k > 1 \,.$$

For the sake of clarity, we depicted in Fig. 2 a section $H_k$ and a subset $\mathcal{T}_k$ of the optimisation period $H$.

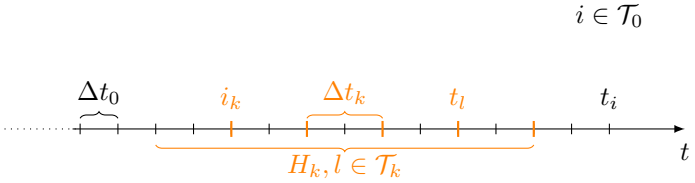


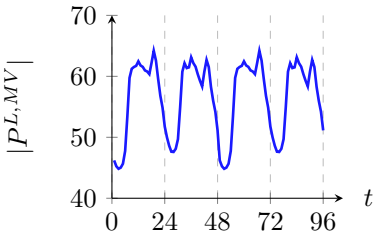

**Figure 2:** Segmentation of the optimisation period $H$ into sections $H_k$.

It is important to note, that the length of an optimisation section $H_k$ is not always an integral multiple of its temporal resolution $\Delta t_k$. In this case we add the "non-fitting" time steps to the next section. If there are time steps left at the end of the optimisation period, we replenish it with an additional section of temporal resolution $\Delta t_0$.

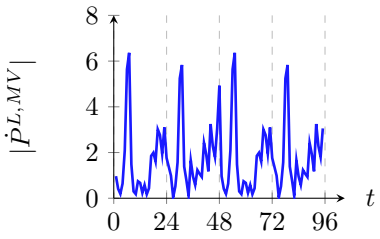
## 2.4 Temporal resolution profile. An example.

We would like to demonstrate the explained time step reduction method by considering a concrete example. We take a look at the power grid and the synthetic 960 hour load profile referred to in Section 3. In Fig. 3a we plotted the first ninety-six hours of the arithmetic mean of the load. The load profile repeats itself after forty-eight hours. In Fig. 3b the load's rate of change is plotted. A brief check of the minima shows that the difference quotient used to compute it is a good approximation. Fig. 3c shows the result of measuring the load's rate of change with the help of $\phi$ and plugging its values into the step function $\Phi$: If

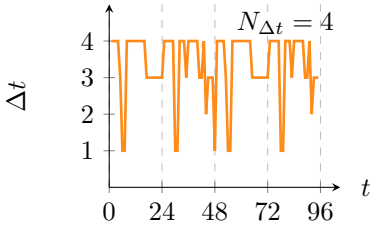
the load does not change drastically in an optimisation section $H_k$, the maximum $\Delta t_{max}$ is chosen. The values of the step function $\Phi$ form the basis of the segmentation of the optimisation period, by grouping its values as described above we obtain the *temporal resolution profile* shown in Fig. 3d[3)].



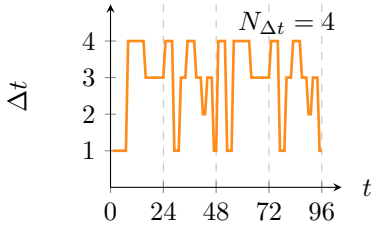

**(a)** Arithmetic mean of the load.




**(b)** Mean of the load's rate of change.




**(c)** Values of the function $\Phi$.




**(d)** Temporal resolution profile.

## 2.5 Interpolation operator

After having applied the time step reduction method, we have the reduced set of time step indices $\mathcal{T}$ to formulate the multi-period Optimal Power Flow problem. It is solved using an Interior Point Method. Thus, after an unknown number of iterations the computed solution is an accord with the Karush-Kuhn-Tucker conditions. This solution is used to determine the values of the optimisation variables at the time steps we have dropped. This is done via linear interpolation. The definition of an according linear interpolation operator $\mathcal{I}_\mathcal{T}$ consists of different parts for different sections of the optimisation period, because it depends on the temporal resolution of the respective section. Before defining one part of it for a specific section of the optimisation period, we remind the reader that the optimisation variables are sorted by time steps: $\bar{x} = [\bar{x}^{t_i}]_{i \in \mathcal{T}}$. In the section $H_k$ the corresponding part of the interpolation operator is defined as:

$$\tilde{x}^{t_{a_k+mw_k+l}} = l \frac{\Delta t_0}{\Delta t_k} \left( \bar{x}^{t_{a_k+(m+1)w_k}} - \bar{x}^{t_{a_k+mw_k}} \right) + \bar{x}^{t_{a_k+mw_k}} . \tag{14}$$

The following definitions and values are used:

$$\begin{aligned} w_k &= \frac{\Delta t_k}{\Delta t_0} , \\ a_k &= \sum_{i=1}^{k-1} \frac{H_i}{\Delta t_0} \qquad (\text{if } k = 0, \text{ then } a_0 = 0) , \\ l &\in \{1, \ldots, \Delta t_k / \Delta t_0 - 1\} , \\ m &\in \{0, \ldots, (H_k / \Delta t_k) - 1\} . \end{aligned}$$

The complete interpolation operator $\mathcal{I}_\mathcal{T}$ is defined as the linear interpolation of all optimisation variables via the equation (14). We abbreviate this operation to $\tilde{x} = \mathcal{I}_\mathcal{T}(\bar{x})$. The dual variables $\tilde{\lambda}$ and $\tilde{\mu}$ are computed analogously, i.e. $\tilde{\lambda} = \mathcal{I}_\mathcal{T}(\bar{\lambda})$ and $\tilde{\mu} = \mathcal{I}_\mathcal{T}(\bar{\mu})$.

[3)] The observant reader may will recognise that the first section $H_1$ should have temporal resolution $\Delta t_1 = 4$. The reason it does not is that we have not defined an extrapolation operator, which would be needed to compute the values of the first time steps. We just set the temporal resolution of $H_1$ to $\Delta t_0$ to avoid extrapolation.

## 2.6 Estimating the error

Since dropping time steps will cause an error in the optimisation problem's objective function, it is important to estimate this error in the worst case. To find out about this, we construct an error estimator, which is supposed to present an approximation of the absolute value of the error. We take advantage of the interpolated optimisation variables $\tilde{x}$ and introduce $x^*$ to denote the solution of the multi-period Optimal Power Flow problem without time step reduction, which, henceforth, will be referred to as the reference problem.

The error of the objective function is given by:

$$f(\bar{x}) - f_0(x^*) = \underbrace{f(\bar{x}) - f_0(\tilde{x})}_{=:\Delta_1} + \underbrace{f_0(\tilde{x}) - f_0(x^*)}_{=:\Delta_2} . \tag{15}$$

Since $\bar{x}$ and $\tilde{x}$ are known after the IPM finished, $\Delta_1$ can be computed directly. In the following we will derive a heuristic expression for the computation of the error $\Delta_2$:

$$\begin{aligned}
\Delta_2 &= f_0(\tilde{x}) - f_0(x^*) \\
&= \mathcal{L}_0(\tilde{x}, \tilde{\lambda}, \tilde{\mu}) - \tilde{\lambda}^T g(\tilde{x}) - \tilde{\mu}^T h(\tilde{x}) - \mathcal{L}_0(x^*, \lambda^*, \mu^*) \\
&= \int_0^1 l'(s)\, ds - \tilde{\lambda}^T g(\tilde{x}) - \tilde{\mu}^T h(\tilde{x}) \qquad (16) \\
&= \frac{1}{2}\left(l'(1) + l'(0)\right) + \mathcal{O}(\|e\|^3) - \tilde{\lambda}^T g(\tilde{x}) - \tilde{\mu}^T h(\tilde{x}) \qquad (17)
\end{aligned}$$

where $l(s) := \mathcal{L}_0(x^* + se_x, \lambda^* + se_\lambda, \mu^* + se_\mu)$. Moreover,

$$e_x := \tilde{x} - x^*,\ e_\lambda := \tilde{\lambda} - \lambda^*,\ e_\mu := \tilde{\mu} - \mu^* \text{ and } e := \begin{pmatrix} e_x & e_\lambda & e_\mu \end{pmatrix}^T .$$

The integral in equation (16) was approximated by applying the trapezoidal rule to it. Computing the derivative $l'(s)$ yields

$$\begin{aligned}
l'(s) &= \nabla_x \mathcal{L}_0(\dots)e_x + \nabla_\lambda \mathcal{L}_0(\dots)e_\lambda + \nabla_\mu \mathcal{L}_0(\dots)e_\mu \\
&= \nabla_x \mathcal{L}_0(\dots)e_x + g^T(\dots)e_\lambda + h^T(\dots)e_\mu ,
\end{aligned}$$

and plugging into it the limits of the integral results in

$$\begin{aligned}
l'(1) &= \nabla_x \mathcal{L}_0(\tilde{x}, \tilde{\lambda}, \tilde{\mu})e_x + g^T(\tilde{x})e_\lambda + h^T(\tilde{x})e_\mu , \\
l'(0) &= \nabla_x \mathcal{L}_0(x^*, \lambda^*, \mu^*)e_x + g^T(x^*)e_\lambda + h^T(x^*)e_\mu = h^T(x^*)e_\mu = h^T(x^*)\tilde{\mu} .
\end{aligned}$$

With the evaluated derivatives at hand Equation (17) becomes

$$\Delta_2 = \frac{1}{2}\nabla_x \mathcal{L}_0(\tilde{x}, \tilde{\lambda}, \tilde{\mu}) \cdot (\tilde{x} - x^*) - \frac{1}{2} g(\tilde{x})^T(\tilde{\lambda} + \lambda^*) - \frac{1}{2} h(\tilde{x})^T(\tilde{\mu} + \mu^*) + \frac{1}{2} h(x^*)^T\tilde{\mu} + \mathcal{O}(\|e\|^3) . \tag{18}$$

Since the exact solution of the reference problem $(x^*, \lambda^*, \mu^*)$ is needed and because of the error term $\mathcal{O}(\|e\|^3)$, it is not possible to actual compute $\Delta_2$. Further modifications are needed:

$$\Delta_2 \approx \frac{1}{2}\nabla_x \mathcal{L}_0(\tilde{x}, \tilde{\lambda}, \tilde{\mu}) \cdot (\tilde{x} - \tilde{x}_{Spline}) - g(\tilde{x})^T\tilde{\lambda} - \frac{1}{2} h(\tilde{x})^T\tilde{\mu} \tag{19}$$

We dropped the error term $\mathcal{O}(\|e\|^3)$ and replaced the solution of the reference problem $x^*, \lambda^*, \mu^*$ with its interpolated counterpart $\tilde{x}, \tilde{\lambda}, \tilde{\mu}$. But note, to avoid that the first term of equation (18) vanishes, we had to replace the first appearance of $x^*$ with $\tilde{x}_{Spline}$. To compute it, we replaced the linear interpolation with a Spline interpolation. These modifications are motivated by an analogous construction of an error estimator in the field of Partial Differential Equations, which, for example, can be found in [1]. Numerical experiments showed, that $\Delta_2$ was bigger than $f(\bar{x})$ and, consequently, not suited to provide a good error estimation. Thus, we attempted to drop the last term of equation (19), which yielded reasonable estimations. Hence, we settled on the following formula for $\Delta_2$:

$$\Delta_2 \approx \frac{1}{2}\nabla_x \mathcal{L}_0(\tilde{x}, \tilde{\lambda}, \tilde{\mu}) \cdot (\tilde{x} - \tilde{x}_{Spline}) - g(\tilde{x})^T\tilde{\lambda} . \tag{20}$$

We would like to emphasise that the above construction of an error estimator is based on a heuristic and the numerical experiments presented in Section 3 show that there might be a better way to compute an estimator.

# 3 Results

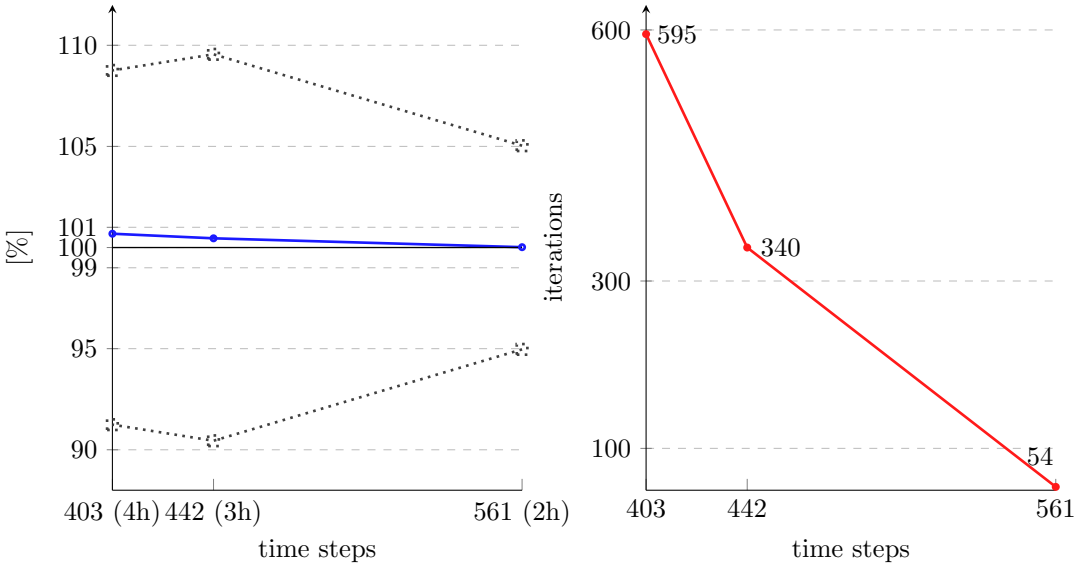
We implemented the proposed time step reduction method in an expanded version of the free and open-source software MATPOWER. MATPOWER already contains the Primal Dual IPM, but it needed to be expanded by a mechanism, which is able to work with multiple time steps and different temporal resolutions to set up the KKT matrix for a multi-period Optimal Power Flow problem.[4] We developed an Octave function, which creates temporal resolution profiles for different $\Delta t_{max}$. Furthermore, we implemented the interpolation operator by means of Octave's `interp1` function and provided an additional function to compute an estimation of the objective function's error.

We tested the time step reduction method on a power grid, which 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 [5].

At the outset we had a synthetic $48h$ load profile, which we concatenated to obtain a test case comprising 48 days with a temporal resolution of $\Delta t_0 = 1h$, i.e. $N_T = 960$ time steps. This causes the linear system (9), which has to be solved in every PDIPM iteration, to be of size $n = n^x + n^\lambda = 6.168 \cdot 10^6$.

In a first step, we solved the reference problem, i.e. we solved the multi-period Optimisation problem considering the complete optimisation period $H = 960h$ without dropping any time steps.[5] In this way we obtained the reference value $f_0(x^*)$ of the original objective function. In a next step, we tested how many time steps can be dropped without causing too big errors in the objective function. Due to the construction of the proposed time step reduction method, the more time steps will be ignored, the bigger $\Delta t_{max}$ is chosen. We tested the following values for $\Delta t_{max}$: $2h, 3h$ and $4h$.

Fig. 4a shows on the $x$-axis the number of time steps, which had to be taken into account after we had applied the time step reduction method. For example, $\Delta t_{max} = 3\Delta t_0 = 3h$ resulted in 442 time steps. The dark blue line represents the deviation of the objective function from the reference value in percentage. The dashed grey lines depict the estimated error of the objective function, which we computed with the help of equation (20). The numerical experiments showed, that the choice $\Delta t_{max} = 2\Delta t_0$ yields,




**(a)** Deviation of the objective function

**(b)** Increase in IPM iterations

from a practical point of view, the most promising results. Although we were able to further decrease the number of time steps by setting $\Delta t_{max} = 3\Delta t_0$ or $\Delta t_{max} = 4\Delta t_0$ without deviating too much from the reference value of the objective function, we observed a destabilisation of the Interior Point Method. To

[4] We would like to express our gratefulness to Nico Meyer-Hübner, who wrote the extension of MATPOWER.

[5] Since the size of the KKT matrix $A$ was very big, we solved the optimisation problems on a cluster taking advantage of a parallelization method, which we had developed previously. For more details, we refer the reader to [3, 8].

illustrate this point, we plotted in Fig. 4b the number of IPM iterations against the number of time steps. It took the PDIPM algorithm 41 iterations to solve the reference problem. A look at this plot shows a dramatic increase in IPM iterations in comparison to the reference. Moreover, convergence of the PDIPM algorithm could only be obtained by easing the convergence criteria. It's tempting to assume that the reduced size of the KKT matrix $A$ and the according decrease in the solution time compensates for the increase in iterations, but that's not the case for choices $\Delta t_{max} > 2\Delta t_0$. In contrast, in case $\Delta t_{max}$ is set to $2\Delta t_0$, we observed only a slight increase in IPM iterations and a negligible deviation from the objective function's reference value. At the same time the number of time steps could be reduced to 561, i.e. by around 40 percent. This led to a large decrease in the solution time: in the reference problem an iteration took on average $441.05s$, which is much more than the $200.36s$ needed in the $2\Delta t_0$ case.

In an attempt to further benchmark the proposed time step reduction method, we dropped every second time step and solved the multi-period Optimal Power Flow problem again. Dropping every second time step reduced the number of time steps to 486[6)]. The IPM needed 58 iterations to converge and every iteration took on average $190.61s$. Hence, it took longer to solve the optimisation problem, even though less time steps were considered. Furthermore, the deviation from the reference value of the objective function was 25 times bigger in comparison with the $2\Delta t_0$ case.

# 4 Conclusion and Outlook

In case the parameter $\Delta t_{max}$ was set to $2\Delta t_0$, the proposed time step reduction method yielded good results. The amount of time steps could be reduced by $\sim$40% while the objective function's deviation from its reference value was only 0.000176%. Moreover, the PDIPM algorithm was almost twice as fast. And in comparison with a naïve reduction of time steps, i.e. dropping every second time step, it performed well.

Unfortunately, if $\Delta t_{max} > 2\Delta t_0$, the time step reduction caused the IPM to become unstable. It is still unclear why the number of iterations increases so dramatically and further research on it is required.

Furthermore, the proposed error estimator over estimates the objective function's error. To find a better heuristic further investigation is needed.

This contribution represents a preliminary work and the proposed approach needs clearly further testing with more power grids and load profiles, especially load profiles which do not repeat themselves and are not synthetic. This is the next step.

# 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] R. Becker, M. Braack, and R. Rannacher. "Numerical simulation of laminar flames at low Mach number by adaptive finite elements". In: *Combustion Theory and Modelling* 3.3 (1999), pp. 503–534. DOI: 10.1088/1364-7830/3/3/305. eprint: https://doi.org/10.1088/1364-7830/3/3/305. URL: https://doi.org/10.1088/1364-7830/3/3/305.

[2] Stephen Frank and Steffen Rebennack. "An introduction to optimal power flow: Theory, formulation, and examples". In: *IIE Transactions* 48.12 (2016), pp. 1172–1197. DOI: 10.1080/0740817X.2016.1189626.

[6)] This value deviates slightly from the expected 480 time steps because of a missing definition of an extrapolation operator, c.f. with footnote 3).

[3] Philipp Gerstner et al. *A Domain Decomposition Approach for Solving Dynamic Optimal Power Flow Problems in Parallel with Application to the German Transmission Grid.* Tech. rep. 01. 37.06.01; LK 01. Universität Heidelberg, Heidelberg, 2016. 21 pp. URL: https://journals.ub.uni-heidelberg.de/index.php/emcl-pp/article/download/33784/27475.

[4] Nico Meyer-Hübner, M. Suriyah, and T. Leibfried. "On efficient computation of time constrained optimal power flow in rectangular form". In: *PowerTech, 2015 IEEE Eindhoven.* 2015, pp. 1–6. DOI: 10.1109/PTC.2015.7232378.

[5] Nico Meyer-Hübner et al. "Time constrained optimal power flow calculations on the German power grid". In: *International ETG Congress 2015; Die Energiewende - Blueprints for the new energy age.* 2015, pp. 1–7.

[6] Nico Meyer-Hübner et al. "Dynamic optimal power flow in AC networks with multi-terminal HVDC and energy storage". In: *2016 IEEE Innovative Smart Grid Technologies - Asia (ISGT-Asia)* (2016). DOI: 10.1109/isgt-asia.2016.7796402. URL: http://dx.doi.org/10.1109/ISGT-Asia.2016.7796402.

[7] Jorge Nocedal and Stephen J. Wright. *Numerical optimization. 2nd ed.* English. 2nd ed. New York, NY: Springer, 2006, pp. xxii + 664. ISBN: 0-387-30303-0/hbk.

[8] Nils Schween et al. "A Domain Decomposition Approach to solve Dynamic Optimal Power Flow Problems in Parallel". In: *Proceedings of the second International Symposium on Energy System Optimization.* planned to be publish in 2019.

[9] Alexandra Von Meier. *Electric power systems: a conceptual introduction.* John Wiley & Sons, 2006.

[10] Jizhong Zhu. *Optimization of power system operation.* Vol. 47. John Wiley & Sons, 2015.

[11] Ray D. Zimmerman and Carlos E. Murillo-Sánchez. *Matpower 6.0 User's Manual.* 2016.

# Preprint Series of the Engineering Mathematics and Computing Lab

recent issues

No. 2018-02 Simon Gawlok, Vincent Heuveline: Nested Schur-Complement Solver for a Low-Mach Number Model: Application to a Cyclone-Cyclone Interaction

No. 2018-01 David John, Michael Schick, Vincent Heuveline: Learning model discrepancy of an electric motor with Bayesian inference

No. 2017-06 Simon Gawlok, Philipp Gerstner, Saskia Haupt, Vincent Heuveline, Jonas Kratzke, Philipp Lösel, Katrin Mang, Maraike Schmidtobreick, Nicolai Schoch, Nils Schween, Jonathan Schwegler, Chen Song, Marin Wlotzka: HiFlow$^3$ Technical Report on Release 2.0

No. 2017-05 Nicolai Schoch, Vincent Heuveline: Towards an Intelligent Framework for Personalized Simulation-enhanced Surgery Assistance: Linking a Simulation Ontology to a Reinforcement Learning Algorithm for Calibration of Numerical Simulations

No. 2017-04 Martin Wlotzka, Thierry Morel, Andrea Piacentini, Vincent Heuveline: New features for advanced dynamic parallel communication routines in OpenPALM: Algorithms and documentation

No. 2017-03 Martin Wlotzka, Vincent Heuveline: An energy-efficient parallel multigrid method for multi-core CPU platforms and HPC clusters

No. 2017-02 Thomas Loderer, Vincent Heuveline: New sparsing approach for real-time simulations of stiff models on electronic control units

No. 2017-01 Chen Song, Markus Stoll, Kristina Giske, Rolf Bendl, Vincent Heuveline: Sparse Grids for quantifying motion uncertainties in biomechanical models of radiotherapy patients

No. 2016-02 Jonas Kratzke, Vincent Heuveline: An analytically solvable benchmark problem for fluid-structure interaction with uncertain parameters

No. 2016-01 Philipp Gerstner, Michael Schick, Vincent Heuveline, Nico Meyer-Hübner, Michael Suriyah, Thomas Leibfried, Viktor Slednev, Wolf Fichtner, Valentin Bertsch: A Domain Decomposition Approach for Solving Dynamic Optimal Power Flow Problems in Parallel with Application to the German Transmission Grid

No. 2015-04 Philipp Gerstner, Vincent Heuveline, Michael Schick : A Multilevel Domain Decomposition approach for solving time constrained Optimal Power Flow problems

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)

Preprint Series of the Engineering Mathematics and Computing Lab (EMCL)


www.emcl.iwr.uni-heidelberg.de