Hahne et al. *Journal of Mathematics in Industry* (2023) 13:6
https://doi.org/10.1186/s13362-023-00134-5

Journal of Mathematics in Industry
a SpringerOpen Journal

RESEARCH **Open Access**

Parallel-in-time optimization of induction motors

Jens Hahne[1*], Björn Polenz[2], Iryna Kulchytska-Ruchka[3], Stephanie Friedhoff[1], Stefan Ulbrich[2] and Sebastian Schöps[3]

*Correspondence:
jens.hahne@math.uni-wuppertal.de
[1]Department of Mathematics, Bergische Universität Wuppertal, Gaußstr. 20, 42119, Wuppertal, Germany
Full list of author information is available at the end of the article

Abstract

Parallel-in-time (PinT) methods were developed to accelerate time-domain solution of evolutionary problems using modern parallel computer architectures. In this paper we incorporate one of the efficient PinT approaches, in particular, the asynchronous truncated multigrid-reduction-in-time algorithm, into a bound constrained optimization procedure applied to an induction machine. Calculation of an optimal motor geometry with respect to its efficiency in the steady state is thus parallelized at each iteration of the optimization algorithm. As a result, a more efficient motor model is obtained about 11 times faster compared to optimization using the standard sequential time stepping.

Keywords: Parallel-in-time; Optimization; Electric motors

1 Introduction

Modern corporate design of electromagnetic devices such as electric motors is based on computer-aided optimization of several key performance indicators such as, e.g., output power, losses, overall efficiency, etc. In this way, customer requirements can be incorporated already within the early design stages before a physical prototype is manufactured. In order to create an optimal digital prototype, one typically has to perform transient simulations of various multi-physical effects (e.g., magnetic and mechanical) in the time domain. Such calculations are often very time consuming due to the need to resolve the arising high-frequency field components, which despite their small amplitudes, can lead to big losses. Parallel-in-time (PinT) methods such as Parareal [1] or multigrid-reduction-in-time (MGRIT) [2] are powerful tools for an acceleration of these development stages, as it is shown for an induction motor in [3] and [4], respectively. The MGRIT algorithm is based on multilevel reduction [5] principles applied to the time dimension. In this process, time integration is applied in parallel to temporal subdomains at the finer levels and serially over the entire time interval at the coarsest level. One of the key advantages of the algorithm is its non-intrusive nature, which allows existing time integrators to be reused and embedded in a time-parallel framework. A variant of the MGRIT algorithm, the asynchronous truncated multigrid-reduction-in-time algorithm (AT-MGRIT) [6], uses multiple inde-

pendent overlapping local coarse grids at the coarsest level to increase the parallelism of the MGRIT algorithm.

In this paper, we optimize the geometry of a three-phase squirrel-cage induction motor with respect to its efficiency in the steady state. For this, a derivative-free algorithm BOBYQA for bound constrained optimization is used, which is based on quadratic interpolation [7, 8]. Each iteration of the optimization procedure includes the PinT time-domain computation with AT-MGRIT until the steady state of the machine is obtained. The mechanical power and the Joule losses are calculated at the post-processing step and used for the construction of the objective function, which is optimized in terms of the rotor bars size, i.e., their width and height.

This paper is organized as follows. In Sect. 2 we provide a mathematical partial differential equation (PDE) model of the electromagnetic phenomena taking place in an induction motor and discretize it using space- and time-domain numerical methods. Section 3 formulates an optimization problem in terms of the motor's efficiency in the steady state and describes the overall optimization procedure based on a time-domain solution. In Sect. 4 we describe the AT-MGRIT method for accelerated PinT solution, which can be used at each iteration of the optimization algorithm. Application of the proposed methodology to a four-pole induction machine is illustrated in Sect. 5. Finally, the paper is completed with a conclusion in Sect. 6.

2 Simulation of induction machines

Electric motors are electric devices that transform electrical energy into mechanical energy. In this paper we consider a specific type of motors called three-phase induction motors, which are among the most widespread electric motors within the power range under 500 kW.

Three-phase induction motors are supplied with a three-phase voltage source v_k given by

$$v_k(t) = \hat{U}\sin\big(2\pi ft - (k-1)\cdot 2\pi/3\big), \quad k = 1,2,3, \tag{1}$$

illustrated in Fig. 1, where f is the frequency and $\hat{U}$ is the amplitude (or peak value) of voltage. The windings carrying three-phase voltage are placed into the stator slots in the outer part of the motor called stator as depicted in Fig. 2 for a two-dimensional (2D) induction motor model.

The inner part of the considered induction motor is a squirrel-cage rotor, which consists of solid conductor bars, placed into the rotor slots and connected at both ends by the conducting end rings. The stator and rotor are separated by the air gap, which is traversed by the magnetic flux and allows for a current flow in the rotor bars. As a result, the *Lorentz force* acts on the squirrel cage and makes the rotor rotate with a mechanical speed ω_{mech}. Finally, the produced electromagnetic torque T_{EM} is transferred to the mechanical load through a shaft placed in the very inner part of the motor.

Solution of a dynamical system like an electric motor excited with a periodic signal (1) typically consists of a transient part, followed by a (periodic) steady state taking place once the transients are eventually damped out. The steady-state operating characteristics such as rotational speed and torque are important design criteria, especially during initial design stages [9]. Figure 3 illustrates the time-domain torque evolution of an induction motor

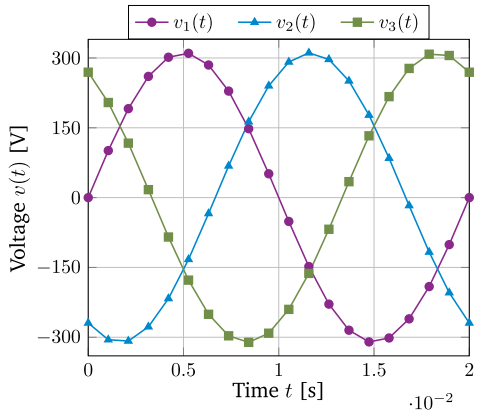

Figure 1 Three-phase voltage of frequency $f = 50$ Hz and amplitude $\hat{U} = 311.1$ V

Figure 2 Cross-section of a four-pole squirrel-cage induction machine [9, 10]. The geometry is given by the "im_3kw" model at onelab.info

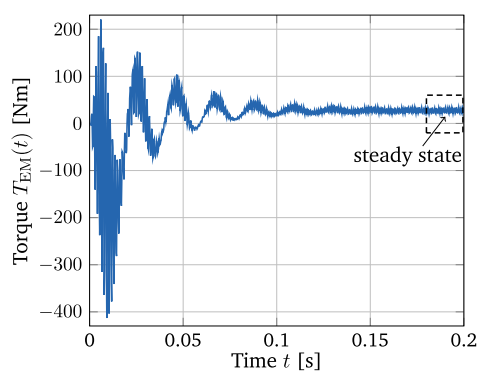

Figure 3 Torque produced in a linear "im_3kw" induction machine model excited with a three-phase voltage supply (1) of frequency $f = 50$ Hz and amplitude $\hat{U} = 311.1$ V. The simulation is performed using the `GetDP` library [11]

rotating at the constant speed $\omega_{\mathrm{mech}} = 1420$ rpm. There, the steady state is reached at the tenth period of length $T = 0.02$ s, i.e., on the interval $[(q-1)T, qT] = [0.18, 0.2]$ s, with $q = 10$. In this section we provide the theoretical basis and numerical approaches for the time-domain simulation of electromagnetic energy converters.

2.1 Mathematical model

The electromagnetic fields in induction motors are commonly modeled by a magneto-quasistatic (MQS) approximation of Maxwell's equations [12], which is suitable for low-frequency applications provided the wavelength is much larger than the problem size [13]. The MQS setting neglects the displacement currents as they are outweighed by the magnetic effects and the Joule losses. One can then derive for the *magnetic vector potential (MVP)* $\mathbf{A}$ the *eddy current problem*

$$\sigma \frac{\partial \mathbf{A}}{\partial t} + \operatorname{curl}(\nu \operatorname{curl} \mathbf{A}) = \mathbf{J}_\mathrm{s} \quad \text{in } \Omega \times (0, T_\mathrm{end}], \tag{2}$$

with $\Omega \subset \mathbb{R}^3$ denoting an open, bounded, simply connected domain with Lipschitz boundary and $T_\mathrm{end} > 0$. Here $\sigma = \sigma(\mathbf{x}) \geq 0$ denotes the *electric conductivity*, $\nu = \nu(\mathbf{x}, |\operatorname{curl} \mathbf{A}|) > 0$ is the *magnetic reluctivity*, and $\mathbf{J}_\mathrm{s}$ is the *source current density* defined by

$$\mathbf{J}_\mathrm{s}(\mathbf{x}, t) = \sum_{k=1}^{3} \boldsymbol{\chi}_k(\mathbf{x}) i_k(t), \tag{3}$$

with each $\boldsymbol{\chi}_k(\mathbf{x}) \in \mathbb{R}^3$ denoting a *winding function* [14], which spatially distributes the current $i_k(t) \in \mathbb{R}$ flowing through the kth stranded conductor. The three-phase input voltage (1) is coupled to the eddy current equation (2) via the relation [15, Sect. 6]

$$v_k(t) = R_k i_k(t) + \int_\Omega \boldsymbol{\chi}_k(\mathbf{x}) \cdot \frac{\partial \mathbf{A}}{\partial t}(\mathbf{x}, t) \, \mathrm{d}\Omega, \quad k = 1, 2, 3, \tag{4}$$

with R_k denoting the DC resistance of the stator stranded conductors. For a complete formulation we include the homogeneous Dirichlet boundary condition and an initial condition (IC)

$$\mathbf{n} \times \mathbf{A}(\mathbf{x}, t) = \mathbf{0}, \quad (\mathbf{x}, t) \in \Gamma \times [0, T_\mathrm{end}], \tag{5}$$

$$\mathbf{A}(\mathbf{x}, 0) = \mathbf{A}_0, \quad \mathbf{x} \in \Omega, \tag{6}$$

where $\mathbf{n}$ denotes the outward normal vector to the boundary $\Gamma = \partial\Omega$. Combining the equations (2) and (4), one obtains a coupled field-circuit system, which we will discretize in the following Sect. 2.2.

Remark 1 For the simulation of electric motors one often assumes that they are invariant under translation in the axial x_3-direction. In this case, 2D models in the x_1x_2-plane as the one from Fig. 2 are considered. This leads to the setting

$$\mathbf{J}_\mathrm{s}(\mathbf{x}, t) = \left[0, 0, \mathbf{J}_{\mathrm{s},3}(x_1, x_2, t)\right]^\top, \qquad \mathbf{A}(\mathbf{x}, t) = \left[0, 0, \mathbf{A}_3(x_1, x_2, t)\right]^\top, \tag{7}$$

which transforms the equation (2) for the vector quantity $\mathbf{A}$ into the equation

$$\sigma \frac{\partial \mathbf{A}_3}{\partial t} - \operatorname{div}(\nu \operatorname{grad} \mathbf{A}_3) = \mathbf{J}_{\mathrm{s},3}, \qquad (x_1, x_2, t) \in \Omega_{2\mathrm{D}} \times (0, T_\mathrm{end}], \tag{8}$$

for the scalar quantity $\mathbf{A}_3$, where $\Omega_{2\mathrm{D}} \subset \mathbb{R}^2$ is an open, bounded, simply connected domain with Lipschitz boundary, [9].

2.2 Numerical solution

A standard approach to solve a space-time dependent system is the *method of lines*, where one first discretizes the problem in space using, e.g., the finite element method (FEM), and then integrates the resulting time-dependent system using a numerical time integrator such as, e.g., the implicit Euler method.

Discretization of the coupled system (2)–(4) in space using FEM with d degrees of freedom leads to a time-dependent system on $(0, T_{\text{end}}]$

$$\mathbf{M}_\sigma \frac{\mathrm{d}\mathbf{a}}{\mathrm{d}t}(t) + \mathbf{K}_\nu\big(\mathbf{a}(t)\big)\mathbf{a}(t) - \mathbf{X}\mathbf{i}(t) = \mathbf{0}, \tag{9a}$$

$$\mathbf{X}^\top \frac{\mathrm{d}\mathbf{a}}{\mathrm{d}t}(t) + \mathbf{R}\mathbf{i}(t) = \mathbf{v}(t), \tag{9b}$$

with respect to $\mathbf{a}(t) \in \mathbb{R}^d$ and $\mathbf{i}(t) = [i_1(t), i_2(t), i_3(t)]^\top \in \mathbb{R}^3$. Here $\mathbf{M}_\sigma$ and $\mathbf{K}_\nu(\cdot)$ are the $(d \times d)$-dimensional mass and curl-curl matrices, respectively. Matrix $\mathbf{X} \in \mathbb{R}^{d\times 3}$ is given by

$$\mathbf{X}_{jk} = \int_\Omega \boldsymbol{\chi}_k(\mathbf{x}) \cdot \mathbf{w}_j(\mathbf{x})\,\mathrm{d}\Omega, \quad j = 1, \ldots, d, k = 1, 2, 3,$$

with basis functions $\mathbf{w}_j$ from the Hilbert space $\mathbf{H}(\text{curl}; \Omega)$, see [16]. The matrix $\mathbf{R} \in \mathbb{R}^{3\times 3}$ is a diagonal matrix of resistances R_k and $\mathbf{v}(t) = [v_1(t), v_2(t), v_3(t)]^\top \in \mathbb{R}^3$ is the three-phase input voltage given by (1). Additionally, we have the prescribed ICs

$$\mathbf{a}(0) = \mathbf{a}_0, \qquad \mathbf{i}(0) = \mathbf{i}_0, \tag{10}$$

which together with the system (9a)–(9b) form an initial-value problem (IVP). The equation (9a) is in general a system of index-1 differential-algebraic equations (DAEs), since the matrix $\mathbf{M}_\sigma$ is singular when the considered domain Ω includes non-conducting regions, i.e., regions where $\sigma = 0$. Therefore, implicit methods have to be applied for the time integration [17]. The implicit Euler discretization of the space-discrete system (9a)–(9b) starting from the given values $\mathbf{a}_0$ and $\mathbf{i}_0$ with the step size $\delta T = T_{\text{end}}/N_t$ reads

$$\frac{1}{\delta T}\mathbf{M}_\sigma\,[\mathbf{a}_j - \mathbf{a}_{j-1}] + \mathbf{K}_\nu(\mathbf{a}_j)\mathbf{a}_j - \mathbf{X}\mathbf{i}_j = \mathbf{0}, \tag{11a}$$

$$\frac{1}{\delta T}\mathbf{X}^\top[\mathbf{a}_j - \mathbf{a}_{j-1}] + \mathbf{R}\mathbf{i}_j = \mathbf{v}_j, \tag{11b}$$

where $\mathbf{a}_j$ and $\mathbf{i}_j$ denote approximate solutions at time step $t_j = j\delta T$, with $j = 1, \ldots, N_t$. Since the equation (11a) is nonlinear, a linearization approach, e.g., the Newton method [18], has to be applied at each time step. In [3] it was shown that the implicit Euler method has an implicit projection property that the consistency of initial values known from DAE theory is not an issue.

2.3 Quantities of interest

The electromagnetic torque induced in the air gap and exerted on the rotor can be calculated using the formula [19], [20, Sect. 1.5]

$$T_{\text{EM}} = \int_S \mathbf{r} \times \boldsymbol{\sigma} \cdot \mathrm{d}\mathbf{S} = \int_S \mathbf{r} \times (\boldsymbol{\sigma} \cdot \mathbf{n})\,\mathrm{d}S, \tag{12}$$

where S is the surface enclosing the rotor, $\mathbf{r}$ is the position vector connecting the rotor origin to S, $\mathbf{n}$ is the unit normal vector to S, and $\boldsymbol{\sigma}$ is the *Maxwell stress tensor* [21, Sect. 6.3] given by

$$\boldsymbol{\sigma}_{ij} = \nu_0\big(\mathbf{B}_i\mathbf{B}_j - 0.5|\mathbf{B}|^2\delta_{ij}\big), \quad i,j = 1,2,3, \tag{13}$$

with the *reluctivity in vacuum* ν_0, the *magnetic flux density* $\mathbf{B}$ given by $\mathbf{B} = \operatorname{curl}\mathbf{A}$, and the Kronecker delta δ_{ij}. The product of the produced torque and the rotational speed defines the *mechanical power* P_{mech}, i.e.,

$$P_{\text{mech}} = T_{\text{EM}}\omega_{\text{mech}}. \tag{14}$$

Since a part of the input power is lost as heat, it is important to calculate also the Joule losses. For the 2D-setting from Remark 1 and length ℓ_3 of the motor in the axial x_3-direction, these losses are given by

$$P_{\text{loss}} = \int_{\Omega_{2\text{D}}} \sigma\left(\frac{\partial \mathbf{A}_3}{\partial t}\right)^2 \ell_3 \,\mathrm{d}\Omega_{2\text{D}}, \tag{15}$$

where $\mathbf{A}_3$ denotes the x_3-component of the MVP $\mathbf{A}$, see (7). The considered quantities of interest can be calculated in a post-processing step of the simulation and will be used in an optimization procedure described in Sect. 3.

3 Optimization

In the optimization we use the height h and width w of the rotor bars as optimization variables and parametrize the domain $\Omega = \Omega(p)$ with these two parameters $p = (h, w)$. Our goal is to find the optimal width and height of the rotor bars, such that our objective function J is minimal under the constraint that the design variables lie in a set of admissible designs D_{ad}. Additionally, we require that the state equations (2) and (4) together with the boundary and initial conditions (5)–(6) are fulfilled. As an objective function we consider:

$$\min_{\mathbf{A}_3,p} J(\mathbf{A}_3,p) := -\frac{P_{\text{out}}(\mathbf{A}_3,p)}{P_{\text{in}}(\mathbf{A}_3,p)}, \tag{16}$$

with

$$P_{\text{out}}(\mathbf{A}_3,p) = \int_{(q-1)T}^{qT} P_{\text{mech}}(\mathbf{A}_3,p)\,\mathrm{d}t, \tag{17a}$$

$$P_{\text{in}}(\mathbf{A}_3,p) = \int_{(q-1)T}^{qT} \big[P_{\text{mech}}(\mathbf{A}_3,p) + P_{\text{loss}}(\mathbf{A}_3,p)\big]\,\mathrm{d}t, \tag{17b}$$

where $q \in \mathbb{N}$ represents the period at which the steady state is reached and $T > 0$ is the length of the period (e.g., $q = 10$ and $T = 0.02$ s in Fig. 3). The objective J can be seen as a negative measure of efficiency, as it is given by the quotient of the output and the input power on the right-hand side in (16). Since P_{mech} and P_{loss} involve integrals over the parametrized domain (see (14), (12), and (15)), they depend on the design p. They both depend on the solution $\mathbf{A}_3$ of the state equation, which depends on the design p itself.

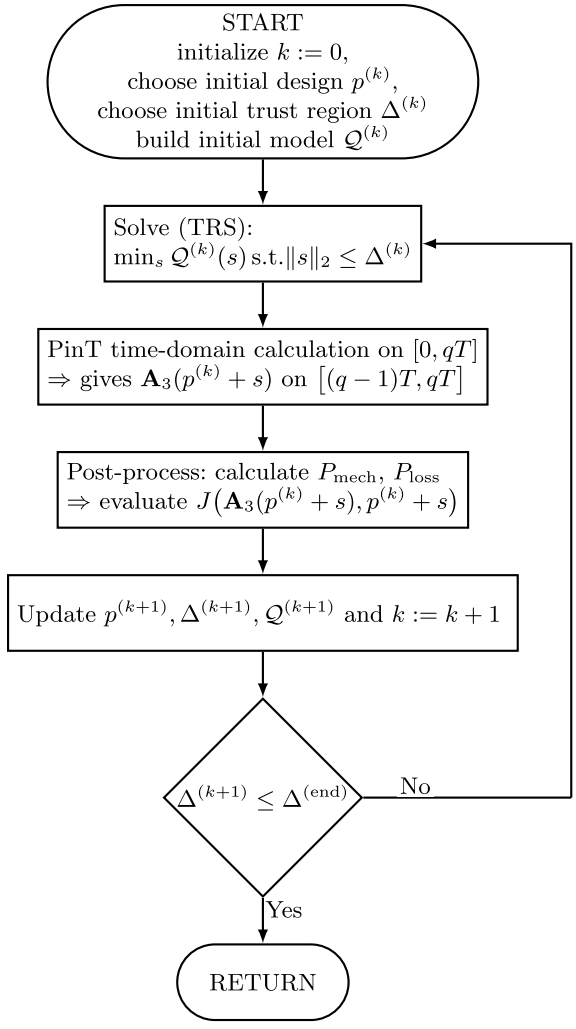

Figure 4 A flowchart of the steady-state optimization procedure with respect to the rotor bar parameter p. It includes a (PinT) time-domain calculation over the interval $[0, qT]$, where the steady state is reached at the qth period of length $T > 0$

In the optimization, we know [22] that for every admissible design p, there is a unique solution to our state equation, which we call $\mathbf{A}_3(p)$ and consider the reduced problem

$$\min_p \hat{J}(p) := J\big(\mathbf{A}_3(p), p\big) \quad \text{subject to } p \in D_{\text{ad}}, \tag{18}$$

which does not involve the state equation as a constraint anymore and where

$$D_{\text{ad}} := \big\{p = (h, w) \in \mathbb{R}^2 : h_{\text{l}} \leq h \leq h_{\text{u}}, w_{\text{l}} \leq w \leq w_{\text{u}}\big\} \tag{19}$$

with $h_{\text{l}}, h_{\text{u}}, w_{\text{l}}, w_{\text{u}} \in \mathbb{R}$.

To solve the optimization problem, we use the derivative-free optimization algorithm Py-BOBYQA [8], which is a Python implementation of BOBYQA [7]. The idea of this algorithm is to use a model for the objective function and improve the model in every iteration to make it approximate the minimum of the objective function sufficiently. Specif-

ically, a quadratic interpolation polynomial $\mathcal{Q}^{(k)}(s) \approx \hat{J}(p^{(k)} + s)$ around the current iterate $p^{(k)}$ is used, which coincides with the true objective function on an interpolation set $Y = \{y_0 = p^{(k)}, \ldots, y_{m-1}\}$:

$$\mathcal{Q}^{(k)}(y_j - y_0) = c + g^T(y_j - y_0) + \frac{1}{2}(y_j - y_0)^T H(y_j - y_0) = \hat{J}(y_j), \quad j = 0, \ldots, m-1 \tag{20}$$

with $c \in \mathbb{R}, g \in \mathbb{R}^n$ and $H \in \mathbb{R}^{n \times n}$.

To improve the model, the current model $\mathcal{Q}^{(k)}$ is minimized inside a trust-region $\{s \in \mathbb{R}^n \colon \|s\|_2 \leq \Delta^{(k)}\}$, where $\Delta^{(k)} > 0$ is the trust-region radius:

$$\min_s \mathcal{Q}^{(k)}(s) \quad \text{s.t. } \|s\|_2 \leq \Delta^{(k)}. \tag{TRS}$$

The initial trust-region radius $\Delta^{(0)}$ is user supplied and has to fulfill $p^{(0)} + \Delta^{(0)} v \in D_{\text{ad}}$ for all $v \in \mathbb{R}^2$ with $\|v\|_2 = 1$. It is called trust-region, because we hope, that when the radius $\Delta^{(k)}$ is small enough, we can trust the model $\mathcal{Q}^{(k)}$ in a neighborhood of our current point $p^{(k)}$.

The quality of the step s computed by solving (TRS) is assessed by calculating the ratio

$$R^{(k)} = \frac{\text{actual reduction}}{\text{predicted reduction}} = \frac{\hat{J}(p^{(k)}) - \hat{J}(p^{(k)} + s)}{\mathcal{Q}^{(k)}(0) - \mathcal{Q}^{(k)}(s)}. \tag{21}$$

If $R^{(k)}$ exceeds a predefined threshold the step is accepted, one of the interpolation points $y_i \in Y$ is replaced by $p^{(k+1)} = p^{(k)} + s$ (the elements of Y then get reordered, such that $y_0 = p^{(k+1)}$), the model $\mathcal{Q}^{(k)}$ is updated and the trust-region radius is enlarged. If $R^{(k)}$ is small or negative, which is the case when the true objective does not decline, or the model is a bad predictor, the step is rejected ($p^{(k+1)} = p^{(k)}$) and the trust-region radius $\Delta^{(k)}$ is reduced, since the model was inaccurate on the former trust-region. The algorithm terminates when the trust-region radius becomes smaller than a predefined tolerance $\Delta^{(\text{end})}$.

A flowchart of the optimization procedure, which iteratively updates the parameter p based on the time-domain calculation until the steady state, is illustrated in Fig. 4.

For a symmetric matrix H, the model $\mathcal{Q}^{(k)}$ has $m = \frac{1}{2}(n+1)(n+2)$ degrees of freedom and, thus, m interpolation points and objective evaluations (right hand side (20)) are needed to compute the model. For every evaluation of the objective function $\hat{J}(p) = J(\mathbf{A}_3(p), p)$ we have to solve our state equation, which is expensive. We therefore want to use as few interpolation points as possible and the idea of BOBYQA is to use only $m = 2n + 1$ and eliminate the rest of the degrees of freedom by requiring that the Hessian of the interpolation polynomial has minimal curvature. To compute such a minimal curvature model, the following optimization problem has to be solved:

$$\begin{aligned} &\min_{c,g,H} \frac{1}{4}\|H - H_{\text{prev}}\|_F^2 \\ &\quad \text{s.t. } \mathcal{Q}^{(k)}(y_j - y_0) = \hat{J}(y_j),\ j = 0, \ldots, m-1 \end{aligned} \tag{22}$$

When computing the initial model, we set $H_{\text{prev}} = 0$. The initial interpolation set is chosen as $Y = \{p^{(0)}, p^{(0)} \pm \Delta^{(0)} e_1, p^{(0)} \pm \Delta^{(0)} e_2\}$, where e_i is a zero vector with a one in the ith entry. When the set Y is poised for interpolation, the solution to (22) is given by the solution of a linear system in dimension $m + n + 1$ (see [23] for details).

What we have omitted in the description of Py-BOBYQA is, that the algorithm has a model improvement phase, in which the geometry of the interpolation set is improved when necessary, see [23].

By describing the geometry of the rotor bars with two design variables and using $m = 2n + 1$ degrees of freedom for the quadratic model, we still have to solve the discrete time-domain problem (11a)–(11b) for five different designs to compute an initial interpolation model and one time in every optimization iteration. As this is the most expensive part in the optimization, we use a PinT algorithm to accelerate the solution of the discrete time-domain problem (11a)–(11b), which we describe in the following Sect. 4.

4 AT-MGRIT

Consider an IVP of the form

$$\mathbf{u}'(t) = \mathbf{f}\big(t, \mathbf{u}(t)\big), \quad t \in (0, T_{\text{end}}], \qquad \mathbf{u}(0) = \mathbf{g}_0, \tag{23}$$

which arises, for example, after the spatial discretization of a space-time PDE. We discretize (23) on a uniform temporal grid $t_j = j\delta T, j = 0, 1, \ldots, N_t$, where N_t is the number of time steps, $\delta T = T_{\text{end}}/N_t$ and $\mathbf{u}_j \approx \mathbf{u}(t_j)$. Let Φ_j be a time integrator which propagates the solution $\mathbf{u}_{j-1}$ from time point t_{j-1} to time point t_j, including all problem-dependent forcing terms. Considering a one-step time integration method such as, e.g., implicit Euler, the time-discrete problem is given by

$$\mathbf{u}_j = \mathbf{\Phi}_j(\mathbf{u}_{j-1}), \quad j = 1, 2, \ldots, N_t,$$

or, considering all time points at once, we obtain the space-time system

$$\mathcal{A}(\mathbf{u}) \equiv \begin{bmatrix} \mathbf{u}_0 \\ \mathbf{u}_1 - \mathbf{\Phi}_1(\mathbf{u}_0) \\ \vdots \\ \mathbf{u}_{N_t} - \mathbf{\Phi}_{N_t}(\mathbf{u}_{N_{t-1}}) \end{bmatrix} = \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix} \equiv \mathbf{g}. \tag{24}$$

We now describe the AT-MGRIT algorithm for the time-parallel solution of the problem (24).

AT-MGRIT is an iterative method for solving IVPs using multigrid reduction techniques [5]. It is therefore based on a hierarchy of temporal grids, restriction and prolongation operators for the transfer between temporal grids, and relaxation schemes. For a given (fine) time grid $\mathcal{T}^{(0)} = \{j\delta T^{(0)} : j = 0, 1, \ldots, N_t^{(0)}\}$, with $N_t^{(0)} = N_t$, and a given coarsening factor $\widetilde{m} > 1$, we define a splitting of all time points into F- and C-points, such that every $\widetilde{m}$th point is a C-point and all others are F-points. The C-points define a global coarse grid $\mathcal{T}^{(1)} = \{j\delta T^{(1)} : j = 0, 1, \ldots, N_t^{(1)}\}$ with time-step size $\delta T^{(1)} = \widetilde{m}\delta T^{(0)}$. For the coarse-grid operator we choose a re-discretization of the problem with step size $\delta T^{(1)}$, although other approaches such as coarsening in space [24–26] can also be used. Applying this strategy recursively for $L > 1$ levels, we obtain a multi-level hierarchy of time grids $\mathcal{T}^{(\ell)}$ with $\ell = 0, 1, \ldots, L - 1$. At the coarsest level, we define $N_t^{(L-1)} + 1$ overlapping local coarse grids based on the global grid $\mathcal{T}^{(L-1)}$. For a given distance $\widetilde{n} \in \mathbb{N}$, the ith local coarse grid

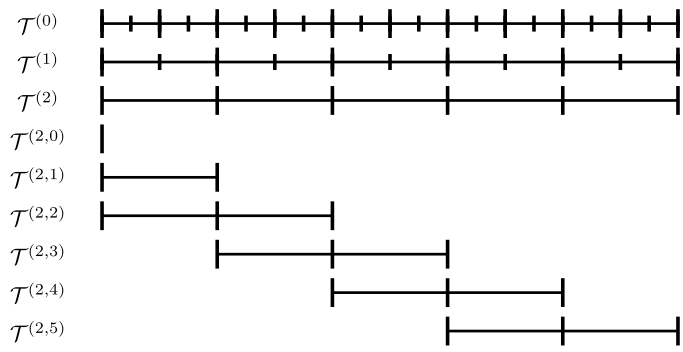

Figure 5 Example of a three-level time grid hierarchy for the AT-MGRIT algorithm for $N_t^{(0)} = 21$, $\widetilde{m} = 2$ and $\widetilde{n} = 3$. C-points are represented by long markers and F-points by short markers

Figure 6 Example of F- and C-relaxation for a temporal grid with 9 time points. Independent blocks of F- and C-points (gray boxes) can be updated simultaneously

is given by

$$\mathcal{T}^{(L-1,i)} = \left\{ j\delta T^{(L-1)} : j \in \left[\max(0, i - \widetilde{n} + 1), i\right] \right\}.$$

An example of a three-level time-grid hierarchy for $N_t^{(0)} = 21$, $\widetilde{m} = 2$ and $\widetilde{n} = 3$ is shown in Fig. 5. Note that all space-time problems associated with the local grids are independent of each other and can be solved simultaneously.

We define two types of relaxation schemes, the so-called F-relaxation and the so-called C-relaxation. The F-relaxation propagates the solution from a C-point to all subsequent F-points up to the next C-point. Similarly, the C-relaxation propagates the solution to a C-point. Figure 6 illustrates the actions of F- and C-relaxation. Both relaxation schemes are highly parallel and can be applied simultaneously to each interval of F- or C-points, respectively. For the transfer between global temporal grids, we define restriction as an injection at C-points and the "ideal" prolongation, corresponding to the transpose of an injection at C-points followed by an F-relaxation. For the transfer from the global temporal grid to the local time grids at the coarsest level, we define both restriction and prolongation as injection.

Using the full approximation storage framework [27], the multilevel AT-MGRIT algorithm can be written as in Algorithm 1. There, $\mathcal{A}^{(\ell)}\mathbf{u}^{(\ell)} = \mathbf{g}^{(\ell)}$ and $\mathcal{A}^{(\ell,i)}\mathbf{u}^{(\ell,i)} = \mathbf{g}^{(\ell,i)}$ specifies the space-time system of equations on levels $\ell = 0, 1, \ldots, L-1$ and on the local coarse grids $i = 0, 1, \ldots, N_t^{(\ell)}$, respectively. The transfer between the global temporal grids is given by the operators $R_I^{(\ell)}$ and $P^{(\ell)}$ and the transfer from the global temporal grid to the local grids at the coarsest level by the operators $R_I^{(\ell,i)}$ and $P_I^{(\ell,i)}$. The relaxation scheme of the method can be controlled by the parameter ν, where $\nu = 1$, i.e., an F-relaxation followed by a C-relaxation and another F-relaxation is a typical choice for the AT-MGRIT algorithm.

Algorithm 1 AT-MGRIT(ℓ)

Repeat
  If ℓ is the coarsest level:
    For $i = 0$ to $N_t^{(\ell)}$:
      Restrict to local grids
        $\mathbf{v}^{(\ell,i)} = R_I^{(\ell,i)}(\mathbf{v}^{(\ell)})$
        $\mathbf{g}^{(\ell,i)} = R_I^{(\ell,i)}(\mathbf{g}^{(\ell)})$
      Solve local problem $\mathcal{A}^{(\ell,i)}(\mathbf{u}^{(\ell,i)}) = \mathbf{g}^{(\ell,i)}$
      Update $\mathbf{u}^{(\ell)} = P_I^{(\ell,i)}\mathbf{u}^{(\ell,i)}$
  Else
    Apply F-relaxation to $\mathcal{A}^{(\ell)}(\mathbf{u}^{(\ell)}) = \mathbf{g}^{(\ell)}$
    Apply ν times CF-relaxation to $\mathcal{A}^{(\ell)}(\mathbf{u}^{(\ell)}) = \mathbf{g}^{(\ell)}$
    Inject the approximation and its residual to the coarse grid
      $\mathbf{u}^{(\ell+1)} = R_I^{(\ell)}(\mathbf{u}^{(\ell)})$
      $\mathbf{v}^{(\ell+1)} = R_I^{(\ell)}(\mathbf{u}^{(\ell)})$
      $\mathbf{g}^{(\ell+1)} = R_I^{(\ell)}(\mathbf{g}^{(\ell)} - \mathcal{A}^{(\ell)}\mathbf{u}^{(\ell)})$
    Compute right-hand side $\mathbf{g}^{(\ell+1)} = \mathcal{A}^{(\ell+1)}(\mathbf{v}^{(\ell+1)}) + \mathbf{g}^{(\ell+1)}$
    Solve on next level: AT-MGRIT($\ell + 1$)
    Correct using ideal interpolation: $\mathbf{u}^{(\ell)} = \mathbf{u}^{(\ell)} + P^{(\ell)}(\mathbf{u}^{(\ell+1)} - \mathbf{v}^{(\ell+1)})$
Until stopping criterion is reached

Note that all components of the algorithm are highly parallel and can be executed simultaneously. The algorithm solves for the exact discrete solution of the fine temporal grid after $N_t^{(0)}/(2\widetilde{m})$ iterations for FCF-relaxation. Furthermore, the algorithm is equivalent to the MGRIT method [2] if $\widetilde{n} = N_t^{(L-1)} + 1$ and equivalent to Parareal [1] for $L = 2$, $\nu = 0$, and $\widetilde{n} = N_t^{(1)} + 1$, [6].

5 Numerical experiments

In this section we apply the optimization procedure described in Sect. 3 combined with the AT-MGRIT algorithm from Sect. 4 to an induction machine. For this, we consider one pole of the 2D four-pole squirrel-cage motor model depicted in Fig. 2, imposing periodic boundary conditions due to the symmetry [10]. A constant rotational speed $\omega_{\text{mech}} = 1420$ rpm and a nonlinear material behavior are chosen for the problem setting. PinT simulation of this machine model was already performed, e.g., in [3, 4], which we now incorporate into a geometry optimization framework.

The motor is excited with a three-phase sinusoidal voltage supply of frequency $f = 50$ Hz and amplitude $\hat{U} = 311.1$ V given by

$$v_k^{\text{rel}}(t) = r(t)v_k(t), \quad k = 1, 2, 3, \tag{25}$$

where v_k is defined in (1) and

$$r(t) = \begin{cases} 0.5(1 - \cos(0.5\pi f t)), & t \in [0, 2T), \\ 1, & t \in [2T, T_{\text{end}}] \end{cases} \tag{26}$$

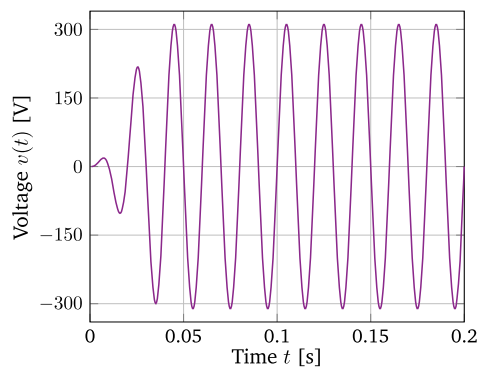

Figure 7 One phase $v_1^{\mathrm{rel}}(t)$ of the voltage (25) on interval $[0, 0.2]$ s with frequency $f = 50$ Hz and amplitude $\hat{U} = 311.1$ V

is an initial ramp-up used on the first two periods of length $T = 1/f = 0.02$ s to reduce the transient response [10]. One phase of the applied signal (25) is shown in Fig. 7.

The time-domain simulation is performed by solving (11a)–(11b) on a (fine) temporal grid with $N_t = 16{,}384$ points and end point $T_{\mathrm{end}} = 0.2$ s. This corresponds to time stepping with a step size of $\delta T \approx 1.2 \cdot 10^{-5}$ s, starting from a homogeneous IC for the MVP, i.e., $\mathbf{a}_0 = \mathbf{0}$ in (10). Within this setting, the steady state is reached at the period $q = 10$ up to the tolerance of $< 5 \cdot 10^{-4}$ in terms of the relative error

$$\epsilon(q-1) = \frac{|T_{\mathrm{EM,avg}}^{(q)} - T_{\mathrm{EM,avg}}^{(q-1)}|}{|T_{\mathrm{EM,avg}}^{(q)}|}, \quad \text{with } T_{\mathrm{EM,avg}}^{(q)} = \frac{1}{N_{\mathrm{p}}} \sum_{i=1+(q-1)N_{\mathrm{p}}}^{qN_{\mathrm{p}}} T_{\mathrm{EM}}(t_i)$$

denoting the average torque at the period q and $N_{\mathrm{p}} = \lfloor N_t/q \rfloor$ being the number of time steps per period.

Based on the steady-state behavior of the induction motor, our goal is to optimize the rotor bars size, in particularly, their width and height using the objective function (18). For this, we set

$$h_{\mathrm{l}} = 0.007, \qquad h_{\mathrm{u}} = 0.015, \qquad w_{\mathrm{l}} = 0.0015, \qquad w_{\mathrm{u}} = 0.0035$$

as the admissible design bounds in (19), choose $h^{(0)} = 0.01425$ and $w^{(0)} = 0.002$ as an initial design depicted in Fig. 9 (left) and set the initial trust-region radius to $\Delta^{(0)} = 10^{-4}$. For each objective function evaluation, we generate a mesh representation of the current geometry using `Gmsh` [28, 29], where each mesh, depending on the geometry, consists of approximately 4,500 degrees of freedom. Afterwards the AT-MGRIT algorithm of the Python package `PyMGRIT` [4, 30] is called based on a two-level strategy with $\widetilde{m} = 64$, $\widetilde{n} = 100$, and F-relaxation. The time integration on the temporal grids of AT-MGRIT is done by means of the external `GetDP` library [11], which implements the implicit Euler method for the time stepping and the Newton method with damping as a nonlinear solver. Furthermore, we choose an initial guess for the AT-MGRIT algorithm based on a global coarse grid solve. This choice was shown in [6] to be a promising setting for the simulation of this model, since too large time steps on the coarse grid can otherwise lead to divergence of at least one nonlinear solve in `GetDP`. The AT-MGRIT algorithm terminates when the

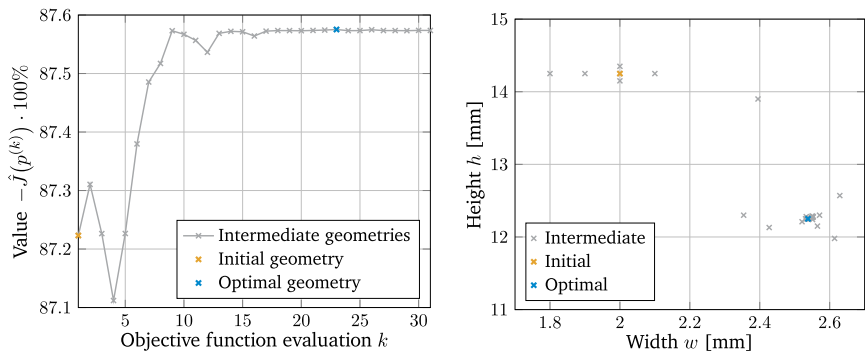

Figure 8 Optimization process over iterations: negative objective function values (left) and geometries (right)

maximums norm of the relative difference of the Joule losses of two successive iterations is less than 1%, see [4] for details.

The calculations were performed on an Intel Xeon Phi cluster consisting of four 1.4 GHz Intel Xeon Phi processors. The code can be found in the `PyMGRIT` repository [30]. The implementation uses a master/worker strategy for the optimization and the simulations, using one process for the optimization and 256 processes for each simulation, where all resources are used for temporal parallelization.

5.1 Optimization results

Figure 8 shows the negative values of the objective function evaluated during the optimization procedure on the left and an overview of the geometries considered on the right. In the 26 optimization iterations a total of 31 function evaluations were required: one evaluation in every iteration and an additional five in the first iteration for building the initial quadratic interpolation model. The optimal geometry $\bar{p}$ with $\bar{w} = 0.00254$ and $\bar{h} = 0.01226$ was found in the 18th iteration. In the iterations after the minimum was found and added to the interpolation set, the solution of (TRS) was always a point worse than $\bar{p}$, resulting in a negative ratio (21), a rejected step and a reduced trust-region radius. The algorithm then terminated when the trust-region radius $\Delta^{(k)}$ became smaller than our chosen tolerance $\Delta^{(\text{end})} = 10^{-6}$. Figure 9 shows the two geometries for the initial design (left) and the optimal design (right). Different structures of the geometry of the rotor bars (orange) can be seen: the bars in the optimal design are less high and significantly wider than those in the initial design. Overall, the optimal design increases the negative of the objective function and thus the efficiency of the electrical machine from 87.22% to 87.57%. Figure 10 gives a detailed comparison of the torque (left) and the Joule losses (right) between the initial and the optimal design in the steady state. Compared to the original design, both values were increased, with an average increase of 16.25% in torque and 20.03% in Joule losses for the optimal design.

In each step of the optimization algorithm we use the AT-MGRIT algorithm to simulate the corresponding geometry. To determine the effect of the time-parallel method compared to sequential time-stepping for solving the problem, we count the calls of the time integrator for both methods. In each iteration of the optimization algorithm, we achieve a theoretical speedup of up to 11.67 by using AT-MGRIT compared to the standard sequential time-stepping. Note that for this application, counting the serial solves of the AT-MGRIT algorithm is a reasonable measure to determine the theoretical speedup, since the

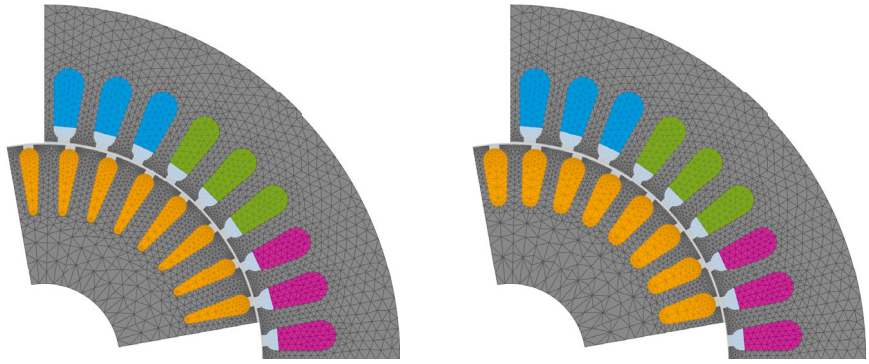

Figure 9 Grid view of the initial design (left) and the optimal design (right)

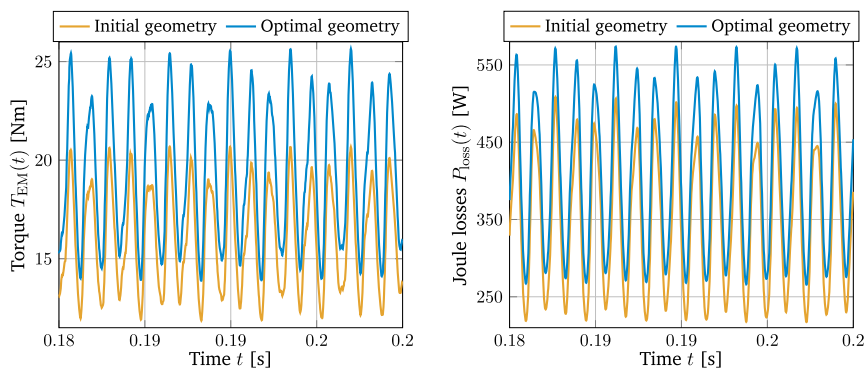

Figure 10 Torque (left) and Joule losses (right) for the initial and the optimal designs of a nonlinear "im_3kw" induction machine model

computational cost dominates the runtime cost of the algorithm and the communication cost is negligible compared to the computational cost.

6 Conclusion

In this paper we incorporated a parallel-in-time solution approach into a box constrained shape optimization of an induction motor. Each iteration of the optimization algorithm includes a time-domain solution until the steady state, which we parallelize using the asynchronous truncated multigrid-reduction-in-time method. The objective function is a measure of the motor's efficiency and is constructed from the calculated steady-state characteristics such as the produced torque and the Joule losses. Thanks to the time parallelization an optimal geometry is obtained about 11 times faster compared to the standard sequential time stepping.

Acknowledgements
The authors thank Dr. Oliver Rain from Robert Bosch GmbH for sharing his expertise in electric motors and corporate engineering.

Funding
This work is supported by the European High-Performance Computing Joint Undertaking (JU) under the grant agreement No 955701. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Belgium, France, Germany, Switzerland. Open Access funding enabled and organized by Projekt DEAL.

Abbreviations
PinT, Parallel-in-time; MGRIT, multigrid-reduction-in-time; AT-MGRIT, asynchronous truncated multigrid-reduction-in-time; PDE, partial differential equation; 2D, two-dimensional; MQS, magnetoquasistatic; MVP, magnetic vector potential; IC, initial condition; FEM, finite element method; IVP, initial-value problem; DAEs, differential-algebraic equations.

Availability of data and materials
The optimization workflow and the parallel implementation of the AT-MGRIT algorithm are available in the `PyMGRIT` package at https://github.com/pymgrit/pymgrit.

Declarations

Competing interests
The authors declare that they have no competing interests.

Author contributions
The electric machine model was adapted by BP for geometry optimization. The optimization procedure was designed by IKR, BP, SF, SSc and JH, and implemented by JH. The manuscript was written by IKR, BP and JH, and edited by SF, SU, and SSc. All authors read and approved the final manuscript.

Author details
[1]Department of Mathematics, Bergische Universität Wuppertal, Gaußstr. 20, 42119, Wuppertal, Germany. [2]Department of Mathematics, Technische Universität Darmstadt, Dolivostr. 15, 64293, Darmstadt, Germany. [3]Computational Electromagnetics, Technische Universität Darmstadt, Schloßgartenstr. 8, 64289, Darmstadt, Germany.

Received: 21 March 2022 Accepted: 29 May 2023 Published online: 22 June 2023

References
1. Lions J-L, Maday Y, Turinici G. Résolution d'EDP par un schéma en temps "pararéel". C R Acad Sci, Ser 1 Math. 2001;332(7):661–8.
2. Falgout RD, Friedhoff S, Kolev TV, MacLachlan SP, Schroder JB. Parallel time integration with multigrid. SIAM J Sci Comput. 2014;36(6):635–61.
3. Schöps S, Niyonzima I, Clemens M. Parallel-in-time simulation of eddy current problems using parareal. IEEE Trans Magn. 2018;54(3):1–4.
4. Bolten M, Friedhoff S, Hahne J, Schöps S. Parallel-in-time simulation of an electrical machine using MGRIT. Comput Vis Sci. 2020;23(1–4):14–14.
5. Ries M, Trottenberg U, Winter G. A note on MGR methods. Linear Algebra Appl. 1983;49:1–26.
6. Hahne J, Southworth B, Friedhoff S. Asynchronous truncated multigrid-reduction-in-time (AT-MGRIT). 2021. arXiv:2107.09596.
7. Powell MJD. The BOBYQA algorithm for bound constrained optimization without derivatives. Cambridge NA Report NA2009/06. Cambridge: University of Cambridge; 2009.
8. Cartis C, Fiala J, Marteau B, Roberts L. Improving the flexibility and robustness of model-based derivative-free optimization solvers. ACM Trans Math Softw. 2019;45(3).
9. Kulchytska-Ruchka I. Parallel-in-time simulation of electromagnetic energy converters. Dissertation. 2021. https://doi.org/10.26083/tuprints-00019280.
10. Gyselinck J, Vandevelde L, Melkebeek JAA. Multi-slice FE modeling of electrical machines with skewed slots-the skew discretization error. IEEE Trans Magn. 2001;37:3233–7.
11. Geuzaine C. Getdp: a general finite-element solver for the de Rham complex. PAMM. 2007;7(1):1010603–4. https://onlinelibrary.wiley.com/doi/pdf/10.1002/pamm.200700750.
12. Jackson JD. Classical electrodynamics. 3rd ed. New York: Wiley; 1999.
13. Ida N, Bastos JPA. Electromagnetics and calculation of fields. 2nd ed. Berlin: Springer; 1992.
14. Schöps S, De Gersem H, Weiland T. Winding functions in transient magnetoquasistatic field-circuit coupled simulations. Compel. 2013;32:2063–83. https://doi.org/10.1108/COMPEL-01-2013-0004.
15. Gander MJ, Kulchytska-Ruchka I, Niyonzima I, Schöps S. A new parareal algorithm for problems with discontinuous sources. SIAM J Sci Comput. 2019;41:375–95. arXiv:1803.05503. https://doi.org/10.1137/18M1175653.
16. Bossavit A. Computational electromagnetism: variational formulations, complementarity, edge elements. San Diego: Academic Press; 1998. https://doi.org/10.1016/B978-0-12-118710-1.X5000-4.
17. Hairer E, Nørsett SP, Wanner G. Solving ordinary differential equations II: stiff and differential-algebraic problems. 2nd ed. Springer series in computational mathematics. Berlin: Springer; 2002.
18. Deuflhard P. Newton methods for nonlinear problems: affine invariance and adaptive algorithms. Berlin: Springer; 2011.
19. Arkkio A. Analysis of induction motors based on the numerical solution of the magnetic field and circuit equations. Phd thesis. urn:nbn:fi:tkk-001267.
20. Pyrhönen J, Jokinen T, Valéria H. Design of rotating electrical machines. New York: Wiley; 2008.
21. Salon SJ. Finite element analysis of electrical machines. Norwell: Kluwer Academic; 1995
22. Alonso Rodríguez A, Valli A. Eddy current approximation of Maxwell equations. Modeling, simulation and applications, vol. 4. Berlin: Springer; 2010. https://doi.org/10.1007/978-88-470-1506-7.
23. Roberts L. Derivative-free algorithms for nonlinear optimisation problems. PhD thesis. University of Oxford; 2019.
24. Ruprecht D. Convergence of parareal with spatial coarsening. PAMM. 2014;14(1):1031–4.
25. Lunet T, Bodart J, Gratton S, Vasseur X. Time-parallel simulation of the decay of homogeneous turbulence using parareal with spatial coarsening. Comput Vis Sci. 2018;19(1):31–44.

26. Howse A, Sterck H, Falgout R, MacLachlan S, Schroder J. Parallel-in-time multigrid with adaptive spatial coarsening for the linear advection and inviscid Burgers equations. SIAM J Sci Comput. 2019;41(1):538–65.
27. Brandt A. Multi-level adaptive solutions to boundary-value problems. Math Comput. 1977;31(138):333–90.
28. Geuzaine C, Remacle J-F. Gmsh: a 3-d finite element mesh generator with built-in pre- and post-processing facilities. Int J Numer Methods Eng. 2009;79(11):1309–31. https://onlinelibrary.wiley.com/doi/pdf/10.1002/nme.2579.
29. Gmsh: A three-dimensional finite element mesh generator with built-in pre- and post-processing facilities. http://www.gmsh.info
30. Hahne J, Friedhoff S. Github repository for `PyMGRIT`. https://github.com/pymgrit/pymgrit, Online; accessed June 21, 2021 (2020)

Publisher's Note