Can time parallelization speed up magnetic transients with just a few cores?

Leonardo Mutti, Pauline Ferrouillat and Frederic Vi
Altair Engineering, Meylan, France

Email: leonardo.mutti01@universitadipavia.it; pferrouillat@altair.com

***Abstract*—We explore ways to accelerate the simulation of magnetic transient phenomena and focus on time parallelization. To beat a classical implicit Euler time-stepping we combine the finite element method in space with a discontinuous Galerkin formulation in time and a ParaReal solver, and obtain a speedup already with two cores. We also make a novel performance comparison of discontinuous and continuous formulations in time. The test case is a 2D linear eddy current problem with solid and coil conductors and magnetic vector potentials.**

I. Introduction

Time is sequential and as such, parallelizing transient simulations is not trivial. Common discretization schemes involve the finite element method (FEM) in space coupled with a time-stepping algorithm. To parallelize the temporal dimension one can alternatively use FEM also in time: two promising approaches for this, namely continuous and discontinuous Galerkin methods, lack a systematic performance comparison [1], and parallel-in-time methods have mostly been tested on massive architectures and complex test cases [2], [3]. We thus provide the following contributions:

- the performances of low order, time-continuous (cG) and discontinuous (dG) Galerkin methods are numerically compared,
- and by coupling dG with the ParaReal (PR) algorithm we obtain a speedup over the implicit Euler (IE) algorithm with few cores and on a small experiment.

Our runs are performed on a 2D eddy current problem with linear magnetic materials, solid and coil conductors, and utilizing a magnetic vector potential.

II. Eddy current problems

We work in 2D, and denote magnetic induction and electric field respectively by b and e. Let us define the magnetic vector potential $a^* = (0, 0, u)$ by $\nabla \times a^* = b$ and $-\partial_t a^* = e$. Also consider σ and ν, conductivity and magnetic reluctivity, as well as $J = (0, 0, j)$, the current density from the coil conductors. By employing Maxwell's equations as in [4], we find

$$-\nu \Delta u + \sigma \partial_t u = j, \tag{1}$$

which is the 2D transient problem of interest, to be solved on some domain Ω (see fig. 1), in the time interval $[0, T]$ and with suitable initial and boundary conditions.

III. Discretization

To simulate (1), three possibilities are now reviewed. They all make use of the finite element method (FEM) in space, but differ in the time discretization. To explain them, we first introduce a uniform discretization $0 = t^0 < t^1 < ... < t^K = T$ of $[0, T]$, where every subinterval $[t^k, t^{k+1}]$ has width δt.

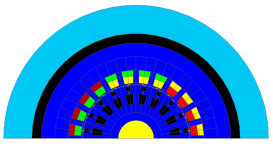

Figure 1. Half-section of electric motor with coil and solid conductors

Having done this, let us now describe the three discretizations of interest.

A. Implicit Euler - FEM

Applying FEM in space and the implicit Euler method in time as done in [5] we find the sequence of linear systems

$$\nu A + \delta t^{-1} \sigma M (\mathbf{u}^{k+1} - \mathbf{u}^k) = \mathbf{f}^{k+1}, \quad k = 0, ..., K-1, \tag{2}$$

where $\mathbf{u}^k$ is the vector of nodal values of the approximate solutions at time t^k, and $\mathbf{u}^0$ is a known initial condition. Here, if $\{\varphi_i\}_i$ is the usual nodal FEM basis in space, we have $\mathbf{f}_i^k := \int_\Omega j(t^k)\varphi_i$, together with $A_{ij} := \int_\Omega \nabla\varphi_i \cdot \nabla\varphi_j$ and $M_{ij} := \int_\Omega \varphi_i \varphi_j$. The scheme in (2), being sequential, is a potentially slow algorithm that does not make full use of the available computational resources. To solve for (2) we adopt the MUMPS solver [6] on 1 core. This yield better performance than with more cores, given the small test case.

B. Space-time methods

If instead of performing a solution at every t^k we solve many time steps at once in "chunks", we can introduce time parallelization. For notational simplicity, we assume in what follows that there is only one chunk on the entire $[0, T]$.

1) Continuous Galerkin - FEM: for this approach we look at a space-time weak formulation of (1), namely

$$\nu \int_\Omega \int_0^T \nabla u \cdot \nabla v + \sigma \int_\Omega \int_0^T \partial_t u \; v = \int_\Omega \int_0^T j v. \tag{3}$$

We now use tensor product finite element functions, that is, $u(x,t) \simeq \sum_i \sum_k \mathbf{u}_i^k \varphi_i(x) \eta^k(t)$, where $\mathbf{u}_i^k$ approximates u at time t^k and node x_i, $\{\eta^k\}_{k=0}^K$ is the FEM linear nodal basis for $[0, T]$, and $v = \varphi_i \eta^k$ for all choices of i and k. We get a block tridiagonal system $X^{\text{cG}} \mathbf{u}^{1:K} = \mathbf{f}^{\text{cG}}$, where $f_{ik}^{\text{cG}} = \int_\Omega j(t^k) v_i \eta_k$ and $X^{\text{cG}} = \sigma \; D \otimes M + \nu \; N \otimes A$, with $D_{kl} = \int_0^T \eta_l \partial_t \eta_k$, $N_{kl} = \int_0^T \eta_k \eta_l$, and A and M are the same matrices defined in the implicit Euler method. The symbol $\otimes$ above denotes the Kronecker product. We use an ILU-preconditioned GMRES method, where the ILU implementation is provided by MUMPS [6].

2) Discontinuous Galerkin - ParaReal - FEM: if instead we choose η^k piecewise continuous, we obtain as in [5] a discontinuous Galerkin formulation. For simplicity, we set the η_k piecewise constant. The dG system $X^{\text{dG}}\mathbf{u}^{1:K} = \mathbf{f}^{\text{dG}}$ is then fully equivalent to a batched version of (2), since

$$X^{\text{dG}} = \begin{pmatrix} A + \frac{M}{\delta t} & & \\ -\frac{M}{\delta t} & A + \frac{M}{\delta t} & \\ & \dots & \dots \end{pmatrix}, \; f^{\text{dG}} = \begin{pmatrix} f^1 + \frac{M}{\delta t} u^0 \\ f^2 \\ \dots \end{pmatrix}.$$

We chose ParaReal [3] as time-parallel solver. It works by splitting the timesteps among available CPUs. Each CPU in parallel solves for its timesteps in a so-called "fine" solve, using a local sequential time-stepping. Solutions from different CPUs are then combined through a "coarse" solve, also using a sequential time-stepping, this time across CPUs. We use implicit Euler in both coarse and fine solves.

IV. Numerical experiments

We now report our numerical findings. We used at most 8 cores running at 2.6 GHz. The number of spatial unknowns is about 40k and we examine different time refinements, that is, we choose $K \in \{48, 96, 960\}$ for the same final time T.

The matrices A and M are constant over $[0, T]$ and could potentially be computed only once, to be then reused for all timesteps. However, in more complicated scenarios, A and M might vary in time. To obtain comparable computational times to these cases, we decided to repeat finite element integration at every timestep, thereby recomputing A and M.

The chunks themselves can contain 24, 48 or 96 timesteps: for every run we report the size that worked best.

All solvers are available in the PETSc [7] library, apart from ParaReal. The finite element integration is provided by a noncommercial version of the Flux software [8].

A. Comparison of space-time methods

Let us start by comparing continuous and discontinuous Galerkin methods in time.

In table I are the cumulative timings for finite element integration and linear system solving for the entire simulation. We only report the case $K = 96$ to illustrate the main conclusions, also found with other K:

- increasing the chunk size accelerates dG-PR method and is detrimental for cG,
- dG-PR is faster than dG.

In table II we find the accuracy in time of the various runs, against a reference IE solution with $K = 960$ timesteps. In particular, we compute the total magnetic flux on the domain and measure the error in the flux in the $L^2(0, T)$ and $L^\infty(0, T)$ norms. As the cG approach has worse stability properties than dG [9], it is not accurate at t^1 and we needed to solve for t^1 with implicit Euler. Moreover, the largest local errors were observed at t^1 for both cG and dG-PR methods, hence the L^∞ global errors are equal. Since cG has a better theoretical order of convergence [9] the L^2 accuracy is better, although negligibly so.

Table I
Strong scaling results for dG-PR and cG methods for $K = 96$

CPUs	2		4		8	
Method	cG	dG-PR	cG	dG-PR	cG	dG-PR
Chunk size	24	96	24	96	24	96
Integration (s)	156.4	11.1	111.0	8.5	75.6	5.7
Solving (s)	223.8	21.2	164.4	12.1	133.5	8.8

Table II
Accuracy comparison of dG-PR and cG methods for $K = 96$

Method	cG	dG-PR
Chunk size	24	96
Flux error in $L^2(0, T)$	0.001704	0.001801
Flux error in $L^\infty(0, T)$	0.000885	0.000885

B. Speedup over Implicit Euler

The dG-PR method is faster than a classical implicit Euler time-stepping already with 2 CPUs, see table III. Note that ParaReal does not provide an optimal parallel scalability.

Table III
Speedup of dG-PR over IE

K	48			96			960		
Chunk size	48	48	48	96	96	96	96	96	96
CPUs	2	4	8	2	4	8	2	4	8
Integration (speedup)	2.3	3.6	4.1	2.1	2.8	4.2	2.6	4.4	5.5
Solving (speedup)	2.1	3.3	3.9	2.1	3.7	5.1	2.1	3.7	5.2

V. Conclusion

We have compared time-continuous and discontinuous Galerkin methods for linear eddy current transients, and shown that the latter are faster than implicit Euler on a few cores, when coupled with a ParaReal solver. The full article will feature more extensive tests and discussion.

Possible next steps include extensions to nonlinear materials and moving meshes, higher order methods in time and achieving better parallel scalability.

References

[1] M. von Danwitz, I. Voulis, N. Hosters, and M. Behr, "Time-continuous and time-discontinuous space-time finite elements for advection-diffusion problems," *International Journal for Numerical Methods in Engineering*, vol. 124, no. 14, pp. 3117–3144, 2023.

[2] M. J. Gander and M. Neumüller, "Analysis of a new space-time parallel multigrid algorithm for parabolic problems," *SIAM Journal on Scientific Computing*, vol. 38, no. 4, pp. A2173–A2208, 2016.

[3] S. Friedhoff, J. Hahne, I. Kulchytska-Ruchka, and S. Schöps, "Exploring parallel-in-time approaches for eddy current problems," in *Progress in Industrial Mathematics at ECMI 2018*. Springer, 2019, p. 373–379.

[4] G. Meunier, *The Finite Element Method for Electromagnetic Modeling*. Wiley-ISTE, 2010.

[5] V. Thomée, *Galerkin Finite Element Methods for Parabolic Problems*, ser. Springer Series in Computational Mathematics. Springer, 2006, vol. 25.

[6] https://mumps-solver.org/.

[7] https://petsc.org/release/.

[8] Altair, https://altair.com/flux/.

[9] M. Schmich, "Adaptive finite element methods for computing nonstationary incompressible flows," 01 2009.