Journal of Computational Physics 279 (2014) 67–79

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

Parareal in time 3D numerical solver for the LWR Benchmark neutron diffusion transient model

Anne-Marie Baudron [b,e], Jean-Jacques Lautard [b,e], Yvon Maday [a,b,c], Mohamed Kamel Riahi [b,f,*], Julien Salomon [d]

[a] *Sorbonne Universités, UPMC Univ Paris 06, UMR 7598, Laboratoire Jacques-Louis Lions and Institut Universitaire de France, F-75005, Paris, France*
[b] *Laboratoire de Recherche Conventionné MANON, CEA/DEN/DANS/DM2S and UPMC-CNRS/LJLL, France*
[c] *Brown Univ, Division of Applied Maths, Providence, RI, USA*
[d] *CEREMADE, Univ Paris-Dauphine, Pl. du Mal. de Lattre de Tassigny, F-75016, Paris, France*
[e] *CEA-DRN/DMT/SERMA, CEN-Saclay, 91191 Gif sur Yvette Cedex, France*
[f] *CMAP, Inria-Saclay and X-Ecole Polytechnique, Route de Saclay, 91128 Palaiseau Cedex, France*

A R T I C L E   I N F O

Article history:
Received 7 March 2014
Received in revised form 8 July 2014
Accepted 10 August 2014
Available online 28 August 2014

Keywords:
Parareal in time algorithm
Time-dependent neutron diffusion equations
High performance computing

A B S T R A C T

In this paper we present a time-parallel algorithm for the 3D neutrons calculation of a transient model in a nuclear reactor core. The neutrons calculation consists in numerically solving the time dependent diffusion approximation equation, which is a simplified transport equation. The numerical resolution is done with finite elements method based on a tetrahedral meshing of the computational domain, representing the reactor core, and time discretization is achieved using a θ-scheme. The transient model presents moving control rods during the time of the reaction. Therefore, cross-sections (piecewise constants) are taken into account by interpolations with respect to the velocity of the control rods. The parallelism across the time is achieved by an adequate use of the parareal in time algorithm to the handled problem. This parallel method is a predictor corrector scheme that iteratively combines the use of two kinds of numerical propagators, one coarse and one fine. Our method is made efficient by means of a coarse solver defined with large time step and fixed position control rods model, while the fine propagator is assumed to be a high order numerical approximation of the full model.
The parallel implementation of our method provides a good scalability of the algorithm. Numerical results show the efficiency of the parareal method on large light water reactor transient model corresponding to the Langenbuch–Maurer–Werner benchmark.

1. Introduction

Accurate knowledge of the time-dependent spatial flux density in nuclear reactors is required for nuclear safety and design. The motivation behind the development of methods for solving the energy-, space-, and time-dependent kinetics equations is not only the challenge of developing a method for solving a large set of coupled partial differential equations,

* Corresponding author.
E-mail addresses: anne-marie.baudron@cea.fr (A.-M. Baudron), jean-jacques.lautard@cea.fr (J.-J. Lautard), maday@ann.jussieu.fr (Y. Maday), riahi@cmap.polytechnique.fr (M.K. Riahi), salomon@ceremade.dauphine.fr (J. Salomon).
URLs: http://www.ljll.math.upmc.fr/~maday (Y. Maday), http://www.cmap.polytechnique.fr/~riahi (M.K. Riahi), http://www.ceremade.dauphine.fr/~salomon (J. Salomon).

http://dx.doi.org/10.1016/j.jcp.2014.08.037
0021-9991/© 2014 Elsevier Inc. All rights reserved.

but also a real need to predict the performance and assess the safety of large commercial reactors, both those presently operating and those being designed for the future.

Modern reactor core design and safety depend heavily on the simulation of the reactor core and plants dynamics as well as their mutual interaction. Significant progress has been made during the last fifteen years in developing accurate techniques to simulate the computationally expensive reactor core models. Modeling the reactor core involves solving a large set of coupled time-dependent partial differential equations (PDEs), where the exact kinetic transport equation is simplified to a multi-group diffusion approximation equation. This model of neutron transport provides a scientific insight and is sufficiently realistic to study the energy of the reactor core for long time scale. The time-dependent multi-group neutron diffusion equation is used to model the scalar flux density. In the time dependent form, we take into account the delayed dynamic of neutrons caused by the presence of so-called *precursors*. Control rods are inserted to absorb neutrons and control the energy during the reaction.

Due to the limitation of the read–write memory in serial computers, it is relevant to propose parallel methods, which solve these large scale system with massively parallel computers. Much successful work has been done in the parallelization of neutron model simulations. For instance [1] studies the static case (eigenvalue problems) with space domain decomposition methods. A very nice strategy employed in [2] and [3] uses quasi-stationary approach to accelerate the simulation.

This paper focuses on neutrons behavior. We investigate the application of the parareal in time algorithm [4,5] on the neutron diffusion equation that governs the time-dependent flux density in the reactor core. The parareal in time algorithm is an iterative scheme, which improves computational time with parallel simulation. In several cases, parareal in time algorithm gives impressive rates of convergence, this is the case for example for linear diffusion equations and also for non-linear case [6]. Stability and convergence results for this algorithm are given in [7–11] particularly for diffusion system and others. The algorithm remains efficient in parallel computer simulation. We find a variety [8,12–24] of versions of this scheme that adapt the original algorithm to tackle new settings. Furthermore the parareal algorithm can be easily coupled with other iterative schemes such as domain decomposition methods (DDM) for instance the basic Schwarz algorithm or more complex ones [25], and optimal control based steepest descent algorithms [26–28].

The paper is organized as follows: After this introduction, we present the model of the kinetics of neutrons inside the reactor core. Section 3 gives a brief introduction of the parareal in time algorithm. Numerical tools are presented in Section 4 and adapt the parareal algorithm to the resolution of the handled problem. Finally, in Section 5, we present and discuss the numerical experiments that demonstrate the speedup following the fully-parallel implementation of the parareal algorithm in a parallel architecture.

2. Model

The neutron dynamics in a reactor core are governed by the kinetic transport Boltzmann equation [29]. The solution to this equation, denoted by Ψ, represents the directional neutron flux. It is a function of time t, the position $\vec{r}$ within the reactor core $\mathcal{R} \subset \mathbb{R}^3$, and the velocity of neutrons $\vec{V} = \sqrt{2E/m}\vec{\Omega}$, where $\vec{\Omega}$ is a unit vector indicating the direction of the velocity, E stands for the energy of the neutron and m for its mass. For computational reason, a simplification of the model has been proposed in [29, Chap. XXI, Section 5] that consists in averaging over the velocity directional variable leading to the introduction of the new function $\phi(\vec{r}, t, E) = \frac{1}{4\pi}\int_S \Psi(\vec{r}, t, E, \vec{\Omega})\, d\vec{\Omega}$ where S is the unit sphere. This method leads to accurate results in standard cases, unfortunately the computational time remains excessively long. Further simplifications consist in also averaging over the energy variable: the energy interval $[E_{\min}, E_{\max}]$ is divided into $\hat{g}$ non-overlapping intervals around a set of discrete energies $\{E^g\}_{g=1}^{\hat{g}}$ and leads to a new unknown $\Phi \equiv \Phi(\vec{r}, t) = \{\phi^g(\vec{r}, t)\}_{g=1}^{\hat{g}}$ composed of the set of the neutron average flux over each subinterval around E^g. This approach is known as the multi-group theory [30] where for each energy group $g = 1, \ldots, \hat{g}$ and any position $\vec{r} \in \mathcal{R} \subset \mathbb{R}^3$, the equations are a set of coupled three-dimensional multi-energy-group neutron kinetics equations involving time delayed contributions due to the fission of some isotope precursors, denoted by $C \equiv C(\vec{r}, t) = \{C^k(\vec{r}, t)\}_{k=1}^{K}$. The set of partial differential equation that govern the kinetics of neutrons in the reactor core is

$$\begin{cases} \frac{1}{V^g}\frac{\partial}{\partial t}\phi^g(\vec{r}, t) = \operatorname{div}(D^g \nabla \phi^g(\vec{r}, t)) - \Sigma_t^g \phi^g(\vec{r}, t) + \chi_p^g \sum_{g'=1}^{\hat{g}} (1 - \beta^{g'})\nu^{g'} \Sigma_f^{g'} \phi^{g'}(\vec{r}, t) \\ \qquad + \sum_{g'=1}^{\hat{g}} \Sigma_s^{g' \to g} \phi^{g'}(\vec{r}, t) + \sum_{k=1}^{K} \chi_d^{k,g} \lambda_k C^k(\vec{r}, t), \quad \forall g \in \{1..\hat{g}\},\ t \in [0, T] \text{ and } \vec{r} \in \mathcal{R}, \\ \phi^g(\vec{r}, t) = 0 \quad \text{on the boundary of the reactor core: } \forall g \in \{1..\hat{g}\},\ t \in [0, T],\ \vec{r} \in \partial\mathcal{R}, \\ \phi^g(\vec{r}, 0) = \phi_0^g(\vec{r}) \quad \text{the initial condition: } \forall g \in \{1..\hat{g}\},\ \vec{r} \in \mathcal{R}. \end{cases} \tag{1}$$

The delayed neutron concentrations satisfy $\forall k \in \{1..K\}$:

$$\frac{\partial C^k}{\partial t}(\vec{r}, t) = -\lambda_k C^k(\vec{r}, t) + \sum_{g'=1}^{\hat{g}} \beta^{k,g'} \nu^{g'} \Sigma_f^{g'} \phi^{g'}(\vec{r}, t), \quad t \in [0, T] \text{ and } \vec{r} \in \mathcal{R}. \tag{2}$$

In Eqs. (1) and (2), the neutron velocity is defined as $V^g = \sqrt{2E^g/m}$, the diffusion coefficient is denoted by D^g. The total and production cross-sections are denoted by Σ_t^g and $\nu\Sigma_f^g$ respectively, and $\Sigma_s^{g' \to g}$ stands for the transfer cross-section from energy group g' to energy group g. The fission spectra of prompt and delayed neutrons are denoted respectively by χ_p^g, $\chi_d^{k,g}$. The concentration of each precursor group is denoted by C^k and their delay fraction and decay constant are denoted by $\beta^{k,g}$ and λ_k respectively. The total delay fraction is denoted by β^g, where $\beta^g = \sum_{k=1}^K \beta^{k,g}$. For further details and analysis of the model we refer to [31] and reference therein.

For the sake of simplicity, the time dependent diffusion equation is written in terms of operators:

$$\frac{1}{V^g}\frac{\partial}{\partial t}\phi^g(t) = [F_\beta \Phi]_g(t) - [M\Phi]_g(t) + Q_d^g C(t), \tag{3}$$

with the appropriate boundary conditions. In this equation the solution $[F_\beta \Phi]_g(t) := \chi_p^g \sum_{g'=1}^{\hat{g}} (1-\beta^{g'})\nu^{g'}\Sigma_f^{g'}\phi^{g'}(t)$ represents the prompt fission source, while $[M\Phi]_g(t) := -\mathrm{div}(D^g \nabla \phi^g(t)) + \Sigma_t^g \phi^g(t) - \sum_{g'=1}^{\hat{g}} \Sigma_s^{g'\to g}\phi^{g'}(t)$ represents the removal of neutrons via net leakage of neutrons to other points in the reactor plus absorption and scattering. The operator $Q_d^g C(t)$ represents the delayed neutrons, and is given by $Q_d^g := (\chi_d^{1,g}\lambda_1, \ldots, \chi_d^{K,g}\lambda_K)$. The multi-group representation associated with Eq. (3) reads:

$$\left[\frac{1}{V}\right]\frac{\partial}{\partial t}\Phi(t) = (\mathbf{F}_\beta - \mathbf{M})\Phi(t) + \mathbf{Q}_d C(t), \tag{4}$$

where $[\frac{1}{V}] = Blockdiag(\frac{1}{V^1}, \ldots, \frac{1}{V^{\hat{g}}})$, the multi-group fission matrix operator $\mathbf{F}_\beta = (\chi_p^1, \ldots, \chi_p^{\hat{g}})^T((1-\beta^1)\nu^1\Sigma_f^1, \ldots, (1-\beta^{\hat{g}})\nu^{\hat{g}}\Sigma_f^{\hat{g}})$, the multi-group removal matrix operator $\mathbf{M} := \mathbf{L} - \mathbf{N}$, where $\mathbf{L} := Blockdiag(-\nabla.D^1\nabla + \Sigma_t^g, \ldots, -\nabla.D^{\hat{g}}\nabla + \Sigma_t^g)$, $\mathbf{N} := \sum_{ij}\Sigma_s^{j\to i}$, and the operator related to delayed neutrons is given by $\mathbf{Q}_d := ((Q_d^1)^T, \ldots, (Q_d^{\hat{g}})^T)^T$.

The energy of the reactor core is computed as the sum over g of the squared $L^2(\mathcal{R})$-norms of the fluxes ϕ^g; which are solutions of the neutron model corresponding to Eq. (1). This energy remains a function of time, where its evolution is essentially caused by the chain reaction of the neutrons fission. In the reactor core this fission chain reaction produces new neutrons exponentially in time, which are responsible for the neutron distribution and hence the energy of the reactor. In order to control this effect, control rods that absorb neutrons are sequentially inserted and withdrawn inside the reactor core. This action ensures the stability of the reaction during the production of electricity.

The simulation of the neutron model generally starts from an equilibrium average flux density, which is characterized by a steady solution of Eq. (1) where the unique source of neutrons considered is fission and we disregard the contribution of the delayed neutron i.e. β^g and $\chi_d^{k,g}$ are assumed nulls. Formally, the energy of distributed neutron population at each point of the domain must be exactly equal to the energy of the neutrons eliminated at this point, including leakage. In this case the neutron balance equation reduces to

$$\mathbf{M}\Phi = \mathbf{F}_0\Phi, \tag{5}$$

and the reactor is therefore critical. In the above equation $\mathbf{F}_0 = \mathbf{F}_\beta(\beta = 0)$. It is worth noting that the knowledge of the material properties is never perfect, so in order to obtain a non-trivial solution, it is common to introduce an eigenvalue $\Lambda \in \mathbb{R}$, which multiplies the fission source term in Eq. (5), which becomes appropriate for the static diffusion equation. The eigenvalue problem reads: find Λ and Φ_Λ such that

$$\mathbf{M}\Phi_\Lambda = \Lambda\mathbf{F}_0\Phi_\Lambda.$$

The closer Λ is to 1, the closer the system is to being critical. In practice it is never equal to unity, and so we will proceed by normalization as follows:

1. Compute the largest eigenpair $(\Lambda_{\max}, \Phi_{\Lambda_{\max}})$,
2. Update the production operator $\mathbf{F}_{0,\Lambda_{\max}} = \Lambda_{\max}\mathbf{F}_0$.

In the reactor physics literature the eigenvalue k_{eff} is known as the effective multiplication factor and is defined as $k_{\text{eff}} = 1/\Lambda_{\max}$. It is easy to observe that by computing the largest eigenpair $(k_{\text{eff}}, \Phi_{k_{\text{eff}}})$ of the new eigenvalue problem:

$$\mathbf{M}\Phi = \frac{1}{k_{\text{eff}}}\mathbf{F}_{0,\Lambda_{\max}}\Phi, \tag{6}$$

we simply obtain $k_{\text{eff}} = 1$. Note that the normalization at step 2 concerns the cross-sections. The multiplication of the balance equation at the equilibrium by the multiplication factor provides control of the equilibrium of the flux density. This feature will be very useful in the design of our algorithm. The solution $\Phi_{k_{\text{eff}}=1}$, associated with the largest eigenvalue $k_{\text{eff}} = 1$, is used as the initial condition to the normalized version of Eq. (1). The associated $C_{k_{\text{eff}}=1}$, corresponding to $\Phi_{k_{\text{eff}}=1}$ through the steady state associated with Eq. (2), is chosen as the initial condition for the precursor concentrations.

For simplicity, from now on, the normalized production operator $\mathbf{F}_\beta$ is identified with $\mathbf{F}_{\beta,\Lambda_{\max}}$, and hence Eqs. (1) and (2) are assumed with a normalized fission term.

3. The parareal in time algorithm

The parareal in time algorithm [4] is a "divide and conquer" method that enables parallelization across the time direction. Following the same strategy as in domain decomposition methods, the parareal in time algorithm is based on breaking up the time interval of simulation into subintervals and solve over each subinterval independently using different processors by updating iteratively the initial condition over each subinterval. The time evolution problem is thus broken up into a series of independent evolution problems on smaller time intervals. The parareal in time algorithm can be presented as a predictor corrector process [6,24], and also as a multi-shooting algorithm also as a kind of Newton method with a time coarse grid defining the Jacobian matrix [16]. Many improvements on the method, in particular for efficient iterative solution procedure on parallel architectures have been proposed. For some examples of these improvements see [23,32].

3.1. Propagation in time

To present our algorithm, we first rewrite Problems (1) and (2) in the following compact form:

$$\underbrace{\frac{\partial}{\partial t}\begin{bmatrix} \Phi \\ C \end{bmatrix}(t)}_{\dot{y}(t)} = \underbrace{\begin{bmatrix} \mathbf{F}_\beta - \mathbf{M} & \mathbf{Q}_d \\ \mathbf{F}^c_\beta & \mathbf{J} \end{bmatrix}}_{\mathbf{A}(t)} \underbrace{\begin{bmatrix} \Phi \\ C \end{bmatrix}(t)}_{y(t)}, \quad \text{on } [0,T] \times \mathcal{R}, \tag{7}$$

which is completed with the initial condition $y(t=0) = (\Phi_{k_{\text{eff}}=1}, C_{k_{\text{eff}}=1})^T$. In the above equation $\mathbf{J} := \mathit{Blockdiag}(\lambda_1, \ldots, \lambda_K)$ and $\mathbf{F}^c_\beta := (\ell^T_{\beta,1}, \ldots, \ell^T_{\beta,K})^T$ when $\ell_{\beta,k} := (\beta^k \nu^1 \Sigma^1_f, \ldots, \beta^{k,\hat{g}} \nu^{\hat{g}} \Sigma^{\hat{g}}_f)$.

The dependence in time of the operator $\mathbf{A}(t)$ is due to the possible change of cross-sections values where the reaction occurs. The existence of a solution to this problem is proven in [29, Chap. XXI, Section 2]. It can be written thanks to a flow map as follows

$$\forall t \geq 0,\ \forall \tau > 0, \quad y(t+\tau) = \mathcal{E}^t_\tau\big(y(t)\big), \tag{8}$$

where the uniqueness of the solution provides the semi-group property of the propagator $\mathcal{E}$ i.e.

$$\mathcal{E}^{t+\tau}_{\tau'} \circ \mathcal{E}^t_\tau = \mathcal{E}^t_{\tau+\tau'}.$$

3.2. Parareal iterations

Starting from the general formulation of Eq. (7), we consider a uniform partition of the time interval into N subintervals $[t_n, t_{n+1}]$, such that $0 = t_0 < t_1 < .. < t_n < t_{n+1} < .. < t_N = T$, and denote $\Delta t = t_{n+1} - t_n$, so $t_n = n\Delta t$. Based on the semigroup property stated in Eq. (8), we have

$$y(t_{n+1}) = \mathcal{E}^{t_n}_{\Delta t}\big(y(t_n)\big) = \mathcal{E}^{t_0}_{(n+1)\Delta t}(y_0) \quad \forall 0 \leq n \leq N-1.$$

In practice, we have to provide a fine enough numerical approximation of the propagator $\mathcal{E}$ denoted by $\mathcal{F}$ and called *fine propagator* in what follows. In our case, it is based on an appropriate classical backward Euler scheme used with a small time step. Let us denote by Y_n such a fine approximation to the solution of the Cauchy problem Eq. (7) at time t_n i.e. $Y_n \simeq y(t_n)$. The sequence of solution $(Y_n)^{n=N-1}_{n=1}$ is a solution of

$$Y_{n+1} = \mathcal{F}^{t_n}_{\Delta t}(Y_n), \quad \forall 0 \leq n \leq N-1 \quad \text{with} \quad Y_0 = y_0.$$

This reflects the sequential nature of the time propagation: one needs to know first the actual solution at time t_n to compute the solution at time t_{n+1}. To circumvent this problem, the parareal in time algorithm involves a *coarse propagator*, denoted by $\mathcal{G}$, which is a coarse approximation of $\mathcal{F}$. The propagator $\mathcal{G}$ is assumed to be faster than $\mathcal{F}$, in order to be able to rapidly carry out the sequential propagation of the solution from time t_0 to t_N. These propagators allow to define a sequence of approximate solutions $(Y^k_n)^{k>0}$ that converges to the right solution Y_n when k goes to infinity. Specifically, given an initial guess

$$Y^0_{n+1} = \mathcal{G}^{t_n}_{\Delta t}\big(Y^0_n\big), \quad \forall 0 \leq n \leq N-1, \qquad Y^0_0 = y_0, \tag{9}$$

the numerical scheme of the parareal method consists of the iteration (from k to $k+1$) knowing $(Y^k_n)_n$, compute $(Y^{k+1}_n)_n$ by the following

$$Y^{k+1}_{n+1} = \mathcal{G}^{t_n}_{\Delta t}\big(Y^{k+1}_n\big) + \mathcal{F}^{t_n}_{\Delta t}\big(Y^k_n\big) - \mathcal{G}^{t_n}_{\Delta t}\big(Y^k_n\big), \quad \text{with} \quad Y^k_0 = y_0. \tag{10}$$

The first contribution on the right hand side of Eq. (10) is the prediction for the propagated solutions at time t_{n+1}, whereas the rest of the right hand side corresponds to the correction scheme. The accurate solution associated with the accurate propagator $\mathcal{F}$ is used to correct the inaccurate solution predicted using the coarse propagator $\mathcal{G}$.

The definitions of the fine propagator $\mathcal{F}$ and the coarse one $\mathcal{G}$ are given in the next section.

4. Numerical method

In some cases, thanks to the property of symmetry in the reactor core, the computational domain is reduced to one of its quarters. Consequently, we consider Neumann boundary condition at these artificial interfaces, that is $\frac{\partial \phi^g}{\partial \vec{n}}(\vec{r}, t) = 0$, where $\vec{n}$ is the outward normal of the domain.

Numerical discretization of the neutron equation is briefly presented in this subsection. Eq. (1) is composed by several time dependent partial differential equations of order two in space and order one in time for the flux density with $\hat{g}$ energy group and K ordinary differential equations for the concentration of precursors. We first give the space discretization of those equations and then give the numerical scheme to approximate the time dependency.

The numerical scheme to approximate the space variable is based on a finite elements approximation, where the computational domain $\mathcal{R}$ (identified to its quarter) is meshed with tetrahedral elements. We use P1-Lagrange finite elements for the average flux and P0-Lagrange finite elements for both concentration of precursors and the physical parameters. In what follows, the space–time approximation of the global unknown $y(t_i)$, is denoted by y_i. We now assume the following partition of the time interval $[t_0, t_N] = \bigcup_{n=0}^{N-1}[t_n, t_{n+1}]$ where $[t_n, t_{n+1}]$ is also divided into I subintervals of equal size denoted by τ. Consequently we have $[t_0, t_N] = \bigcup_{n=0}^{N-1}\bigcup_{i=0}^{I-1}[t_{n,i}, t_{n,i+1}]$. With the obvious notations, $t_{n,0} = t_n$ and $t_{n,I} = t_{n+1}$. In the following the double subscript is dropped and the use of the subscript $\underline{i} = (n, i)$ stands for the index of any time $t_{\underline{i}}$, whereas the use of the subscript n stands only for time $t_n = n\triangle t$.

The evolution in time of the solution $y_{\underline{i}}$ is approximated using the θ-scheme, which assumes at time $t_{\underline{i}+1}$ the solution $y_{\underline{i}}$ is known. The solution at time $t_{\underline{i}+1}$ is thus computed by solving $y_{\underline{i}+1} - y_{\underline{i}} = \tau\theta \mathbf{A}_{\underline{i}+1} y_{\underline{i}+1} + \tau(1-\theta)\mathbf{A}_{\underline{i}} y_{\underline{i}}$, where $\theta \in (0, 1)$, and $\mathbf{A}_{\underline{i}}$ represents the approximation matrix of the operator $\mathbf{A}(t_{\underline{i}})$ at time $t_{\underline{i}}$ and we have used the notation $\underline{i}+1 = (n, i+1)$. Finite elements matrix representation can be found in, e.g., [33, Chap. 3]. The resolution in the forward time, with $\theta \neq 0$, requires matrix inversion at each time step, such that

$$y_{\underline{i}+1} = (I - \tau\theta \mathbf{A}_{\underline{i}+1})^{-1}\big(I + \tau(1-\theta)\mathbf{A}_{\underline{i}}\big) y_{\underline{i}}. \tag{11}$$

The efficient choice of the parameter θ is discussed in the experimental part.

More technical discussion concerning the fine propagation, in addition to the classical discretization scheme presented above, is given below.

4.1. Fine propagator $\mathcal{F}_{\Delta t}^{t_n}$

The numerical approximation of the transient Langenbuch–Maurer–Werner (LMW) benchmark application [34] requires a particular attention to the control rods motion, where cross-sections have to be interpolated in order to avoid oscillations and instabilities of the numerical solver. Similar cross-sections interpolation techniques are applied in [35] for the neutron nodal expansion method.

The cross-sections are approximated with piecewise constant functions. To take into account the unsteadiness of the balance neutron equation, which is caused by the immersion of control rods within the reactor, one could refine the mesh in order to follow the exact motion of the rods. This would be quite expensive. Another possibility would be to tune the time step with the velocity of the motion so that the ends of rods match the mesh. This adds strong constraints to the design of the discretization. To cope with this, we assume that the triangulation of the domain is extracted from a first decomposition of $\mathcal{R}$ into several horizontal layers of prisms each of them being subsequently cut into three tetrahedrons as explained in Fig. 1 (left). This enables interpolating the cross-sections, as sketched in Fig. 1 (right) where the immersion of the control rod into the tetrahedron is illustrated. The cross section (represented with color) are interpolated relatively with the volume occupied by control rods. In this case positive values between 0 and 1, is attributed to the cross section related to the rods, whereas the cross section related to fuel are deduced automatically by interpolating using the complement value. It is worth noting that this procedure is done before the finite elements matrix assembly. We update cross-sections at each time step using interpolation, then assemble the main matrix and solve Eq. (11). Hence the definition of the fine propagator $\mathcal{F}_{\Delta t}^{t_n}$ can be given as follows:

For a given set of solutions $y_{\underline{0}} \equiv y_{n,0} \equiv y_{\underline{0}}$ at each initial time t_n for the fine propagation, the solution at time t_{n+1} computed with the fine propagator is given by

$$\mathcal{F}_{\Delta t}^{t_n}(y_n) = \underbrace{(I - \tau\theta \mathbf{A}_{\underline{I}})^{-1}\big(I + \tau(1-\theta)\mathbf{A}_{\underline{I}-1}\big)\dots(I - \tau\theta \mathbf{A}_{\underline{0}+1})^{-1}\big(I + \tau(1-\theta)\mathbf{A}_{\underline{0}}\big)}_{\text{I times}}\, y_n.$$

4.2. Coarse propagator $\mathcal{G}_{\Delta t}^{t_n}$ using model reduction

In this subsection, we present and discuss a reduction of the previous model. The simplified system will be used in the parareal in time algorithm as a coarse solver, which is an inexpensive sequential solver. We shall neglect in Eq. (1) the coupling with precursors that are described by Eq. (2) in the fine model. Moreover, we simplify the model by reducing the motion of control rods to a two specific positions. This allows us to minimize computational complexity when updating the matrices according to the evolution of control rods position.

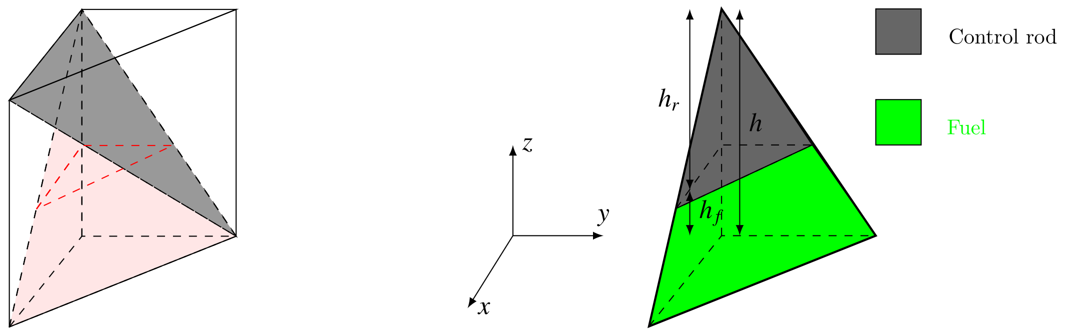


Fig. 1. P0-interpolation of the cross-sections over one simplex (tetrahedron). The triangulation of the domain $\mathcal{R} \subset \mathbb{R}^3$ is structured in the sense that each tetrahedron has one vertical edge. If h is the height of one tetrahedron, we denote the length of the penetration of control rods in the tetrahedron and the rest of the length by h_r and h_f respectively. The distance h_r and h_f are therefore perfectly determined using the velocity of control rods. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

Note that some care has to be taken so that the reduced system remains coherent with the exact dynamics. Indeed, the introduction of the effective multiplication factor k_{eff} (here different from the unity) allows us to tune the equations and further define *subcritical, critical* and *supercritical* states. Of course the motion of the rods changes the state of the core. Modifying the coarse model as we did substantially changes this equilibrium, both due to the fact that the delayed neutrons have a very important role in stability and because the damping of the constantly increasing energy produced by fission. The contribution of the concentrations of precursors is fundamental to access accurate information about flux density at time t during the reaction. Hence the absence of the delayed neutrons in the reaction behavior inside the reactor core reduces the damping and energy production is accelerated. It means that the rate of producing energy of the reduced model is therefore more important than the one of the fine model. One has to manipulate the velocity of neutrons in order to reach, approximately in time-reaction and quantity, the peak of the energy. Indeed, we tune the effective multiplication factor k_{eff} (in the normalized equations) to two values slightly larger and slightly smaller than one. Indeed, the first period of the benchmark corresponding to a supercritical nature is represented by a factor k_{eff} slightly larger than 1 whereas the second stage of the benchmark, associated with the rod going down, is represented by a k_{eff} slightly smaller than 1. For $0 < \iota \ll 1$, the dynamics of the discussed reduced model can be described with the equations

$$\begin{cases} \left[\frac{1}{V(\iota)}\right]\frac{\partial}{\partial t}\Phi(t) = \left(\frac{1}{1+\iota}\mathbf{F}_0 - \mathbf{M}\right)\Phi(t), & \text{for } t \in (0,24), \\ \left[\frac{1}{V(\iota)}\right]\frac{\partial}{\partial t}\Phi(t) = \left(\frac{1}{1-\iota}\mathbf{F}_0 - \mathbf{M}\right)\Phi(t), & \text{for } t \in (24,80), \\ \Phi(t=0) = \Phi_{\text{keff}=1}. \end{cases} \tag{12}$$

In addition to Eq. (12) control rods are fixed during the simulation time. The resulting reduced model produces a good approximate state of the reactor before and after the peak. This reduction enables us to make computation cheaper while keeping accurate results. We show this explicitly in Section 5. Eq. (12) is designed to be solved in serial and the matrix is assembled only once at $t = 0$. The flux density during simulation is managed by the change of the factor k_{eff} as explained above.

4.3. The algorithm

We are now in the position to fully describe the algorithm, which we present below in Algorithm 1. The time-parallel algorithm is implemented with a master–slave strategy for which we have two types of communications: a *distributive* and a *collective* operations. In the *distributive* operation, the main processor sends information towards all of its agent processors. On the other hand, in the *collective* operation; the master itself receives and collects information from its agents. In both cases, it is about the same quantity of information which passes in the two directions. The second type of communication is devoted to the correction of the coarse error, which requires fine information to be communicated by agent processors toward master. We use some keywords from the parallel programing language to describe communication procedures. For instance, **Recv**(*data, sender*) and **Send**(*data, receiver*) mean that the processors which execute those commands receives the *data* from the *sender* and send *data* to the *receiver*. The **Broadcast**(*sender, data*) command means that the processor *sender* sends *data* to all other processors.

5. Numerical experiments

In this section, we simulate the LMW benchmark [34], which is a kinetic model that describes the effect of control rod motion on the flux density. The LMW models the kinetics of neutrons as in Eq. (1), but with time and space dependent cross-section coefficients to take into account the rod motion. The LMW benchmark presented in Fig. 2, as a quarter of the

Algorithm 1: Parareal kinetics of neutrons.

```
Input:  $N := \#slave\,proc$, $\tau_F$, $\Delta t$
Input:  $Y_0^0 = (\Phi_{k_{\text{eff}}=1}, C_{k_{\text{eff}}=1})^T$ as initial conditions, $\epsilon^\infty$ a tolerance of the algorithm;
Input:  Solver $\mathcal{A}$, data vector $y$;
Routine ($\mathcal{A}$, $y$)
  1) Define the absorber rods position with respect to their dynamic chronology;
  2) Assemble matrices related to Eqs. (1)–(2);
  3) Solve iteratively in time the system of (11), the result is denoted by $\mathcal{A}y$;
end Routine;
$k \longleftarrow 0$;
repeat
    if master processor then
        foreach $n \in \{0, .., N-1\}$ do
            1) Call: Routine ($\mathcal{G}_{\Delta t}^{t_n}$, $Y_n^k$) (i.e. coarse-serial propagation);
            2)
              if $k = 0$ then
                repeat return to 1 with $Y_{n+1}^k$ until $n = N-1$
              else
                Construct $(Y_n^k)_{n\geq 1}$ with respect to Eq. (10);
              end
            3) Send ($Y_n^k$, $processor(n)$);
        end
    else
        forall the $slave\,processor(n)/n \in \{0, \dots, N-1\}$ do
            Recv ($Y_n^k$, $master\,processor$);
            Call: Routine ( $\mathcal{F}_{\Delta t}^{t_n}$, $Y_n^k$) (i.e. fine-parallel propagation);
            Send ($\mathcal{F}_{\Delta t}^{t_n} Y_n^k$, $processor(n)$);
        end
    end
    if master processor then
        foreach $n \in \{0, .., N-1\}$ do
            Recv ($\mathcal{F}_{\Delta t}^{t_n} Y_n^k$, $processor(n)$);
            Evaluate $\epsilon_{n+1}^k = \|\mathcal{F}_{\Delta t}^{t_n} Y_n^k - Y_{n+1}^k\|_2^2 / \|Y_{n+1}^k\|_2^2$;
        end
    end
    $k \leftarrow k+1$;
    Broadcast ($master\,processor$, $\epsilon_n^k$);
until $\max_n \epsilon_n^k \leq \epsilon^\infty$;
```

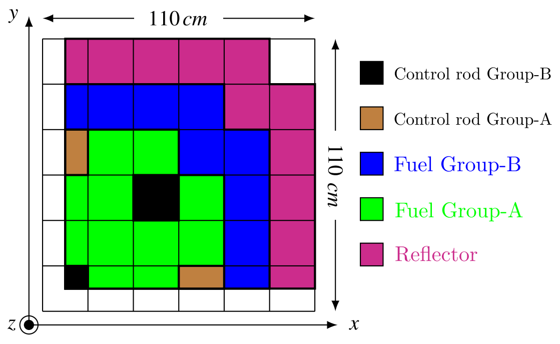


Fig. 2. LMW transient problem: cross-section configuration in regard to rods positioning. Horizontal cross-sections.

whole domain, initiates by withdrawing a bank of four partially inserted control rods inside the reactor core for certain time, after which, another bank of five control rods is inserted. The global transient time is about 60 s when the velocity of all banks of control rods is about 3 cm/s. As explained before, the rod motion model has important discrepancy on the cross-sections that should be interpolated in order to avoid errors and undesired oscillations on the solution. We give in Appendix A the cross-sections values of the LMW transient problem and discuss experiments involving parallelization across the time for the numerical simulation.

Table 1
Change of the k_{eff} factor with respect to the time-reaction.

	$t \in [0, 20]$	$t \in]20, 80]$
Group 1	$z = 160$ cm	$z = 180$ cm
Group 2	$z = 180$ cm	$z = 60$ cm
k_{eff}	10008.e−5	9998.e−5

5.1. The parameters of the cross-sections

The motion of control rods in the LMW benchmark application reads as follows: at time $t = 0$ s the first group of rods has an initial position at $z = 100$ cm, while the second group has a higher position at $z = 180$ cm. The velocity of the control rods motion is about 3 cm/s and is summarized as follows:

$$\text{Group 1: } (t = 0 \,|\, z = 100 \text{ cm}) \nearrow (t = 26,5 \,|\, z = 180 \text{ cm}),$$

$$\text{Group 2: } (t = 7,5 \,|\, z = 180 \text{ cm}) \searrow (t = 47,5 \,|\, z = 60 \text{ cm}).$$

Cross-sections of the fuel (Group-A and Group-B), control rods (Group-A and Group-B) and reflectors are given in Table 4 of Appendix A. Scattering cross-section for the different media are given in Table 5 of Appendix A. Precursors data are given in Table 6 of Appendix A.

5.2. Numerical tests and discussions

This section discusses numerical tests of the LMW parallel in time transient problem. We outline our discussion into four steps: We first give simulation results related to the direct application of the parareal in time algorithm on the model described in Section 2. Next we present results related to the reduced model where we disregard delayed neutrons on the initial model. Then, we couple the two models. We finally discuss the speed-up achieved with the algorithm.

The numerical simulations were carried out on a parallel shared memories machine, equipped with a 2.0 GHz 64-core processor, and 256 Gb of shared memory and a communication network Numalink (15 Gb/s). The parallelization of the procedure is carried out using MPI library and the scientific computation software FreeFem++ [36].

Since the analytical solution of (1) is not available, we produce a reference approximated numerical solution using a refined time step. Concerning space approximation, we use first order polynomial approximation with nodal P1-Lagrange finite elements, where the domain is provided with a fine enough mesh with tetrahedra, since the main focus here is about temporal error.

As described above, the parareal in time algorithm involves two types of solvers that alternate during the time process between coarse-sequential resolution and fine-parallel resolution of (1). In this paper the coarse solver is only based on a coarse time step, we refer to [22] for a consideration of a coarse time and space based solver with application to Navier–Stokes problem.

We have used a θ-scheme (see (11)) in order to approximate the time variation of the solution, where $\theta = 0$ leads to the forward Euler scheme and $\theta = 1$ leads to the backward Euler scheme of order 1. Numerical tests showed that the case $\theta = 1$ is the most stable scheme that provides an accurate solution without oscillations on its power. We present in Fig. 3 the L^2 trajectory of the flux density solution of the LMW produced with the parareal in time algorithm, where the fine and coarse propagators only differ from the choices of time steps. The series of plots given in Fig. 3 represent the first four iterations of the parareal algorithm. We remark that convergence of the trajectory occurs in few iterations using parallel simulation. These results show the evolution of the energy production with respect to the simulated time of the reaction (in seconds). The energy is initialized with a value of 1.25e+6 representing the equilibrium of the reactor. We recall that a bank of four control rods already inserted in the reactor core at time $t = 0$ are withdrawn simultaneously and after 7.5 s another group of control rods is inserted. The power of the reactor (presented here as L^2 norm of the neutron flux density) achieves a peak before decreasing from the effect of the neutron absorption by the inserted rods.

The parallel simulation of the reduced model is presented in Fig. 4. We recall here that delayed neutrons don't contribute in the solution. The fine and coarse propagators solve here Eq. (12) and only differ from the choices of time steps. We test and study the sensitivity of the model with respect to the reduction of the involved unknown and the sudden change of the k_{eff} factor. This change of k_{eff} allows to avoid interpolating cross-sections at each position change when dealing with rods motions. The global matrix is assembled only once where rods are fixed on their initial positions. Table 1 shows the corresponding change of the effective multiplicity factor with respect to the desired behavior of the reactor core.

We now show numerical convergence results. In Fig. 5 we present the convergence of the algorithm with a threshold ϵ taken as the maximum among ϵ_n^k, where,

$$\epsilon^k := \max_n \epsilon_n^k, \quad \text{for } \epsilon_n^k = \frac{\|Y_n^k - Y_n\|_2^2}{\|Y_n\|_2^2},$$

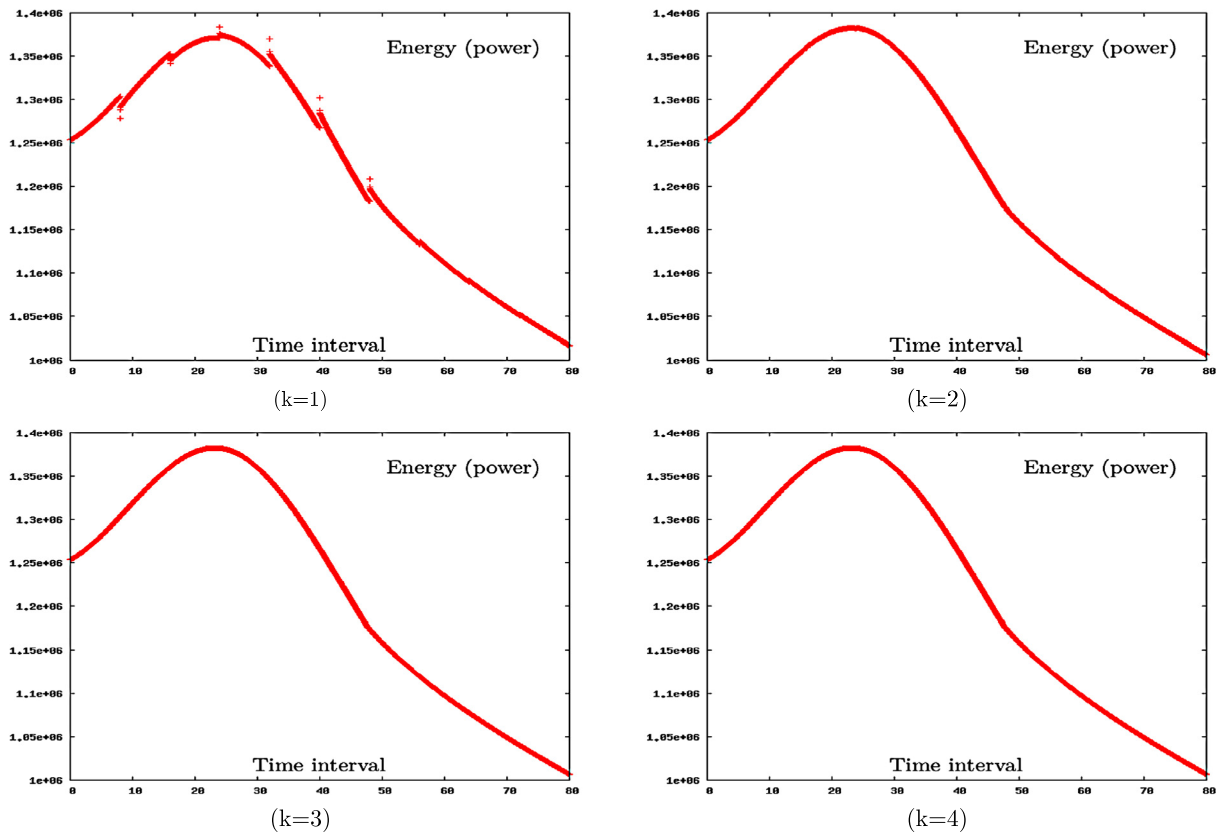


Fig. 3. Power of the nuclear reactor, the first four iterations of the parareal in time algorithm for the LMW transient model: $\tau = 1.e{-}1$, $\Delta t = 4$ and $N = 10$.

Table 2
Iterations 1, 2, 3 and 4 of the algorithm, complete model with dynamic rods scenario.

τ	Δt	$\max_{n\geq 0} \epsilon_n^1$	$\max_{n\geq 0} \epsilon_n^2$	$\max_{n\geq 0} \epsilon_n^3$	$\max_{n\geq 0} \epsilon_n^4$
0.01	2	1.42e−02	3.04e−05	1.73e−05	1.12e−06
0.01	4	2.93e−02	1.16e−03	1.22e−04	1.46e−05
0.01	8	6.09e−02	4.11e−03	7.85e−04	1.43e−04
0.1	0.5	2.92e−03	1.31e−05	1.41e−07	2.08e−09
0.1	1	6.53e−03	6.48e−05	1.67e−06	5.12e−08
0.1	2	1.35e−02	2.81e−04	1.51e−05	9.22e−07
0.1	4	2.88e−02	1.11e−03	1.13e−04	1.31e−05
0.1	8	6.04e−02	4.02e−03	7.53e−04	1.35e−04
0.5	2	1.13e−02	1.71e−04	7.02e−06	3.35e−07
0.5	4	2.68e−02	0.80e−04	7.80e−05	7.92e−06
0.5	8	5.83e−02	3.55e−03	6.17e−04	1.02e−04
1	2	7.58e−03	7.42e−05	1.94e−06	6.09e−08
1	4	2.29e−02	6.30e−04	4.64e−05	3.91e−06
1	8	5.43e−02	3.01e−03	4.77e−04	7.12e−05

where Y_n is assumed to be the reference numerical solution, and Y_n^k represents the numerical parallel solution produced with the parareal in time algorithm. We consider the convergence of the parareal in time algorithm to an error ϵ^∞ of order 10^{-3} which is consistent with the fact that the reference backward Euler time-discretization scheme is first order and has been implemented with a time step 10^{-2}.

Tables 2 and 3 illustrate the robustness of the methods with different choices of time-steps discretization.

The final illustration deals with the main contribution of this paper for the parareal in time algorithm where the two models are used: the fine propagator solves Eq. (7) with a fine time step and the coarse propagator solves Eq. (12) with a coarse time step. The convergence results related to this combination of the initial model and the reduced model are given in Fig. 6. This procedure accelerates the computational time. Indeed, only one multiplication of the matrix (already in memory) by an adequate real coefficient is sufficient to reproduce the new matrix related to the operator $\mathbf{A}(\tau)$ at a desired time τ.

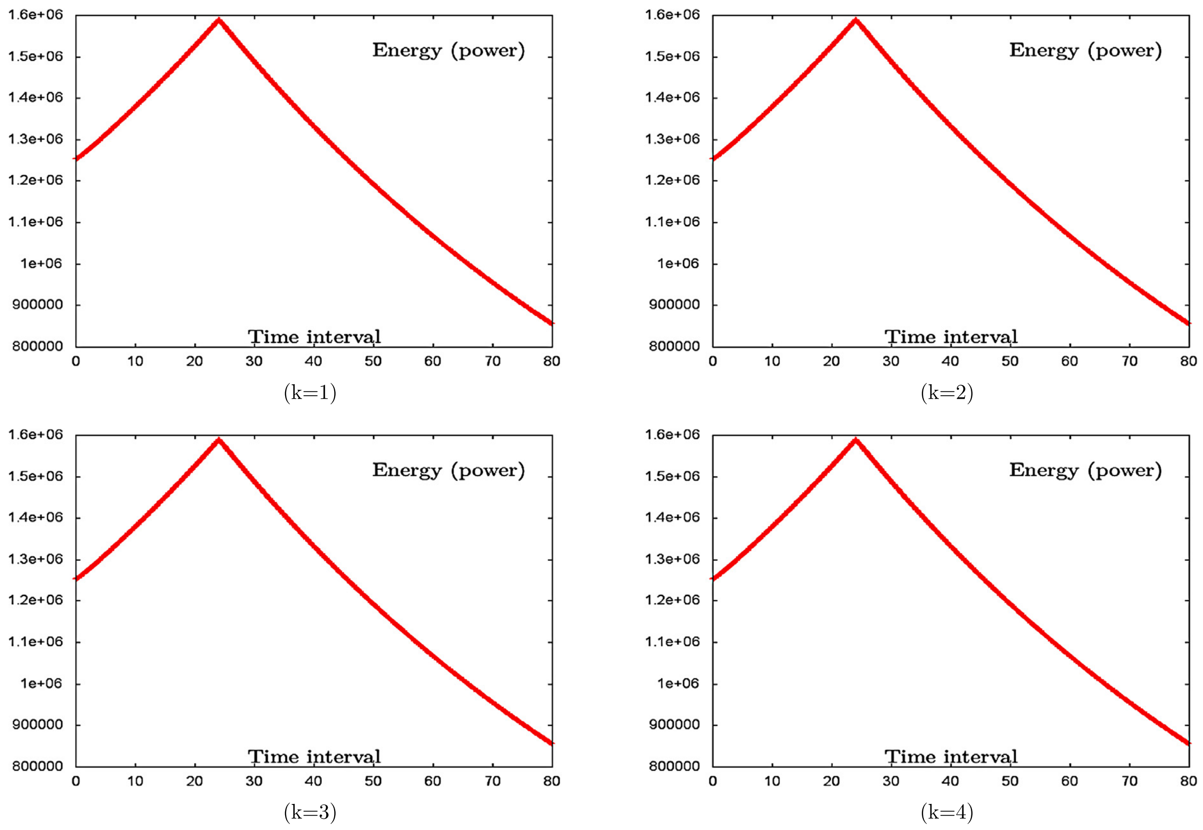


Fig. 4. Power of the nuclear reactor, the first four iterations of the parareal in time algorithm of the reduced neutron model. $\tau = 1$, $\Delta t = 4$ and $N = 10$.

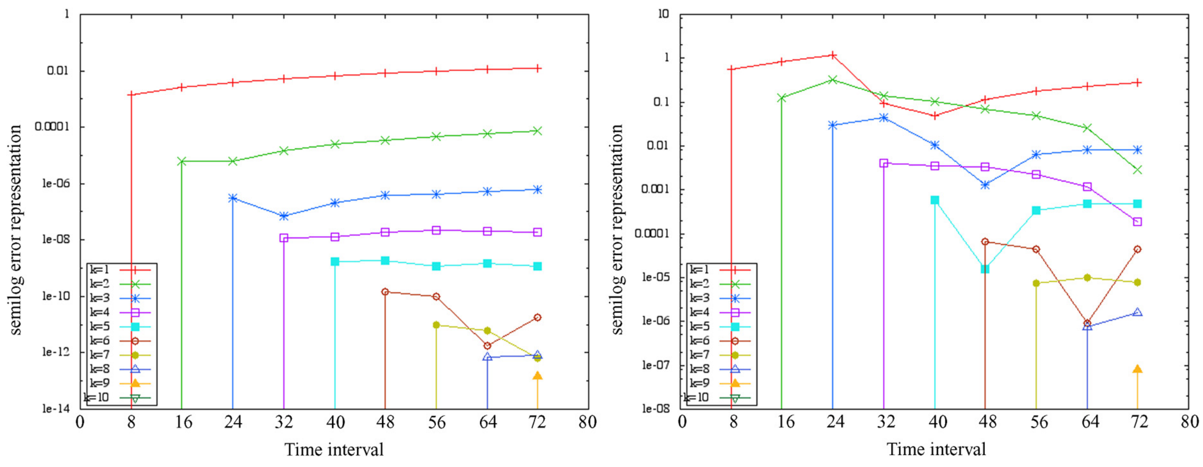


Fig. 5. Convergence of the algorithm (iterations $1 \to 10$) that couple the LWM transient model with the reduced model: (left) $\tau = 1.e{-}1$, $\Delta t = 2$ and $N = 10$ (right) $\tau = 1.e{-}1$, $\Delta t = 4$ and $N = 10$. Note that the error lines start at time $(k+1)T/N$ with vertical line. This occurs because the errors before that time is vanishing, indeed the parallel solution is exactly equal to the serial solution at time $t < (k+1)T/N$.

The parallel implementation of the parareal in time algorithm we have presented in this paper speeds up the resolution with a good efficiency using 16 processor unit (see Fig. 7). We refer to [37] for alternative parallel implementations. It is worth noticing that, even though the parareal in time algorithm shares some similarities with the parallel domain decomposition method, the scaling cannot be as good since the work per subdomain (in time) is proportional to the size of the subdomain, in opposition to what happens with spacial domain decomposition where the work is super linear, hence the gain by dividing is larger than the number of subdomains. That is the reason why, in order to get a full speed up, the parareal algorithm should be coupled with other iterative techniques (see [38]).

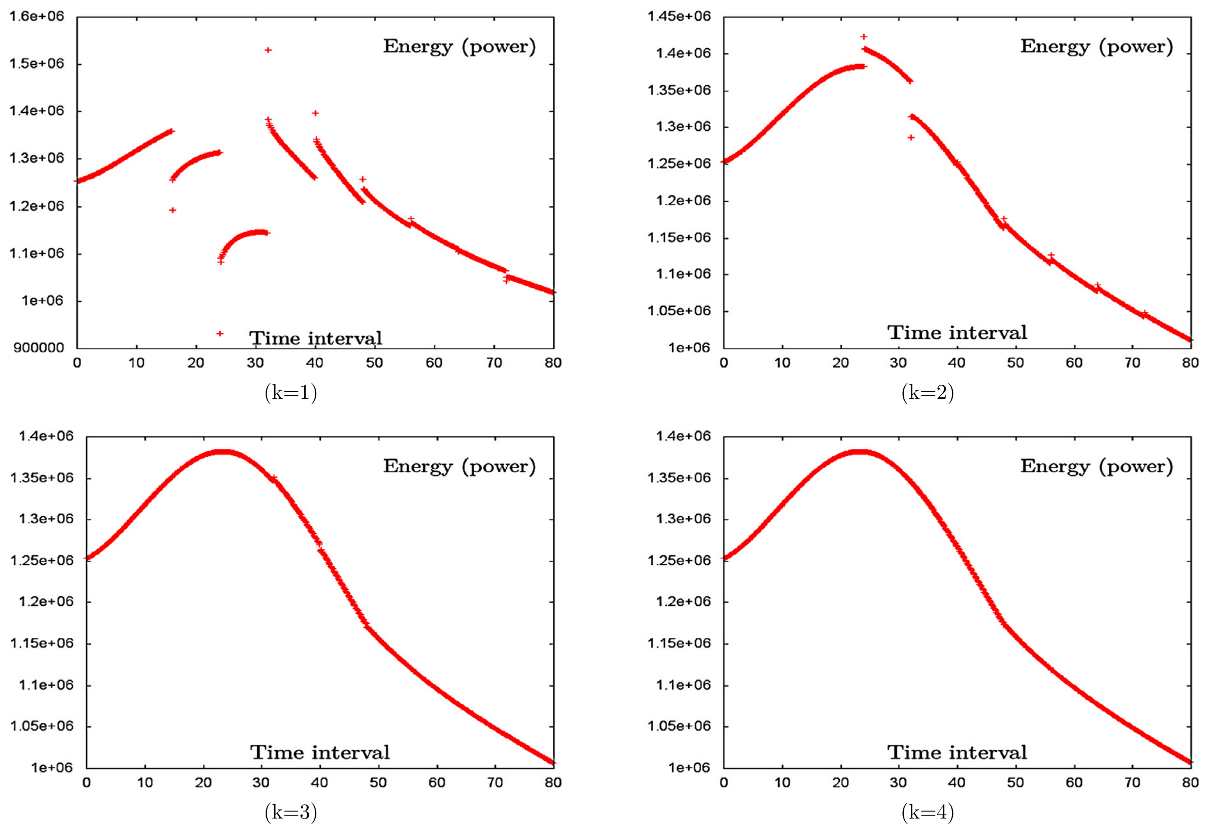


Fig. 6. Power of the nuclear reactor, the first four iterations of the parareal in time algorithm for the LMW transient model with the reduced neutron model. Coarse time step is also used for the reduced model: $\tau = 1.e{-}1$, $\Delta t = 2$ and $N = 10$.

Table 3
Iterations 1, 2, 3 and 4 of the algorithm, reduced model with dynamic rods scenario.

τ	Δt	$\max_{n\geq 0} \epsilon_n^1$	$\max_{n\geq 0} \epsilon_n^2$	$\max_{n\geq 0} \epsilon_n^3$	$\max_{n\geq 0} \epsilon_n^4$
0.1	2	1.39e−02	5.19e−05	1.91e−07	5.71e−09
0.1	4	2.78e−02	1.88e−04	1.74e−06	1.52e−07
0.1	8	5.34e−02	5.64e−04	1.63e−05	2.53e−06
0.5	2	1.09e−02	3.17e−05	1.01e−07	2.69e−09
0.5	4	2.48e−02	1.47e−04	1.30e−06	7.39e−08
0.5	8	5.03e−02	4.91e−04	1.21e−05	2.17e−06
1	2	7.22e−02	1.33e−05	3.18e−08	6.96e−10
1	4	2.10e−02	1.03e−04	8.35e−07	5.50e−08
1	8	4.65e−02	4.06e−04	9.42e−06	1.86e−06

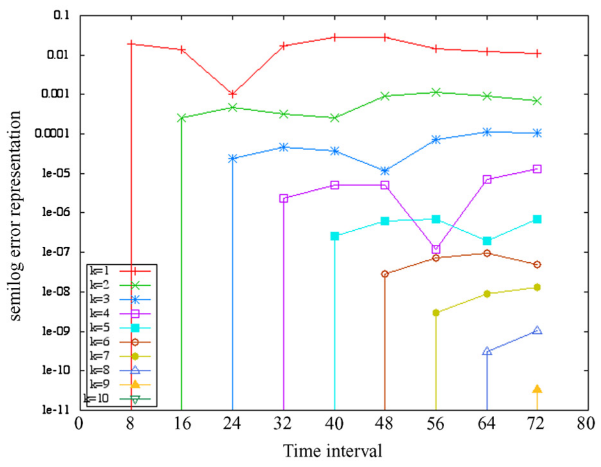
6. Conclusion

We have presented in this paper an application of the parareal algorithm to the parallelization across the time of the simulation of the neutron diffusion multi-group kinetics equations. In order to improve computational time, the model is reduced by the use of an adequate coarse solver based on several properties of the steady solution. Numerical results show that the algorithm speeds up the simulation and converges quite fast, when the stopping criterion is in most cases reached in two or three iterations of the parareal algorithm.

Acknowledgements

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions to improve the quality of the paper. They also gratefully acknowledge the support of LRC-MANON,[1] which also supported the forth author

[1] Joint research program MANON between CEA-Saclay and University Pierre and Marie Curie-Paris 6.



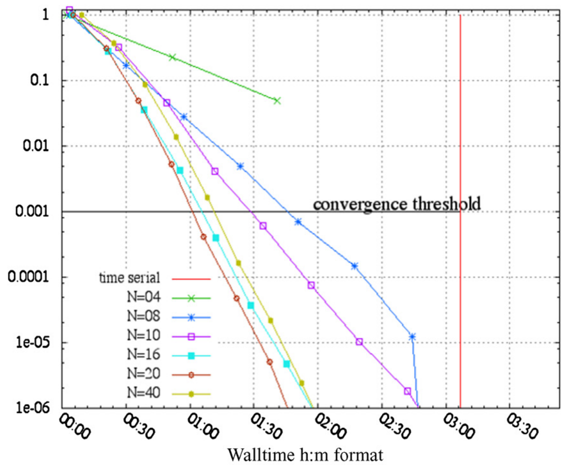


Fig. 7. (Left) Convergence of the algorithm (iterations $1 \rightarrow 10$) for the LWM transient problem coupled with the reduced model: $\tau = 1.\text{e}{-1}$, $\Delta t = 4$ and $N = 10$. (Right) Wall-time simulation (format: h:m) parareal in time algorithm and the serial one, we vary the number of used processor/subinterval; $\tau = 1.\text{e}{-1}$, $\Delta t = 2$ and $N \in \{1, 4, 8, 10, 16, 20, 40\}$.

during the preparation of a part of his PhD thesis [33, Chap. 3] in 2011. The fifth author was partially supported by the "Agence Nationale de la Recherche" (ANR), Project Blanc EMAQS number ANR-2011-BS01-017-01.

Appendix A

For the sake of convenience, we recall here the various physical constants used in the numerical simulation of the Langenbuch–Maurer–Werner (LMW) benchmark [34].

Table 4
Total and fission cross-sections of the LMW 3d benchmark.

Physical data	Medium			
Cross-sections	Fuel A		Fuel B	
	Group-1	Group-2	Group-1	Group-2
Σ_t	0.23409670	0.93552546	0.23381787	0.95082160
Σ_f	0.006477691	0.1127328	0.007503284	0.1378004
Celerity V^g	1.25e+7	2.5e+5	1.25e+7	2.5e+5
	Rods		Reflector	
	Group-1	Group-2	Group-1	Group-2
Σ_t	0.23409670	0.93552546	0.20397003	1.26261670
Σ_f	0.006477691	0.1127328	0.0	0.0
Celerity V^g	1.25e+7	2.5e+5	1.25e+7	2.5e+5

Table 5
"Scattering" cross-sections data.

Fuel A

$\Sigma_s^{g' \to g}$	Group-1	Group-2
Group-1	0.20613914	0.01755550
Group-2	0.0	0.84786329

Fuel B

$\Sigma_s^{g' \to g}$	Group-1	Group-2
Group-1	0.20564756	0.01717768
Group-2	0.85156526	0.0

Rods of control

$\Sigma_s^{g' \to g}$	Group-1	Group-2
Group-1	0.20558914	0.01755550
Group-2	0.84406329	0.0

Reflector

$\Sigma_s^{g' \to g}$	Group-1	Group-2
Group-1	0.17371253	0.02759693
Group-2	1.21325319	0.0

Table 6
Precursors data.

Precursor	Group-1	Group-2	Group-3	Group-4	Group-5	Group-6
$\lambda(s^{-1})$	0.0127	0.0317	0.115	0.311	1.4	3.87
β	0.000247	0.0013845	0.001222	0.0026455	0.000832	0.000169

References

[1] P. Guérin, A.-M. Baudron, J.-J. Lautard, Domain decomposition methods for the neutron diffusion problem, Math. Comput. Simul. 80 (11) (2010) 2159–2167.
[2] M. Dahmani, A. Baudron, J. Lautard, L. Erradi, A 3D nodal mixed dual method for nuclear reactor kinetics with improved quasistatic model and a semi-implicit scheme to solve the precursor equations, Ann. Nucl. Energy 28 (8) (2001) 805–824.
[3] S. Chauvet, Multi-scale method for the resolution of the neutronic kinetics equations, Ph.D. thesis, Université de Nantes, 2008.
[4] J.-L. Lions, Y. Maday, G. Turinici, Resolution d'EDP par un schema en temps "parareel", C. R. Acad. Sci. Paris, Ser. I, Math. 332 (7) (2001) 661–668.
[5] Y. Maday, G. Turinici, The parareal in time iterative solver: a further direction to parallel implementation, in: Domain Decomposition Methods in Science and Engineering, in: Lect. Notes Comput. Sci. Eng., vol. 40, Springer, Berlin, 2005, pp. 441–448.
[6] G. Bal, Y. Maday, A "parareal" time discretization for non-linear PDE's with application to the pricing of an American put, in: Recent Developments in Domain Decomposition Methods, Zürich, 2001, in: Lect. Notes Comput. Sci. Eng., vol. 23, Springer, Berlin, 2002, pp. 189–202.
[7] G. Bal, On the convergence and the stability of the parareal algorithm to solve partial differential equations, in: Domain Decomposition Methods in Science and Engineering, in: Lect. Notes Comput. Sci. Eng., vol. 40, Springer, Berlin, 2005, pp. 425–432.
[8] M.J. Gander, E. Hairer, Nonlinear convergence analysis for the parareal algorithm, in: Domain Decomposition Methods in Science and Engineering XVII, in: Lect. Notes Comput. Sci. Eng., vol. 60, Springer, Berlin, 2008, pp. 45–56.
[9] D.S. Daoud, Stability of the parareal time discretization for parabolic inverse problems, in: Domain Decomposition Methods in Science and Engineering XVI, in: Lect. Notes Comput. Sci. Eng., vol. 55, Springer, Berlin, 2007, pp. 275–282.
[10] M.J. Gander, S. Vandewalle, Analysis of the parareal time-parallel time-integration method, SIAM J. Sci. Comput. 29 (2) (2007) 556–578 (electronic).
[11] G.A. Staff, E.M. Rønquist, Stability of the parareal algorithm, in: Domain Decomposition Methods in Science and Engineering, in: Lect. Notes Comput. Sci. Eng., vol. 40, Springer, Berlin, 2005, pp. 449–456.
[12] M. Sarkis, C.E. Schaerer, T. Mathew, Block diagonal parareal preconditioner for parabolic optimal control problems, in: Domain Decomposition Methods in Science and Engineering XVII, in: Lect. Notes Comput. Sci. Eng., vol. 60, Springer, Berlin, 2008, pp. 409–416.
[13] C. Farhat, J. Cortial, C. Dastillung, H. Bavestrello, Time-parallel implicit integrators for the near-real-time prediction of linear structural dynamic responses, Int. J. Numer. Methods Eng. 67 (5) (2006) 697–724.
[14] G. Bal, Q. Wu, Symplectic parareal, in: Domain Decomposition Methods in Science and Engineering XVII, in: Lect. Notes Comput. Sci. Eng., vol. 60, Springer, Berlin, 2008, pp. 401–408.
[15] S. Ulbrich, Generalized SQP methods with "parareal" time-domain decomposition for time-dependent PDE-constrained optimization, in: Real-Time PDE-Constrained Optimization, in: Comput. Sci. Eng., vol. 3, SIAM, Philadelphia, PA, 2007, pp. 145–168.
[16] M.J. Gander, S. Vandewalle, On the superlinear and linear convergence of the parareal algorithm, in: Domain Decomposition Methods in Science and Engineering XVI, in: Lect. Notes Comput. Sci. Eng., vol. 55, Springer, Berlin, 2007, pp. 291–298.
[17] L. He, The reduced basis technique as a coarse solver for parareal in time simulations, J. Comput. Math. 28 (5) (2010) 676–692.
[18] M. Gander, M. Petcu, Analysis of a Krylov subspace enhanced parareal algorithm for linear problems, in: Paris-Sud Working Group on Modelling and Scientific Computing 2007–2008, in: ESAIM Proc., vol. 25, EDP Sciences, Les Ulis, 2008, pp. 114–129.
[19] M.J. Gander, Analysis of the parareal algorithm applied to hyperbolic problems using characteristics, Bol. Soc. Esp. Mat. Apl. S$\vec{e}$MA (42) (2008) 21–35.
[20] Y. Maday, Parareal in time algorithm for kinetic systems based on model reduction, in: High-dimensional Partial Differential Equations in Science and Engineering, in: CRM Proc. Lecture Notes, vol. 41, Am. Math. Soc., Providence, RI, 2007, pp. 183–194.
[21] D. Guibert, D. Tromeur-Dervout, Adaptive parareal for systems of ODEs, in: Domain Decomposition Methods in Science and Engineering XVI, in: Lect. Notes Comput. Sci. Eng., vol. 55, Springer, Berlin, 2007, pp. 587–594.
[22] P.F. Fischer, F. Hecht, Y. Maday, A parareal in time semi-implicit approximation of the Navier–Stokes equations, in: Domain Decomposition Methods in Science and Engineering, in: Lect. Notes Comput. Sci. Eng., vol. 40, Springer, Berlin, 2005, pp. 433–440.
[23] M.L. Minion, A hybrid parareal spectral deferred corrections method, Commun. Appl. Math. Comput. Sci. 5 (2) (2010) 265–301.
[24] L. Baffico, S. Bernard, Y. Maday, G. Turinici, G. Zérah, Parallel-in-time molecular-dynamics simulations, Phys. Rev. E 66 (2002) 057701.
[25] R. Guettat, Couplage de l'algorithme parareel avec quelques methodes de décomposition de domaine, Ph.D. thesis, Université Pierre et Marie Curie, Paris, 2011.
[26] Y. Maday, G. Turinici, A parareal in time procedure for the control of partial differential equations, C. R. Math. Acad. Sci. Paris 335 (4) (2002) 387–392.
[27] Y. Maday, J. Salomon, G. Turinici, Monotonic parareal control for quantum systems, SIAM J. Numer. Anal. 45 (6) (2007) 2468–2482 (electronic).
[28] T.P. Mathew, M. Sarkis, C.E. Schaerer, Analysis of block parareal preconditioners for parabolic optimal control problems, SIAM J. Sci. Comput. 32 (3) (2010) 1180–1200.
[29] R. Dautray, J.-L. Lions, Analyse mathématique et calcul numérique pour les sciences et les techniques, Collection du Commissariat à l'Énergie Atomique: Série Scientifique (Collection of the Atomic Energy Commission: Science Series), Tome 1, Masson, Paris, ISBN 2225812950, 1984, with the collaboration of Michel Artola, Marc Authier, Philippe Bénilan, Michel Cessenat, Jean-Michel Combes, André Gervat, Hélène Lanchon, Bertrand Mercier, Claude Wild and Claude Zuily.
[30] P. Reuss, Précis de neutronique, EDP Sciences, Les Ulis, 2003.
[31] B. Akdeniz, Improved Neutron Kinetics for Coupled Three-Dimensional Boiling Water Reactor Analysis, The Pennsylvania University, 2007.
[32] E. Aubanel, Scheduling of tasks in the parareal algorithm, Parallel Comput. 37 (3).
[33] M.K. Riahi, Conception and analyze of time-parallel algorithm for evolutionary system, Ph.D. thesis, Université Pierre et Marie Curie, Paris, 2012.
[34] W.W.S. Langenbuch, W. Maurer, Coarse-mesh flux expansion method for the analysis of space–time effects in large light water reactor cores, Nucl. Sci. Eng. 63 (4) (1977) 437–456.
[35] H. Ikeda, T. Takeda, Development and verification of an efficient spatial neutron kinetics method for reactivity-initiated event analyses, J. Nucl. Sci. Technol. 38 (7) (2001) 492–502.
[36] O. Pironneau, F. Hecht, K. Ohtsuka, FreeFem++-mpi, third edition, 2012.
[37] A.-M. Baudron, J.-J. Lautard, Y. Maday, O. Mula, The parareal in time algorithm applied to the kinetic neutron diffusion equation, in: 21st International Conference on Domain Decomposition Methods, 2013.
[38] O. Mula, Some contributions towards the parallel simulation of time dependent neutron transport and the integration of observed data in real time, Ph.D. thesis, Université Pierre et Marie Curie, Paris, 2014.