ESAIM: M2AN 47 (2013) 717–742
DOI: 10.1051/m2an/2012046

ESAIM: Mathematical Modelling and Numerical Analysis
www.esaim-m2an.org

SYMMETRIC PARAREAL ALGORITHMS FOR HAMILTONIAN SYSTEMS

Xiaoying Dai[1,3], Claude Le Bris[2], Frédéric Legoll[2] and Yvon Maday[3,4]

Abstract. The parareal in time algorithm allows for efficient parallel numerical simulations of time-dependent problems. It is based on a decomposition of the time interval into subintervals, and on a predictor-corrector strategy, where the propagations over each subinterval for the corrector stage are concurrently performed on the different processors that are available. In this article, we are concerned with the long time integration of Hamiltonian systems. Geometric, structure-preserving integrators are preferably employed for such systems because they show interesting numerical properties, in particular excellent preservation of the total energy of the system. Using a symmetrization procedure and/or a (possibly also symmetric) projection step, we introduce here several variants of the original plain parareal in time algorithm [L. Baffico, et al. *Phys. Rev. E* **66** (2002) 057701; G. Bal and Y. Maday, A parareal time discretization for nonlinear PDE's with application to the pricing of an American put, in *Recent developments in domain decomposition methods*, *Lect. Notes Comput. Sci. Eng.* **23** (2002) 189–202; J.-L. Lions, Y. Maday and G. Turinici, *C. R. Acad. Sci. Paris, Série I* **332** (2001) 661–668.] that are better adapted to the Hamiltonian context. These variants are compatible with the geometric structure of the exact dynamics, and are easy to implement. Numerical tests on several model systems illustrate the remarkable properties of the proposed parareal integrators over long integration times. Some formal elements of understanding are also provided.

Mathematics Subject Classification. 65L05, 65P10, 65Y05.

Received November 24, 2010. Revised February 14, 2012.
Published online March 4, 2013.

1. Introduction

Increasingly intensive computations now become possible thanks to the improvement of both the efficiency and the clock rate of processors, the interprocessor's connections and the access to different levels of memory. In addition, parallel computing platforms, which allow many processors to work concurrently, also become

Keywords and phrases. Parallel integrators, Hamiltonian dynamics, long-time integration, symmetric algorithms, symmetric projection, geometric integration.

[1] LSEC, Institute of Computational Mathematics and Scientific/Engineering Computing, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, China. daixy@lsec.cc.ac.cn

[2] École Nationale des Ponts et Chaussées, 6 et 8 avenue Blaise Pascal, 77455 Marne-La-Vallée Cedex 2, France and INRIA Rocquencourt, MICMAC team-project, Domaine de Voluceau, B.P. 105, 78153 Le Chesnay Cedex, France.
lebris@cermics.enpc.fr; legoll@lami.enpc.fr

[3] UPMC Univ. Paris 06, UMR 7598, Laboratoire J.-L. Lions, Boîte courrier 187, 75252 Paris Cedex 05, France.
maday@ann.jussieu.fr

[4] Division of Applied Mathematics, Brown University, Providence, RI, USA

Article published by EDP Sciences

© EDP Sciences, SMAI 2013

https://doi.org/10.1051/m2an/2012046 Published online by Cambridge University Press

available. This second feature can only be useful if the problem to be solved can be decomposed into a series of independent tasks, each of them being assigned to one of the processors.

The design of efficient algorithms for parallel architectures is the subject of intense current research. In the case of models governed by partial differential equations, most of – if not all – the contributions of the last three decades perform domain decomposition. We refer to [35, 42] for a review on recent advances, and also to the proceedings of the Domain Decomposition Method meetings (see `www.ddm.org`) for various achievements. When the problem is time-dependent, or when the problem is solely governed by a system of ordinary differential equations, relatively few contributions are available. We refer *e.g.* to the book of Burrage [9] for a synthetic approach on the subject (see also [10]). In this book, the various techniques of parallel in time algorithms are classified into three categories: (i) parallelism *across the system*, (ii) parallelism *across the method* and (iii) parallelism *across time*.

The parareal in time method, our focus here, was first proposed in the work of Lions, Maday and Turinici in 2001 [30]. It belongs to the third category where parallelism is achieved by breaking up the integration interval into sub-intervals and concurrently solving over each sub-interval. The obvious difficulty is to provide the correct initial value for the integration over each of these sub-intervals. Most of the techniques in the third category are *multishooting techniques*, starting from the precursory work by Bellen and Zennaro [6]. This next led to the *waveform relaxation methods* introduced by Lelarasmee, Ruehli and Sangiovanni-Vincentelli [29], and to the *multigrid approaches* introduced by Hackbush [19]. See also the contribution [11] by Chartier and Philippe.

As it has been explained by Gander and Vandewalle in [17], the parareal in time algorithm can be interpreted both as a type of multishooting technique and also as a type of multigrid approach, even though the bottom line of the approach is closer to that of spatial domain decomposition with coarse grid preconditioners.

It is intuitively clear why so few contributions propose parallel in time algorithms: time-dependent problems are intrinsically sequential. On the other hand, the development of parallel computing provides computational opportunities that exceed the needs of parallelization in space. This motivates the development of efficient parallel in time approaches.

The parareal in time algorithm is based on the decomposition of the temporal domain into several temporal subdomains and the combination of a coarse and a fine solution procedure. The name "parareal" has been adopted to indicate that the algorithm has, in principle, the potential to so efficiently speed up the simulation process that real time approximation of the solution of the problem becomes plausible. Since the original work [30], where the convergence of the algorithm is proven, the parareal algorithm has received some attention and new developments have been proposed. In [4], Bal and Maday have provided a new interpretation of the scheme as a predictor-corrector algorithm (see also Baffico *et al.* [2]). The scheme involves a prediction step based on a coarse approximation for a global propagation of the system and a correction step computed in parallel and based on a fine approximation. Bal (in [3]) and Ronquist and Staff (in [41]) next provided some analysis on the convergence and the stability of the scheme. In [16], Gander and Vandewalle proved a superlinear convergence of the algorithm when used on bounded time intervals, and a linear convergence on unbounded time intervals.

In the past few years, the parareal algorithm has been successfully applied to various types of problems (see [31] for a review): nonlinear PDEs [4], control problems [32], quantum control problems [33], Navier Stokes equations [14], structural dynamics [13], reservoir simulation [18], ...

Although the plain parareal algorithm has proved to be efficient for many time-dependent problems, it also has some drawbacks in some specific cases. This is for instance the case for molecular dynamics simulations (as pointed out in [2]), or, more generally, for Hamiltonian problems. The exact flow of the system then enjoys many specific *geometrical* properties (symplecticity, possibly time-reversibility, ...). Some quantities are preserved along the trajectory: the Hamiltonian (*e.g.* the total energy of the system), and, in some cases, other quantities such as the angular momentum, ... See the textbooks [22, 27, 40] for a systematic introduction to numerical integration techniques for Hamiltonian systems. Symplectic and symmetric integrators are known to be suitable integrators for Hamiltonian systems. It turns out that, even in the case when the parareal algorithm is based on coarse and fine integrators that enjoy adequate geometrical properties (such as symplectic or symmetric

https://doi.org/10.1051/m2an/2012046 Published online by Cambridge University Press

integrators), the global parareal algorithm itself does not enjoy any of these properties. Consequently, the long time properties of the numerical flow (including *e.g.* energy preservation) are not as good as expected. In this article, our aim is to design a parareal scheme that preserves some geometrical structure of the exact dynamics.

Our article is organized as follows. Sections 2 and 3 collect some preliminary material. We first briefly recall the parareal algorithm, as presented in [2,4]. We next discuss numerical schemes for Hamiltonian problems, and show the deficiencies of the parareal algorithm on such problems. In Section 4, we develop a *symmetric* version of the parareal algorithm, in a sense made precise at the end of Section 4.1. This symmetric version unfortunately provides unsatisfactory results. Modifications are in order. As an alternative to symmetrization, we explore in Section 5 the idea of projecting the trajectory onto the constant energy manifold. Next, in Section 6, we couple a projection step with the symmetric algorithm developed in Section 4, while keeping the overall scheme symmetric. Combining these two ideas, symmetry and projection, yields the most efficient algorithms. The performances of all the schemes proposed in these sections are illustrated by numerical simulations on two low-dimensional systems, the harmonic oscillator and the two-dimensional Kepler problem. In Section 7, we consider a test case in a higher dimensional phase space, namely the simulation of the solar system (see also [38] for the derivation of algorithms specific to this test case, involving some parallel computations). We demonstrate that the efficiency and the qualitative properties observed in the previous test cases carry over to this more challenging example. Using the symmetric parareal scheme with symmetric projection developed in Section 6 (see Algorithm 1), we obtain a speed-up of more than 60 with respect to a fully sequential computation (provided a sufficient number of processors are available), for an equal accuracy. Section 8 summarizes our conclusions.

An extended version of the present article is available at [12].

2. The plain parareal algorithm

In this section, we review the plain parareal algorithm for a time-dependent problem in a general setting. The reader familiar with this scheme can skip this section and directly proceed to Section 3.

Consider u solution to the Cauchy problem

$$\frac{\partial u}{\partial t} + f(u) = 0,\ t \in [0,T], \quad \text{supplied with the initial condition } u(0) = u_0. \tag{1}$$

We assume standard appropriate conditions on f ensuring existence, uniqueness and continuity with respect to perturbations of the solution u to this problem (which is hence well-posed in the sense of Hadamard). Let $\mathcal{E}$ be the exact propagator, defined by $\mathcal{E}_\tau(u_0) = u(\tau)$, where $u(\tau)$ denotes the solution at time τ of problem (1).

In what follows, for reference, we approximate the exact propagator $\mathcal{E}$ by using an accurate numerical scheme. We can use any of the classical one-step schemes (explicit or implicit Euler schemes for simple problems, velocity Verlet scheme in the case of Hamiltonian dynamics) with a sufficiently small time step δt. The associated discrete propagator is denoted as $\mathcal{F}$, with no reference to the time step δt, in order not to make the notation too heavy. $\mathcal{F}_\tau(u_0)$ is thus an approximation of $\mathcal{E}_\tau(u_0)$. Assuming that the approximation u_n of $u(T_n)$ is known at some time T_n, the approximation of $u(T_{n+1})$ at a latter time T_{n+1} is computed by performing $(T_{n+1} - T_n)/\delta t$ steps of the fine scheme, that is denoted, with the previous notations,

$$u_{n+1} = \mathcal{F}_{T_{n+1}-T_n}(u_n).$$

In this article, we consider the dynamics

$$\dot{q} = M^{-1}p, \quad \dot{p} = -\nabla V(q), \quad u = (q,p) \in \mathbb{R}^d \times \mathbb{R}^d, \tag{2}$$

where M is a diagonal matrix, and V is a smooth scalar function depending on q. We will integrate (2) using the velocity Verlet scheme, which is explicit and of order 2, and reads

$$\begin{aligned} q_{n+1} &= q_n + M^{-1}\left(\delta t\ p_n - \frac{\delta t^2}{2}\nabla V(q_n)\right), \\ p_{n+1} &= p_n - \frac{\delta t}{2}\left(\nabla V(q_n) + \nabla V(q_{n+1})\right). \end{aligned} \tag{3}$$

https://doi.org/10.1051/m2an/2012046 Published online by Cambridge University Press

The plain parareal algorithm builds a sequence of N-tuples $\boldsymbol{u}^k \equiv \left\{u_n^k\right\}_{1\leq n\leq N}$ that converges, when $k\to\infty$, to the solution given by the fine scheme $\mathcal{F}$: $\lim_{k\to\infty} u_n^k = u_n$. In the sequel, we consider Hamiltonian problems, for which designing schemes with a non-uniform discretization is not straightforward. We thus restrict ourselves to the case of a regular discretization, namely $T_n = n\Delta T$ with $\Delta T = T/N$ for some $N\in\mathbb{N}^\star$, where $[0,T]$ is the time interval of interest. We introduce another approximation $\mathcal{G}$ of the exact flow $\mathcal{E}$, which is not as accurate as $\mathcal{F}$, but is much less expensive to use than $\mathcal{F}$. For example, we can choose the same discretization scheme as for $\mathcal{F}$, but with a larger time step $\mathrm{d}T \gg \delta t$. Hence, computing $\mathcal{G}_\tau$ amounts to performing $\tau/\mathrm{d}T$ time steps of length $\mathrm{d}T$. Here again, the dependency of $\mathcal{G}$ with respect to $\mathrm{d}T$ is not explicitly written. Another possibility is to define the solver $\mathcal{G}$ from a simpler problem, which does not contain as much information as the original problem, and thus is easier to solve. We use this opportunity in Section 7. In all the other sections, the only difference between $\mathcal{G}$ and $\mathcal{F}$ lies in the choice of the time step.

Assume that we know an approximation $\left\{u_n^k\right\}_{0\leq n\leq N}$ of the solution to (1), at the end of some iteration k. For $k=0$, this approximation is computed sequentially, using the coarse propagator $\mathcal{G}_{\Delta T}$. Then the parareal scheme defines the next iteration $\left\{u_n^{k+1}\right\}_{0\leq n\leq N}$ by

$$u_{n+1}^{k+1} = \mathcal{G}_{\Delta T}\left(u_n^{k+1}\right) + \mathcal{F}_{\Delta T}(u_n^k) - \mathcal{G}_{\Delta T}\left(u_n^k\right), \tag{4}$$

with the initial condition $u_0^{k+1} = u_0$. At the beginning of iteration $k+1$, we first compute $\mathcal{F}_{\Delta T}(u_n^k) - \mathcal{G}_{\Delta T}(u_n^k)$ in *parallel* over each interval $[n\Delta T, (n+1)\Delta T]$. Once this is completed, we only need to compute $\mathcal{G}_{\Delta T}(u_n^{k+1})$ and add to it the stored correction $\mathcal{F}_{\Delta T}(u_n^k) - \mathcal{G}_{\Delta T}(u_n^k)$. This is a sequential process, the complexity of which is negligible compared to the computation of $\mathcal{F}_{\Delta T}(u_n^k)$. Note that an improved implementation in parallel has recently been proposed in [8]. It will be explained in Section 3.3. The analysis of the complexities of the different parareal schemes proposed here will use this implementation.

From the construction of the scheme, convergence in at most N iterations can be proved [2, 4]. It can also be proven that, on a fixed time interval $[0,T]$ and under some regularity conditions, the scheme (4) yields a numerical solution u_n^k at iteration k that approximates $u(T_n)$ with an error which is bounded from above (up to a multiplicative constant independent of the time steps δt, $\mathrm{d}T$ and ΔT) by $\mathrm{err}\,\mathcal{F} + [\mathrm{err}\,\mathcal{G}]^{1+k}$, where $\mathrm{err}\,\mathcal{F}$ is the error on the solution at time T for the fine solver and $\mathrm{err}\,\mathcal{G}$ is the error on the solution at time T for the coarse solver (see [3, 30, 31]).

3. Parareal integration of Hamiltonian systems

Our purpose is to design parallel in time numerical schemes, derived from the parareal in time algorithm (4), for the integration of Hamiltonian dynamical systems. We first review the specificities of such dynamics before discussing their numerical integration with the parareal algorithm (4). The reader familiar with numerical integration of Hamiltonian systems can skip Section 3.1 and directly proceed to Section 3.2.

3.1. Numerical integration of Hamiltonian systems

In this article, we consider finite dimensional Hamiltonian systems, namely dynamical systems that read

$$\dot{q} = \frac{\partial H}{\partial p}, \quad \dot{p} = -\frac{\partial H}{\partial q}, \tag{5}$$

where the Hamiltonian $H(q,p)$ is a smooth scalar function depending on the variables $q = (q_1,\ldots,q_d)\in\mathbb{R}^d$ and $p = (p_1,\ldots,p_d)\in\mathbb{R}^d$.

The evolution of many physical systems can be written as a Hamiltonian dynamics. Examples include systems in molecular dynamics (where q and p respectively represent the positions and momenta of the atoms composing the system, see [15]), celestial mechanics (where q and p represent the positions and momenta of planets or satellites [25, 26, 39]), solid mechanics (after space discretization, the wave equation modelling elastodynamics

https://doi.org/10.1051/m2an/2012046 Published online by Cambridge University Press

yields a Hamiltonian dynamics of the type (5), see [23, 24]). In all these cases, $H(q,p)$ is, physically, the total energy of the system.

Hamiltonian dynamics have very specific properties that we now briefly review (see [22, 40] for more comprehensive expositions). First, the energy $H(q,p)$ is preserved along the trajectory of (5). Second, the flow of a Hamiltonian system is *symplectic*. It is well-known that symplectic schemes, such as the velocity Verlet scheme (3), are appropriate schemes for long time numerical integration of Hamiltonian dynamics. They indeed define a symplectic numerical flow, and thus preserve, at the discrete level, a fundamental geometrical property of the exact flow. In addition, the numerical flow given by a symplectic scheme applied on the dynamics (5) can be shown to be almost equal to the exact flow of a Hamiltonian dynamics, for a so-called *modified Hamiltonian*, which is close to the original Hamiltonian H of (5). This is one of the main results of the celebrated backward error analysis for Hamiltonian dynamics (see [7, 21, 36], and the comprehensive survey in [22], Chap. IX). As a consequence, preservation of the energy can be shown, in the sense that, for all n such that $n\delta t \leq C\exp(c/\delta t)$,

$$|H(q_n,p_n) - H(q_0,p_0)| \leq C(\delta t)^r,$$

where δt denotes the time step, r the order of the scheme, and the positive constants c and C are independent of δt. The numerical error on the energy hence does not grow with the simulation interval (at least for time intervals exponentially long in the inverse of the time step). In some cases (namely, the case of completely integrable systems and of almost completely integrable systems, such as the solar system), the difference between the numerical and the exact trajectories can be shown to grow linearly in time, rather than exponentially, as would be the case for a generic integrator used on a generic dynamics $\dot{u} = f(u)$.

In this article, we consider Hamiltonian systems for which the energy reads

$$H(q,p) = \frac{1}{2}p^T M^{-1} p + V(q), \tag{6}$$

where, as above, M is a diagonal matrix and V is a smooth scalar function depending on the positions q. For such an energy, the dynamics (5) is exactly (2). Setting $u = (q,p)$, we recast this system as $\dot{u} = f(u)$. We then observe that the system is reversible with respect to ρ defined by $\rho(q,p) = (q,-p)$, that is f satisfies $f \circ \rho = -\rho \circ f$. As a consequence, for any t, the exact flow $\Psi = \mathcal{E}_t$ of (2) is *ρ-reversible*, that is

$$\rho \circ \Psi = \Psi^{-1} \circ \rho. \tag{7}$$

Another class of interesting schemes for *reversible* Hamiltonian systems such as (2) is the family of ρ-reversible schemes: a one-step scheme defined by $u_{n+1} = \Psi_{\delta t}(u_n)$ is ρ-reversible if $\Psi_{\delta t}$ satisfies (7). Properties similar to the ones obtained with a symplectic scheme (good preservation of the energy, ...) can be shown when a reversible Hamiltonian dynamics is integrated using a ρ-reversible scheme. Conditions under which such properties are proven are however more restrictive than those of the backward error analysis for symplectic schemes: the Hamiltonian system should be *reversible integrable* (this implies that, if q and p are in $\mathbb{R}^d$, then the dynamics preserves at least d invariants) and a *non resonant* condition should be satisfied ([22], Chap. XI).

The construction of ρ-reversible schemes is often done by first designing *symmetric* schemes. A one-step scheme defined by $u_{n+1} = \Psi_{\delta t}(u_n)$ is *symmetric* (see [22], Def. V.1.4) if, for any δt,

$$\Psi_{\delta t} \circ \Psi_{-\delta t} = \mathrm{Id}, \tag{8}$$

where Id is the identity map. If a symmetric scheme satisfies the additional property

$$\rho \circ \Psi_{\delta t} = \Psi_{-\delta t} \circ \rho, \tag{9}$$

then the scheme is ρ-reversible in the sense of (7) (see [22], Thm. V.1.5). In practice, condition (9) is much less restrictive than the symmetry condition (8), and is satisfied by many schemes (including *e.g.* the forward Euler

https://doi.org/10.1051/m2an/2012046 Published online by Cambridge University Press

scheme). In what follows, we will design symmetric schemes, satisfying (8), and check *a posteriori* that they indeed satisfy (9). These schemes are hence ρ-reversible, namely satisfy (7).

Despite the restrictions on the type of Hamiltonian systems for which ρ-reversible schemes allow for good long-time properties, symmetric schemes represent a very interesting alternative to symplectic schemes, because they are often easier to design.

We conclude this section by recalling that the velocity Verlet scheme (3), which is symplectic, as pointed out above, is also symmetric and ρ-reversible.

3.2. Parareal integration of Hamiltonian dynamics

We observe that, even if the coarse and fine propagators employed in the definition of the parareal algorithm (4) are symplectic (respectively symmetric), then, for a given parareal iteration $k \geq 1$ and at a given time step n, with $n > k$, the application $u_0 = (q_0, p_0) \mapsto u_n^k = (q_n^k, p_n^k)$ is not symplectic (respectively symmetric).

This lack of structure has immediate practical consequences when employing the plain parareal algorithm (4) on Hamiltonian systems. Consider as a first example the one-dimensional harmonic oscillator, with Hamiltonian

$$H(q,p) = \frac{1}{2}p^2 + \frac{1}{2}q^2, \quad p \in \mathbb{R},\ q \in \mathbb{R}, \tag{10}$$

that we integrate up to time $T = 10^4$. Choose the initial condition $q(0) = 1.2$, $p(0) = 0.01$, set $\Delta T = 0.2$, and consider the velocity Verlet scheme (which, we have recalled, is both symplectic and symmetric) for the fine and the coarse propagators, with time steps $\delta t = 10^{-3}$ and $\mathrm{d}T = 0.1$, respectively.

As a confirmation of the lack of geometric structure, we observe the lack of energy preservation. On Figure 1, we plot the relative error on the energy preservation, defined, at time $n\Delta T$ and iteration k, as

$$\mathrm{err}_n^k = \frac{\left|H(q_n^k, p_n^k) - H(q_0, p_0)\right|}{|H(q_0, p_0)|}. \tag{11}$$

We note that, for short times (say up to $T = 10^3$ for the iteration $k = 5$), energy preservation is equally good for the parareal algorithm and for the sequential fine scheme. However, it deteriorates for larger times. Not unexpectedly, we do not observe the traditional behavior of geometric schemes (either symplectic or symmetric), namely a good preservation of energy, even when the numerical trajectory is very different from the exact trajectory.

On Figure 1, we also plot, for several iterations k, the error

$$\mathrm{err}_n^k = \left\|q_n^k - q_n^{\mathrm{ref}}\right\| + \left\|p_n^k - p_n^{\mathrm{ref}}\right\| \tag{12}$$

on the trajectory (q, p), with respect to a reference trajectory $(q^{\mathrm{ref}}, p^{\mathrm{ref}})$ computed with the velocity Verlet algorithm – used sequentially – with the time step $\delta t/10$, where δt is the time step of the fine propagator $\mathcal{F}$. For $k = 5$, on the time interval $[0, 10^3]$, we note that results are very good: the parareal trajectory is very close to the fine scheme trajectory, for a much smaller computational effort. Indeed, at $k = 5$, assuming that the complexity of the coarse propagations is negligible, each processor has only computed 5 fine scale trajectories of length ΔT, in contrast to the situation when a sequential algorithm is used, and the processor has to compute $T/\Delta T = 5000$ fine scale trajectories of length ΔT to reach the time $T = 10^3$. However, we also see that the error at iteration $k = 5$ increases for $t \geq 10^3$, and becomes unacceptable for times t of the order of 10^4. With more iterations (say $k = 15$), convergence of the trajectories up to the time $T = 10^4$ is obtained (see [12]).

Similar observations hold for the two-dimensional Kepler problem, where

$$H(q,p) = \frac{1}{2}p^T p - \frac{1}{\|q\|}, \quad p \in \mathbb{R}^2,\ q \in \mathbb{R}^2, \tag{13}$$

for which we have considered the initial condition $q(0) = (1 - e, 0)$ and $p(0) = \left(0, \sqrt{\dfrac{1+e}{1-e}}\right)$, with $e = 0.6$. We do not include them here for the sake of brevity.

https://doi.org/10.1051/m2an/2012046 Published online by Cambridge University Press

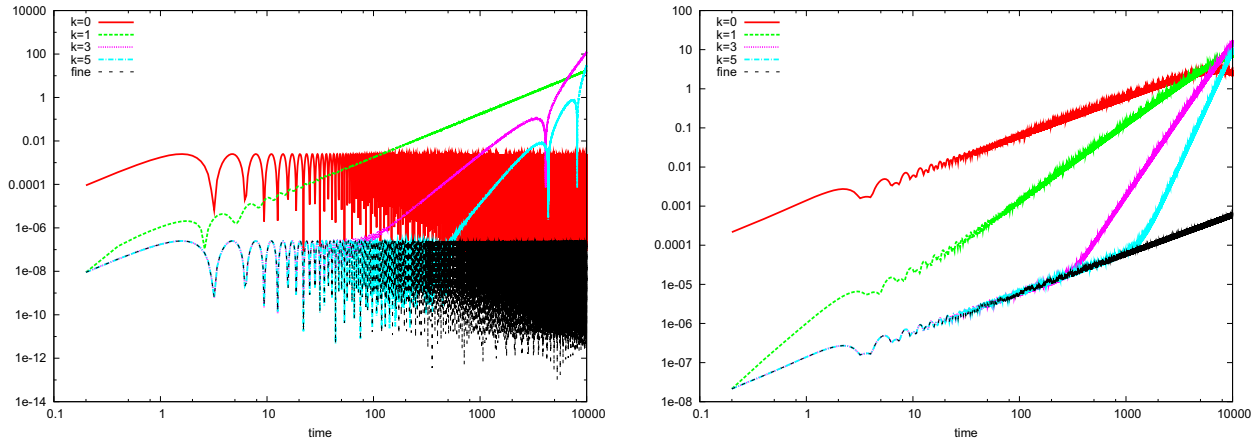


FIGURE 1. Error (11) on the energy (left) and errors (12) on the trajectory (right) for the harmonic oscillator problem, obtained by the parareal method (4) with $\delta t = 10^{-3}$, $\mathrm{d}T = 0.1$, $\Delta T = 0.2$.

3.3. Evaluation of the complexity of the plain parareal algorithm

In this section, for future reference, we assess the computational complexity of the parareal algorithm (4) for the integration of the Hamiltonian dynamics (2). We perform this evaluation under the assumptions that (i) the coarse and the fine propagators integrate the same dynamics, using an algorithm that requires the same fixed number of calls (set here to one) to the right hand side of (2) per time step (we consider that the complexities of evaluating ∇H or ∇V are equal, and we denote it by C_∇), and that (ii) we have all the necessary processors to perform all the fine scale computations in parallel.

We use here the implementation proposed in [8]. It consists in starting the computation of $\mathcal{F}_{\Delta T}\left(u_n^{k+1}\right)$ immediately after u_n^{k+1} has been computed in (4), and not waiting that all $(u_n^{k+1})_{1\le n\le N}$ are available. This allows us to start the parareal iteration $k+2$ much sooner. The complexity of the first coarse propagation scales as $C_\nabla\ T/\mathrm{d}T$. We next distinguish two extreme cases, whether the complexity of the fine propagator on a time interval of size ΔT is smaller (respectively larger) than the complexity of the coarse solver on the whole interval $[0,T]$. The first case corresponds to the situation when $\dfrac{\Delta T}{\delta t} < \dfrac{T}{\mathrm{d}T}$, the second case when $\dfrac{\Delta T}{\delta t} > \dfrac{T}{\mathrm{d}T}$.

In the first case, the complexity of each iteration is dominated by the complexity of the coarse solver on the whole interval $[0,T]$. The complexity of such a parareal iteration is thus again of the order of $C_\nabla\ T/\mathrm{d}T$. In the second case, the complexity of each parareal iteration is of the order of $C_\nabla\ \Delta T/\delta t$. In both cases, a final coarse iteration needs to be performed.

Denoting by K_{P} the number of parareal iterations, the approximate complexity of the scheme (4) is of the order of

$$C_{\mathrm{P}} = (K_{\mathrm{P}} + 1)\, C_\nabla\, \frac{T}{\mathrm{d}T}$$

in the first case, and

$$C_{\mathrm{P}} = K_{\mathrm{P}}\, C_\nabla\, \frac{\Delta T}{\delta t}$$

in the second one.

In comparison, the complexity of the fully sequential algorithm, using the small time step δt, is

$$C_{\mathrm{seq}} = C_\nabla\, \frac{T}{\delta t}.$$

https://doi.org/10.1051/m2an/2012046 Published online by Cambridge University Press

Note that the second case above corresponds more to the paradigm of the parareal scheme (remind also that, if possible, the coarse solver should be based on a less expensive dynamics than the fine solver, as in Section 7 below), especially for the integration of large Hamiltonian systems as the one considered at the end of the article. In that case, the speed-up is of the order $\dfrac{T}{K_{\mathrm{P}}\Delta T}$, which is the number of processors divided by the number of iterations.

For the complexity analysis, we assume that we are in the second case above, *i.e.* we assume that

$$\frac{\Delta T}{\delta t} > \frac{T}{\mathrm{d}T}. \tag{14}$$

4. A symmetric variant of the parareal in time algorithm

In [5], a *symplectic* variant of the parareal algorithm has been introduced. The strategy there is based on the reconstruction of the generating function $S : (q,p) \in \mathbb{R}^d \times \mathbb{R}^d \mapsto \mathbb{R}$ associated with the symplectic map $\mathcal{F}_{\Delta T} \circ \mathcal{G}_{\Delta T}^{-1}$. An interpolation procedure is used to obtain an approximation of that generating function. The optimal way to perform the interpolation is still an open question, especially when working with high dimensional problems ($d \gg 1$). In this article, we limit ourselves to the design of a *symmetric* scheme, using the fact that it is often simpler to symmetrize a given scheme than to make it symplectic. We hope to return to symplectic variants in future publications.

4.1. Derivation of the scheme

Our idea is based on the following well-known observation (see [22] Chap. V). Consider a general one-step scheme $U_{n+1} = \Psi_{\Delta T}(U_n)$. Then the scheme

$$U_{n+1} = \Psi_{\Delta T/2} \circ \left(\Psi_{-\Delta T/2}\right)^{-1} (U_n) \tag{15}$$

is symmetric. For future use, we introduce the intermediate variables $U_{n+1/2} = \left(\Psi_{-\Delta T/2}\right)^{-1} (U_n)$, and write the above scheme as

$$U_n = \Psi_{-\Delta T/2}\left(U_{n+1/2}\right), \quad U_{n+1} = \Psi_{\Delta T/2}\left(U_{n+1/2}\right). \tag{16}$$

We now write the parareal algorithm (4) in a form more appropriate for our specific purpose. Consider the first K iterations of the parareal algorithm, and denote

$$U_n := \left(u_n^0, u_n^1, \cdots, u_n^K\right).$$

Then the parareal scheme (4) can be written as

$$U_{n+1} = \Psi_{\Delta T}(U_n),$$

where the map $\Psi_{\Delta T}$ is defined by

$$\Psi_{\Delta T}(U_n) = \begin{pmatrix} \mathcal{G}_{\Delta T}(u_n^0) \\ \mathcal{G}_{\Delta T}(u_n^1) + \mathcal{F}_{\Delta T}\left(u_n^0\right) - \mathcal{G}_{\Delta T}\left(u_n^0\right) \\ \cdots \\ \mathcal{G}_{\Delta T}\left(u_n^K\right) + \mathcal{F}_{\Delta T}\left(u_n^{K-1}\right) - \mathcal{G}_{\Delta T}\left(u_n^{K-1}\right) \end{pmatrix}.$$

We now apply the symmetrization procedure (16) to the map $\Psi_{\Delta T}$, and consider the scheme

$$U_n = \Psi_{-\Delta T/2}\left(U_{n+1/2}\right), \quad U_{n+1} = \Psi_{\Delta T/2}\left(U_{n+1/2}\right). \tag{17}$$

https://doi.org/10.1051/m2an/2012046 Published online by Cambridge University Press

We next write this scheme in a more detailed manner. The first equation of (17) reads

$$\begin{aligned}
u_n^0 &= \mathcal{G}_{-\Delta T/2}\left(u_{n+1/2}^0\right),\\
u_n^1 &= \mathcal{G}_{-\Delta T/2}\left(u_{n+1/2}^1\right) + \mathcal{F}_{-\Delta T/2}\left(u_{n+1/2}^0\right) - \mathcal{G}_{-\Delta T/2}\left(u_{n+1/2}^0\right),\\
&\ldots = \ldots\\
u_n^{K-1} &= \mathcal{G}_{-\Delta T/2}\left(u_{n+1/2}^{K-1}\right) + \mathcal{F}_{-\Delta T/2}\left(u_{n+1/2}^{K-2}\right) - \mathcal{G}_{-\Delta T/2}\left(u_{n+1/2}^{K-2}\right),\\
u_n^K &= \mathcal{G}_{-\Delta T/2}\left(u_{n+1/2}^K\right) + \mathcal{F}_{-\Delta T/2}\left(u_{n+1/2}^{K-1}\right) - \mathcal{G}_{-\Delta T/2}\left(u_{n+1/2}^{K-1}\right).
\end{aligned}$$

We hence obtain

$$\begin{aligned}
u_{n+1/2}^0 &= \mathcal{G}_{-\Delta T/2}^{-1}\left(u_n^0\right),\\
u_{n+1/2}^1 &= \mathcal{G}_{-\Delta T/2}^{-1}\left[u_n^1 - \mathcal{F}_{-\Delta T/2}\left(u_{n+1/2}^0\right) + \mathcal{G}_{-\Delta T/2}\left(u_{n+1/2}^0\right)\right],\\
&\ldots = \ldots\\
u_{n+1/2}^{K-1} &= \mathcal{G}_{-\Delta T/2}^{-1}\left[u_n^{K-1} - \mathcal{F}_{-\Delta T/2}\left(u_{n+1/2}^{K-2}\right) + \mathcal{G}_{-\Delta T/2}\left(u_{n+1/2}^{K-2}\right)\right],\\
u_{n+1/2}^K &= \mathcal{G}_{-\Delta T/2}^{-1}\left[u_n^K - \mathcal{F}_{-\Delta T/2}\left(u_{n+1/2}^{K-1}\right) + \mathcal{G}_{-\Delta T/2}\left(u_{n+1/2}^{K-1}\right)\right].
\end{aligned}$$

We now collect the above set of relations with the second equation of (17) and obtain the following formulation:

$$u_{n+1/2}^0 = \mathcal{G}_{-\Delta T/2}^{-1}\left(u_n^0\right), \quad u_{n+1}^0 = \mathcal{G}_{\Delta T/2}\left(u_{n+1/2}^0\right), \tag{18}$$

at iteration $k = 0$, and next, for any $k \geq 0$,

$$\left\{\begin{aligned}
u_{n+1/2}^{k+1} &= \mathcal{G}_{-\Delta T/2}^{-1}\left[u_n^{k+1} - \mathcal{F}_{-\Delta T/2}\left(u_{n+1/2}^k\right) + \mathcal{G}_{-\Delta T/2}\left(u_{n+1/2}^k\right)\right],\\
u_{n+1}^{k+1} &= \mathcal{G}_{\Delta T/2}\left(u_{n+1/2}^{k+1}\right) + \mathcal{F}_{\Delta T/2}\left(u_{n+1/2}^k\right) - \mathcal{G}_{\Delta T/2}\left(u_{n+1/2}^k\right).
\end{aligned}\right. \tag{19}$$

We call the scheme (18)–(19) the *symmetric parareal scheme*.

Remark 1. We have recalled in Section 3.1 that ρ-reversible schemes have good geometrical properties. By construction, the scheme (18)–(19) is symmetric. We have checked that it also satisfies condition (9). As a consequence (see again Sect. 3.1), the scheme (18)–(19) is ρ-reversible.

We note that, apart from the computation of $\mathcal{G}_{-\Delta T/2}^{-1}$, the scheme (18)–(19) is completely explicit, as soon as the propagators $\mathcal{G}$ and $\mathcal{F}$ themselves are explicit. In particular, we point out that the inverse of the fine propagator $\mathcal{F}$ is not needed.

We now show that, in this new algorithm, all the expensive computations (involving $\mathcal{F}$) can actually be performed in parallel. Assume that, at a given iteration k, we know $\left\{u_n^k\right\}_{0\leq n\leq N}$ and $\left\{u_{n+1/2}^k\right\}_{0\leq n\leq N-1}$. Then the correction terms

$$\mathcal{F}_{\Delta T/2}\left(u_{n+1/2}^k\right) - \mathcal{G}_{\Delta T/2}\left(u_{n+1/2}^k\right)$$

and

$$\mathcal{F}_{-\Delta T/2}\left(u_{n+1/2}^k\right) - \mathcal{G}_{-\Delta T/2}\left(u_{n+1/2}^k\right)$$

can be computed independently, over each processor. The algorithm can next proceed with sequential, but inexpensive, computations. Note again that, as pointed out in Section 3.3, we can start the computation of

https://doi.org/10.1051/m2an/2012046 Published online by Cambridge University Press

$\mathcal{F}_{\Delta T/2}(u^k_{n+1/2})$ and $\mathcal{F}_{-\Delta T/2}(u^k_{n+1/2})$ as soon as $u^k_{n+1/2}$ is available (we do not have to wait for all the computations of iteration k to be completed). The complexity of this algorithm is hence the same as that of the plain parareal algorithm (4).

Some comments are in order. The parareal scheme (4) is not a one-step scheme defining u^{k+1}_{n+1} from u^{k+1}_n, and hence the symmetric form (18)–(19) cannot be considered, strictly speaking, as a symmetric integrator in the classical sense. However, several reasons lead us to believe that this algorithm is the appropriate generalization of symmetric integrators when dealing with parareal-type algorithms.

First, the scheme at iteration $k = 0$, defined by (18), is exactly the symmetrization (15) of the coarse propagator $\mathcal{G}_{\Delta T}$.

Second, the flow (18)–(19) is symmetric in the sense that, if for some n and some k,

$$\left(u^0_{n+1}, \cdots, u^k_{n+1}, u^{k+1}_{n+1}\right)$$

is obtained from

$$\left(u^0_n, \cdots, u^k_n, u^{k+1}_n\right)$$

by the flow implicitly defined in (18)–(19), then

$$\left(u^0_n, \cdots, u^k_n, u^{k+1}_n\right)$$

is obtained from

$$\left(u^0_{n+1}, \cdots, u^k_{n+1}, u^{k+1}_{n+1}\right)$$

by the exact same algorithm reversing the time. In fact, only the consideration (and the storage) of the last two iterates (in terms of "parareal iterates") (u^k_n, u^{k+1}_n) is required to perform the iterations, and the above argument.

Third, if the coarse propagator happens to be identical to the fine one (which is of course not supposed to be the case for efficiency of the parareal integrator!), then the symmetrized form (19) reads

$$u^{k+1}_{n+1/2} = \mathcal{G}^{-1}_{-\Delta T/2}\left(u^{k+1}_n\right), \quad u^{k+1}_{n+1} = \mathcal{G}_{\Delta T/2}\left(u^{k+1}_{n+1/2}\right),$$

and it thus coincides with the standard symmetrized version of the coarse propagator.

Finally, formally taking the limit $k \to +\infty$ in (19) yields

$$u^\infty_{n+1} = \mathcal{F}_{\Delta T/2} \circ (\mathcal{F}_{-\Delta T/2})^{-1}(u^\infty_n).$$

This shows that the limit of the symmetric parareal algorithm in terms of parareal iterations coincides with a standard symmetrized form of the map $\mathcal{F}_{\Delta T}$. Note also that this latter algorithm is *not* the symmetrized version of the fine propagator, since symmetrization does not occur after each time step, but after $\Delta T/(2\delta t)$ time steps.

These observations show the formal consistency of our notion of symmetrization for parareal-type integrators with the classical notion of symmetry.

Remark 2. Instead of considering (15) to symmetrize a given scheme $\Psi_{\Delta T}$, one can alternatively consider the symmetric scheme

$$U_{n+1} = \left(\Psi_{-\Delta T/2}\right)^{-1} \circ \Psi_{\Delta T/2}(U_n).$$

In the parareal context, this yields an algorithm with similar properties as the above algorithm (18)–(19), and with comparable numerical results. In the sequel, for the sake of brevity, we only consider the symmetrization procedure (15).

https://doi.org/10.1051/m2an/2012046 Published online by Cambridge University Press

4.2. Numerical examples

We again consider the example of the harmonic oscillator (10), and integrate this Hamiltonian system with our newly constructed symmetric parareal algorithm (18)–(19). Note that system (10) is an integrable reversible Hamiltonian system, and thus belongs to a class of problems for which symmetric integrators have been shown to preserve energy ([22], Chap. XI), as has been recalled in Section 3.1.

The results obtained with this symmetrized version of the algorithm are disappointing: they are indeed very similar, both qualitatively and quantitatively, to those obtained with the plain parareal scheme (4) (see Sect. 3.2, Fig. 1). Similar conclusions are drawn for the Kepler problem.

It is useful and instructive to now explain such a poor behavior, using the following formal elements of analysis. We first observe that a parareal integrator may be seen, at parareal iteration k, as an integrator of a system consisting of $k+1$ identical replicas of the original system under consideration. From (18), we know that the first replica is integrated by a symmetric algorithm. If the system under study is an integrable reversible Hamiltonian system, its energy is thus preserved in the long time by the simulation at parareal iteration $k = 0$ (this is confirmed by numerical experiments). Next, since the replicas are noninteracting, the system of replicas is evidently an integrable reversible system, and has an energy that is equal to $k + 1$ times the energy of the original system. Assume now that the symmetric propagator (18)–(19) applied to the system of $k + 1$ identical replicas conserves the energy of the global system (*i.e. the sum* $\sum_{\ell=0}^{k} H(q^\ell, p^\ell)$ of the energies of the replicated systems), as we could expect from a symmetric scheme applied to an integrable reversible Hamiltonian system. Under this assumption, it follows, by induction on k, that the energy is preserved along the trajectory, at *each* parareal iteration k. This is clearly in contradiction with the observed numerical results! The flaw in the above argument is that, for a symmetric scheme to preserve the energy of an integrable reversible system, we have recalled that, among other conditions, the frequencies present in the system have to satisfy a non-resonant diophantine condition ([22], Condition X.2.4 and Thm. XI.3.1). This is precisely not the case here for the system of replicas, by replication of the original frequencies!

The theoretical argument showing energy preservation does not apply, and numerical results confirm that energy is indeed not well-preserved, in the long-time limit. The symmetric parareal scheme (18)–(19) hence needs to be somehow amended.

4.3. Symmetric parareal algorithm with frequency perturbation

To prevent the different replicas from being resonant, a possibility is to consider, at each parareal iteration k, a system slightly different from the original system. For instance, in the case of the harmonic oscillator, we may consider at each iteration k a harmonic oscillator with a specific frequency ω_k:

$$H_k(q,p) = \frac{1}{2}p^2 + \frac{1}{2}\omega_k^2\, q^2.$$

The unperturbed case corresponds to $\omega_k = \omega_{\text{exact}} = 1$ in the above energy. Provided the ω_k are all different from one another (and non-resonant), the system of replicas is non-resonant. The shift is chosen such that it vanishes when $k \to \infty$, *i.e.* $\lim_{k\to\infty} \omega_k = \omega_{\text{exact}}$. Likewise, for the Kepler problem, we introduce a perturbation by considering

$$H_k(q,p) = \frac{1}{2}p^T p - \frac{\alpha_k}{\|q\|}, \tag{20}$$

where the unperturbed case corresponds to $\alpha_k = \alpha_{\text{exact}} = 1$.

More precisely, we introduce a fine propagator $\mathcal{F}_{\Delta T}^{(k)}$ and a coarse propagator $\mathcal{G}_{\Delta T}^{(k)}$ for the Hamiltonian dynamics associated to the Hamiltonian H_k. Rather than symmetrizing the parareal algorithm (4), we consider the scheme

$$\begin{aligned} u_{n+1}^0 &= \mathcal{G}_{\Delta T}^{(0)}\left(u_n^0\right), \\ u_{n+1}^{k+1} &= \mathcal{G}_{\Delta T}^{(k+1)}\left(u_n^{k+1}\right) + \mathcal{F}_{\Delta T}^{(k+1)}\left(u_n^k\right) - \mathcal{G}_{\Delta T}^{(k+1)}\left(u_n^k\right), \end{aligned} \tag{21}$$

https://doi.org/10.1051/m2an/2012046 Published online by Cambridge University Press

(note the upper indices $(k+1)$) and symmetrize this scheme along the lines of Section 4.1. We thus obtain, at iteration $k=0$, the algorithm

$$u^0_{n+1/2} = \left(\mathcal{G}^{(0)}_{-\Delta T/2}\right)^{-1}(u^0_n), \quad u^0_{n+1} = \mathcal{G}^{(0)}_{\Delta T/2}\left(u^0_{n+1/2}\right), \tag{22}$$

and, for $k \geq 0$,

$$\begin{cases} u^{k+1}_{n+1/2} = \left(\mathcal{G}^{(k+1)}_{-\Delta T/2}\right)^{-1}\left[u^{k+1}_n - \mathcal{F}^{(k+1)}_{-\Delta T/2}\left(u^k_{n+1/2}\right) + \mathcal{G}^{(k+1)}_{-\Delta T/2}\left(u^k_{n+1/2}\right)\right] \\ u^{k+1}_{n+1} = \mathcal{G}^{(k+1)}_{\Delta T/2}\left(u^{k+1}_{n+1/2}\right) + \mathcal{F}^{(k+1)}_{\Delta T/2}\left(u^k_{n+1/2}\right) - \mathcal{G}^{(k+1)}_{\Delta T/2}\left(u^k_{n+1/2}\right). \end{cases} \tag{23}$$

We briefly discuss the consistency of this scheme. First, at iteration $k+1$, if the coarse propagator $\mathcal{G}^{(k+1)}$ is identical to the fine propagator $\mathcal{F}^{(k+1)}$, then (23) reads

$$u^{k+1}_{n+1/2} = \left(\mathcal{F}^{(k+1)}_{-\Delta T/2}\right)^{-1}\left(u^{k+1}_n\right), \quad u^{k+1}_{n+1} = \mathcal{F}^{(k+1)}_{\Delta T/2}\left(u^{k+1}_{n+1/2}\right),$$

which is a scheme consistent with the Hamiltonian dynamics driven by H_{k+1}. Since the perturbation can be neglected when $k \to \infty$, we recover, in the limit of large k, a scheme consistent with the original dynamics.

In addition, formally taking the limit $k \to +\infty$ in (23) yields

$$u^\infty_{n+1} = \mathcal{F}_{\Delta T/2} \circ (\mathcal{F}_{-\Delta T/2})^{-1}(u^\infty_n),$$

where $\mathcal{F}$ is the fine propagator associated to the original dynamics. This shows that the limit of the algorithm (22)–(23) in terms of parareal iterations coincides with a standard symmetrized form of the fine propagator of the original dynamics.

We have first applied this algorithm to the harmonic problem, and we refer the reader to [12] for the corresponding results. On Figure 2, we plot the errors (11) on the energy and the errors (12) on the trajectory, obtained with the algorithm (22)–(23) applied to the Kepler problem. We observe that the energy does not drift, while the trajectory is not extremely accurate. We are hence in the regime of geometric integration, with a good accuracy on energy being *not* a consequence of a good accuracy on the trajectory. This is an improvement in comparison to the previous algorithms (parareal algorithm (4) and symmetric parareal algorithm (18)–(19)). Similar conclusions hold for the harmonic oscillator.

We hence numerically observe, in both cases considered, that the energy does not drift when we use the scheme (22)–(23), and that energy preservation does not owe to trajectory accuracy. This formally validates our understanding: the energy drifts observed with the symmetric parareal algorithm (18)–(19) are due to resonances. As soon as the replicas are made non-resonant, we recover a behavior in terms of energy preservation that is typical of geometric algorithms.

Nevertheless, even though energy does not drift, we also see that it is not extremely well preserved. It is often the case that, at the final iteration, energy preservation is actually not really better than if obtained using the coarse propagator on the original dynamics. We also observe that the error on the trajectory remains quite large, for the practical values of k that we consider. So the algorithm (22)–(23), although better than (4) and (18)–(19), is not satisfactory.

Remark 3. Another possibility to integrate at each parareal iteration systems that are non resonant from one another is to use, at each iteration k, a different time step $\mathrm{d}T_k$ for the coarse propagator. This strategy, that is more general than the one discussed above (indeed, "shifting the frequency" may not be easy if the system is not explicitly an harmonic oscillator), yields results that are qualitatively similar to the ones reported here.

Remark 4. The idea of "shifting the frequency" is independent from the fact that the algorithm is symmetric. It is hence possible to apply the *standard* parareal method (4) on systems that are slightly different from one another at each parareal iteration (in (23), we have used the *symmetric* version of the parareal algorithm on such shifted systems). Results are slightly better when this strategy is used with the symmetric version of the algorithm.

https://doi.org/10.1051/m2an/2012046 Published online by Cambridge University Press

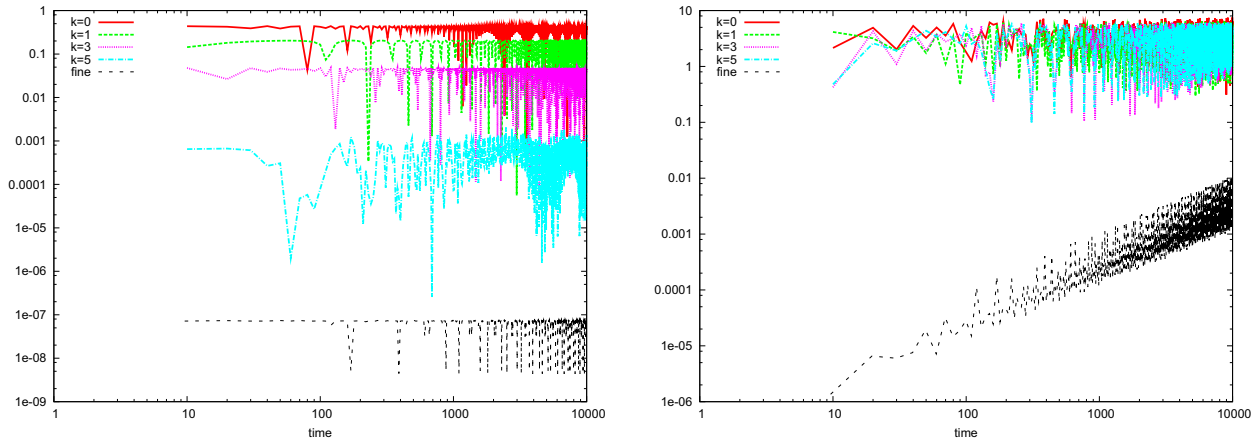


FIGURE 2. Error on the energy (left) and errors on the trajectory (right) for Kepler problem, obtained with the symmetric parareal method (22)–(23), with frequency perturbation. The coefficient α_k in (20) is $\alpha_{k=0} = 0.9$, $\alpha_{k=1} = 0.95$, $\alpha_{k=2} = 0.975$, $\alpha_{k=3} = 0.9875$, $\alpha_{k=4} = 0.99375$ and $\alpha_{k=5} = \alpha_{\text{exact}} = 1$ ($\delta t = 10^{-4}$, $\mathrm{d}T = 10^{-2}$, $\Delta T = 0.2$).

5. PARAREAL ALGORITHM WITH PROJECTION

As an alternative to symmetrization described in Section 4, we consider in this section a different idea. We couple the plain parareal method with a projection of the trajectory on a specific manifold, defined by the preservation of some invariants of the system (namely the energy, and possibly other ones).

Observe that many systems have actually more than one preserved quantity: this is the case for the two-dimensional Kepler problem, where the angular momentum $L = q \times p$ and the Runge–Lenz–Pauli vector $A = p \times L - \dfrac{q}{\|q\|}$ are also preserved, or for any completely integrable system. However, the generic situation is that some invariant quantities are known (among which the energy), and that there may be other quantities that are preserved by the dynamics, but may not have been identified. As a consequence, we first consider a generic method, based on projecting on the constant energy manifold

$$\mathcal{M} = \left\{(q,p) \in \mathbb{R}^{2d};\ H(q,p) = H_0\right\},$$

for the Kepler problem. Note that the harmonic oscillator is not a discriminating test case in this respect, since the energy is then the *unique* invariant. Next, for the sake of completeness, again for the Kepler problem, we consider a method based on projecting on the manifold where *both* H and L are kept constant (see Sect. 5.3).

Remark 5. Symplectic algorithms, such as SHAKE [37] or RATTLE [1, 28], have been proposed for the numerical integration of Hamiltonian systems restricted to some manifold defined by holonomic constraints. Note however that here, the constraint at hand, $H(q,p) = H_0$, is *not* holonomic.

Following ([22], Chap. IV), we consider the following definition of the projection $\pi_{\mathcal{M}}$ onto the manifold $\mathcal{M}$. For any $\widetilde{y} = (\widetilde{q}, \widetilde{p}) \in \mathbb{R}^{2d}$, we define $(q,p) = y = \pi_{\mathcal{M}}(\widetilde{y}) \in \mathbb{R}^{2d}$ by

$$y = \pi_{\mathcal{M}}(\widetilde{y}) = \widetilde{y} + \lambda \nabla H(\widetilde{y}), \quad \text{where } \lambda \in \mathbb{R} \text{ is such that } \quad H(y) = H_0. \tag{24}$$

5.1. Parareal algorithm with projection

The coarse propagator being the velocity Verlet algorithm, which yields an acceptable energy preservation, we do not need to project on the constant energy manifold at iteration $k = 0$. But, as shown by our numerical

https://doi.org/10.1051/m2an/2012046 Published online by Cambridge University Press

TABLE 1. To stop the projection procedure, we use three criteria. We gather here how often each criterion triggered the projection procedure to stop in Algorithm (25)–(26).

Criterion C1	Criterion C2	Criterion C3
91.6 %	6.8 %	1.6 %

results of the previous sections, the trajectory departs from this manifold for the iterations $k \geq 1$, in the long time limit. This motivates the consideration of the following algorithm.

At iteration $k = 0$, we set

$$u_{n+1}^0 = \mathcal{G}_{\Delta T}(u_n^0). \tag{25}$$

For the next iterations, we simply couple the parareal algorithm (4) with a projection step. We hence define the solution at iteration $k+1$ from the solution at iteration k by

$$u_{n+1}^{k+1} = \pi_{\mathcal{M}} \left[\mathcal{G}_{\Delta T}\left(u_n^{k+1}\right) + \mathcal{F}_{\Delta T}\left(u_n^k\right) - \mathcal{G}_{\Delta T}\left(u_n^k\right) \right]. \tag{26}$$

The additional complexity of the above algorithm, in comparison to the parareal algorithm (4), comes from solving the nonlinear projection step (24). Note however that we expect to have a good initial guess for this problem, since the solution at the previous parareal iteration is expected to be a coarse approximation of the exact solution. We hence expect to solve the nonlinear projection step with a few iterations of, say, a Newton algorithm. We will see below that this is indeed the case.

5.2. Numerical results

We consider the Kepler problem associated to the energy (13), and we plot the relative errors (11) on the energy preservation and the errors (12) on the trajectory on Figures 3 and 4 respectively.

The projection step (24) is solved using a Newton algorithm. Note yet that the energy is not exactly preserved, because only a few steps of the Newton algorithm are performed. Indeed, the algorithm is terminated as soon as one of the following three convergence criteria is satisfied:

C1: the quantity `err` is smaller than the tolerance `tol`,
C2: the maximum number $N_{\text{iter}}^{\max}$ of iterations has been reached,
C3: the error on the energy does not decrease,

where `err` is here the relative error on the energy (11). We have chosen to work with $N_{\text{iter}}^{\max} = 2$. It turns out that the best choice for `tol` is the relative error on the energy preservation of the fine scheme. In the current case, with $\delta t = 10^{-4}$, this corresponds to taking $\texttt{tol} = 10^{-7}$.

We observe an excellent energy preservation, although we possibly stop the Newton algorithm before convergence: the prescribed tolerance is reached over the complete time range for all parareal iterations $k \geq 7$. This is confirmed in Table 1, where we show that, in the Newton algorithm, the criterion C1 (the requested tolerance has been reached) is most often satisfied before the criteria C2 and C3.

We also observe that only 11 iterations are needed to obtain convergence of the trajectory (that is, the trajectory obtained at iteration $k = 11$ is as accurate as the one given by the fine scheme) on the time range $[0, 10^4]$. With this algorithm, we are hence able to recover accurately the trajectory on the complete time range $[0, 10^4]$, for a moderate number of iterations. Results are much better than with the other modifications of the parareal algorithm that we have described so far (for the same value of the parameters δt, $\mathrm{d}T$ and ΔT).

On Figure 5, we plot the relative errors on the angular momentum $L(q,p) = q \times p$, which is another invariant of the Kepler problem. The relative error is defined by

$$\mathrm{err}_n^k = \frac{\left| L(q_n^k, p_n^k) - L(q_0, p_0) \right|}{|L(q_0, p_0)|}. \tag{27}$$

https://doi.org/10.1051/m2an/2012046 Published online by Cambridge University Press

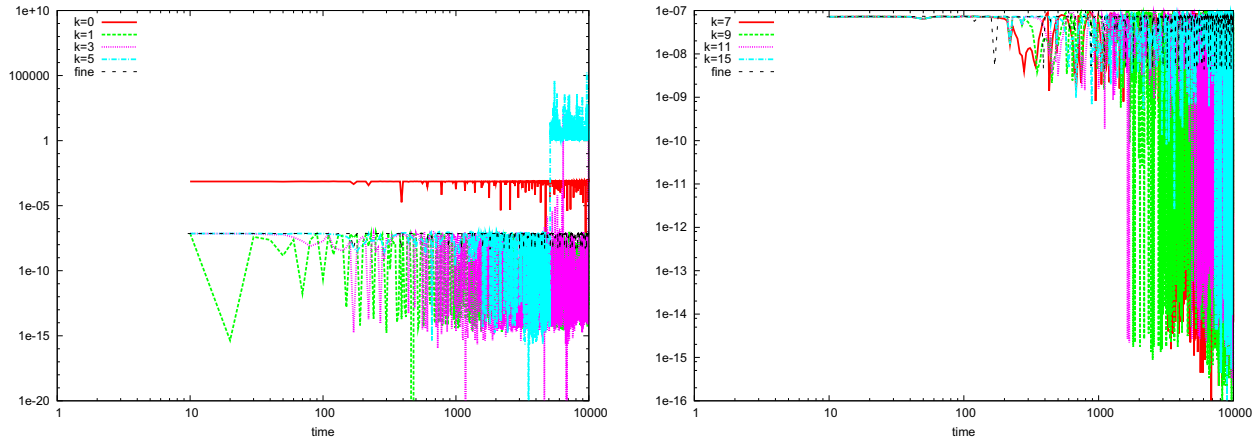


FIGURE 3. Errors on the energy for Kepler problem, obtained by algorithms (25)–(26) ($\delta t = 10^{-4}$, $\mathrm{d}T = 0.01$, $\Delta T = 0.2$).

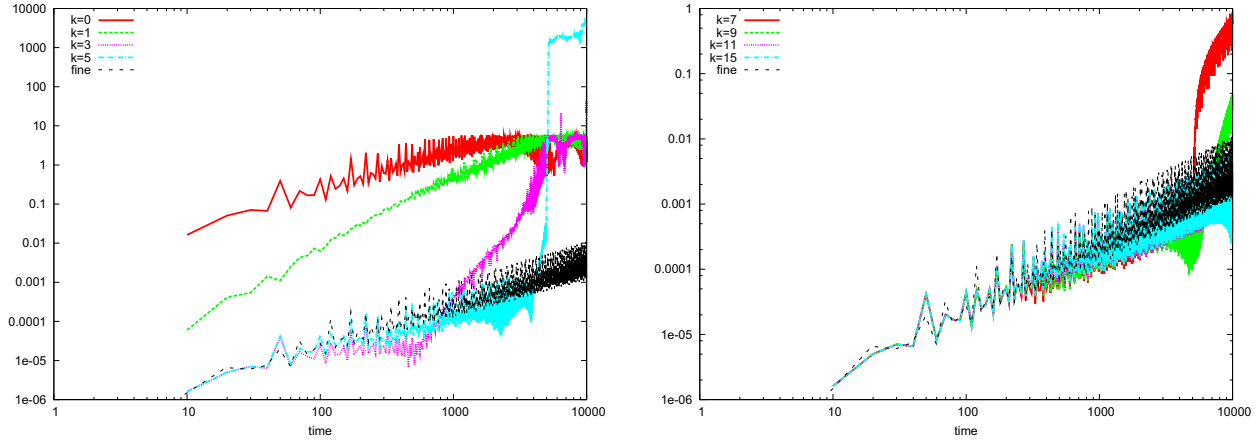


FIGURE 4. Errors on the trajectory for Kepler problem, obtained by algorithms (25)–(26) ($\delta t = 10^{-4}$, $\mathrm{d}T = 0.01$, $\Delta T = 0.2$).

This quantity does not blow-up. For $k \geq 7$, this error is always smaller than 10^{-2}, and it decreases down to 10^{-4} for the iterations $k \geq 11$. We yet observe that it is not preserved as well as would be the case with a geometric integrator. The rather good behavior of this invariant is a consequence of the accuracy of the trajectory.

Remark 6. Note that, in the above numerical simulations, we have considered a large final time T, too large for the assumption (14) to be satisfied. Of course, if one only looks at the left part of the above curves, corresponding to the interval $[0, \overline{T}]$ for a not too large $\overline{T}$, then (14) is satisfied and we already observe some interesting properties.

Note also that our aim with the two toy-problems considered so far (harmonic oscillator and Kepler system) is more to understand the basic features of the variants of the parareal algorithm than to be as efficient as possible. Our viewpoint will be different when we consider a system of higher dimensionality, in Section 7 below. We will then work with parameters such that (14) *is* satisfied, corresponding to the interesting regime when the complexity at each parareal iteration is dominated by the complexity of the fine solver.

https://doi.org/10.1051/m2an/2012046 Published online by Cambridge University Press

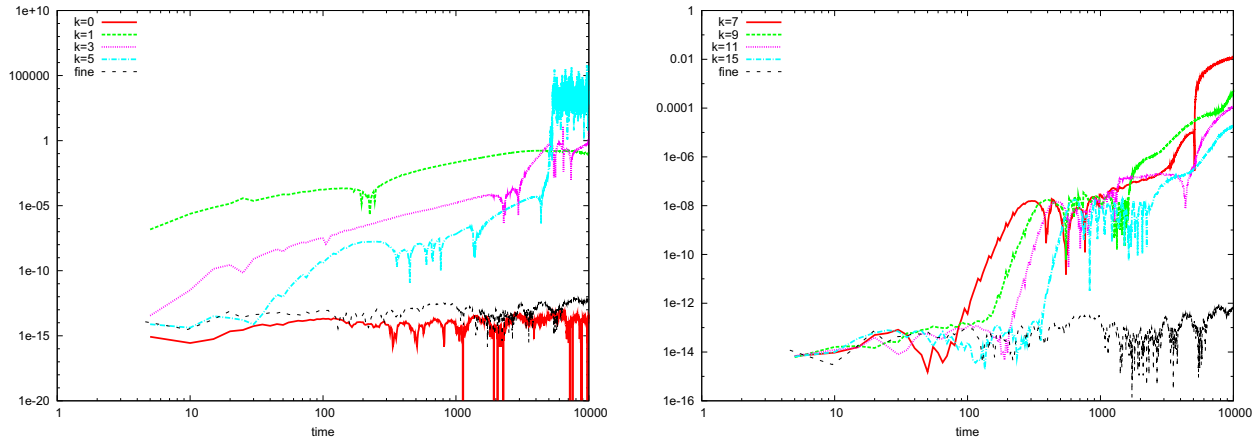


FIGURE 5. Errors on the angular momentum for Kepler problem, obtained by algorithms (25)–(26) ($\delta t = 10^{-4}$, $\mathrm{d}T = 0.01$, $\Delta T = 0.2$).

5.3. Considering more than one invariant

In this section, we again consider the case of the Kepler problem, and now take into account that, for this specific system, we know another invariant besides the energy, namely the angular momentum $L = q \times p$. Define the manifold

$$\mathcal{M}_2 = \left\{(q,p) \in \mathbb{R}^{2d};\ H(q,p) = H_0,\ L(q,p) = L_0\right\},$$

and consider the projection $\pi_{\mathcal{M}_2}$ onto that manifold, defined by

$$y = \pi_{\mathcal{M}_2}(\widetilde{y}) = \widetilde{y} + \lambda_1 \nabla H(\widetilde{y}) + \lambda_2 \nabla L(\widetilde{y}),$$

where $(\lambda_1, \lambda_2) \in \mathbb{R}^2$ is such that $H(y) = H_0$ and $L(y) = L_0$.

We couple this projection step with the parareal algorithm as follows. We first define u_n^0 at iteration $k = 0$ by (25):

$$u_{n+1}^0 = \mathcal{G}_{\Delta T}\left(u_n^0\right).$$

We next define the solution at iteration $k+1$ from the solution at iteration k by

$$u_{n+1}^{k+1} = \pi_{\mathcal{M}_2}\left[\mathcal{G}_{\Delta T}\left(u_n^{k+1}\right) + \mathcal{F}_{\Delta T}\left(u_n^k\right) - \mathcal{G}_{\Delta T}\left(u_n^k\right)\right]. \tag{28}$$

The algorithm is similar to algorithms (25)–(26) where $\pi_{\mathcal{M}}$ is replaced by $\pi_{\mathcal{M}_2}$.

We integrate the Kepler dynamics with this algorithm (we again use a Newton algorithm to compute the Lagrange multipliers, with the same three stopping criteria as in the previous section). We observe qualitatively the same behavior as for algorithms (25)–(26), which is based on projection on the constant energy manifold. Only 8 iterations are needed to obtain convergence of the trajectory, rather than 11 before (see Fig. 6, where we plot the error (12) on the trajectory). Of course, we observe an excellent energy and angular momentum preservations (results not shown): the error for both quantities is lower than the prescribed tolerance for all parareal iterations $k \geq 1$.

In summary, this algorithm, that projects the trajectory on the manifold $\mathcal{M}_2$ where *both* energy *and* angular momentum are constant, yields results that are slightly more accurate than those obtained with algorithms (25)–(26), where the trajectory is projected on the manifold $\mathcal{M}$ where *only* the energy is constant. However, the behavior of the trajectory is not much improved. From this prototypical analysis, it does not seem worth to project the trajectory on the manifold where several invariants (besides the energy) are constant, especially if we bear in mind that the determination of these other invariants might be a difficult task in the general case.

https://doi.org/10.1051/m2an/2012046 Published online by Cambridge University Press

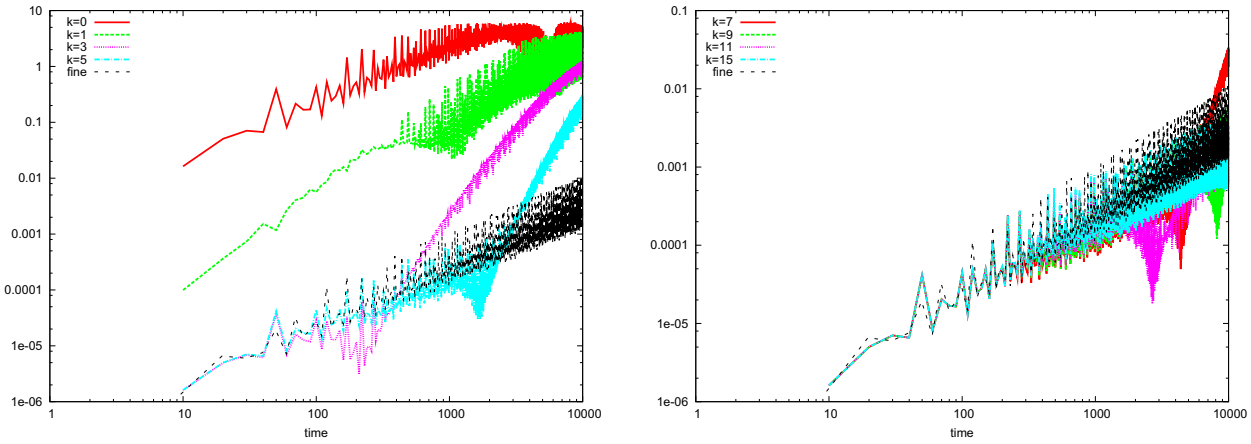


FIGURE 6. Errors on the trajectory for Kepler problem, obtained by the parareal method (25)–(28) with projection on the manifold $\mathcal{M}_2$ of constant energy and angular momentum ($\delta t = 10^{-4}$, $\mathrm{d}T = 0.01$, $\Delta T = 0.2$).

6. Symmetric parareal algorithm with symmetric projection

We have introduced in Section 5 a parareal algorithm using some projection step. Of course, just as the original parareal algorithm (4), this algorithm is neither symplectic nor symmetric (see Sect. 3.2).

Yet, as introduced and analyzed in [20], a symmetric algorithm can be coupled with an appropriate projection step, while preserving symmetry. We first briefly recall this idea in Section 6.1. The reader familiar with it can skip this section and directly proceed to Section 6.2, where we use this symmetric projection technique in our context.

6.1. Symmetric projection

Consider the constant energy manifold

$$\mathcal{M} = \left\{ y = (q,p) \in \mathbb{R}^{2d};\ H(q,p) = H_0 \right\},$$

and assume that we have at hand a symmetric algorithm with numerical flow $\Psi_{\Delta T}$. Consider now the following algorithm, introduced in [20] (see also [22], Fig. V.4.1 for a schematic representation). Assume that $y_n \in \mathcal{M}$. The approximation y_{n+1} is next defined by the set of equations

$$\begin{cases} \widetilde{y}_n = y_n + \mu \nabla H(y_n), \\ \widetilde{y}_{n+1} = \Psi_{\Delta T}(\widetilde{y}_n), \\ y_{n+1} = \widetilde{y}_{n+1} + \mu \nabla H(y_{n+1}), \end{cases} \tag{29}$$

with μ chosen such that $y_{n+1} \in \mathcal{M}$. Consequently, μ and y_{n+1} satisfy the equations

$$\begin{cases} y_{n+1} - \Psi_{\Delta T}(y_n + \mu \nabla H(y_n)) - \mu \nabla H(y_{n+1}) = 0, \\ H(y_{n+1}) = H_0. \end{cases} \tag{30}$$

We denote by $\pi^{\mathrm{sym}}_{\mathcal{M}}$ this algorithm:

$$y_{n+1} = \pi^{\mathrm{sym}}_{\mathcal{M}}(y_n) \quad \text{if and only if (29) is satisfied.}$$

https://doi.org/10.1051/m2an/2012046 Published online by Cambridge University Press

As the *same* Lagrange multiplier μ is used in the first and third lines of (29), and as $\Psi_{\Delta T}$ is symmetric, one can easily see that $\pi_{\mathcal{M}}^{\mathrm{sym}}$ is a symmetric algorithm.

To solve the nonlinear equations (30), we first recast them in the form $\mathcal{S}(y_{n+1},\mu)=0$, with

$$\mathcal{S}(y_{n+1},\mu)=\begin{pmatrix}\mathcal{S}_1(y_{n+1},\mu)\\ \mathcal{S}_2(y_{n+1},\mu)\end{pmatrix}=\begin{pmatrix} y_{n+1}-\Psi_{\Delta T}(y_n+\mu\nabla H(y_n))-\mu\nabla H(y_{n+1})\\ H(\Psi_{\Delta T}(y_n+\mu\nabla H(y_n))+\mu\nabla H(y_{n+1}))-H_0\end{pmatrix}.$$

We use a Newton-like algorithm, where the jacobian matrix of $\mathcal{S}$ is approximated by

$$\mathrm{Jac}\ \mathcal{S}(y_{n+1},\mu)\approx\begin{pmatrix} I & -\nabla H(y_n)-\nabla H(y_{n+1})\\ 0 & \nabla H(\hat{y}_{n+1})^T(\nabla H(y_n)+\nabla H(y_{n+1}))\end{pmatrix}, \tag{31}$$

with $\hat{y}_{n+1}=\Psi_{\Delta T}(y_n+\mu\nabla H(y_n))+\mu\nabla H(y_{n+1})$. To stop the Newton algorithm, we again use the three criteria C1, C2, and C3 presented above, where `err` is here the relative residu,

$$\texttt{err}=\frac{|S_1(y_{n+1},\mu)|}{|y_{n+1}|}+\frac{|S_2(y_{n+1},\mu)|}{|H_0|}.$$

An alternative to the algorithm (29) is to use $\nabla H(\widetilde{y}_{n+1})$ rather than $\nabla H(y_{n+1})$ in the third line of (29). This yields the following algorithm:

$$\begin{cases}\widetilde{y}_n=y_n+\mu\nabla H(y_n),\\ \widetilde{y}_{n+1}=\Psi_{\Delta T}(\widetilde{y}_n),\\ y_{n+1}=\widetilde{y}_{n+1}+\mu\nabla H(\widetilde{y}_{n+1}),\end{cases} \tag{32}$$

with again μ chosen such that $y_{n+1}\in\mathcal{M}$. The advantage in comparison to (29) is that the nonlinear system to be solved is easier. Indeed, once μ is identified, the approximation y_{n+1} can be obtained in an explicit fashion. The Lagrange multiplier μ now solves the equation $\mathcal{S}(\mu)=0$, with

$$\mathcal{S}(\mu)=H\left[\Psi_{\Delta T}\left(y_n+\mu\nabla H(y_n)\right)+\mu\nabla H\left(y_n+\mu\nabla H(y_n)\right)\right]-H_0,$$

which is a *scalar* nonlinear equation, to be solved for a *scalar* unknown. We denote by $\pi_{\mathcal{M}}^{\mathrm{q-sym}}$ this algorithm, that we call the *quasi-symmetric projection algorithm*:

$$y_{n+1}=\pi_{\mathcal{M}}^{\mathrm{q-sym}}(y_n)\quad\text{if and only if (32) is satisfied.}$$

Note that this algorithm is *not* symmetric. It will yet turn out to be inexpensive, and to yield interesting results.

In practice, the nonlinear problem $\mathcal{S}(\mu)=0$ is solved using a Newton algorithm, where the derivative of $\mathcal{S}$ is approximated by

$$\mathcal{S}'(\mu)\approx\nabla H(y_{n+1})^T(\nabla H(y_n)+\nabla H(\widetilde{y}_n)).$$

The algorithm is stopped as soon as one of the three criteria C1, C2, C3 is satisfied (in C1, `err` is the relative error on the energy).

6.2. Application to the parareal setting

We now wish to use the above two projection procedures, the symmetric projection algorithm $\pi_{\mathcal{M}}^{\mathrm{sym}}$ and the quasi-symmetric projection algorithm $\pi_{\mathcal{M}}^{\mathrm{q-sym}}$, in the parareal context. To this end, we return to the formalism used in Section 4.1.

Consider the first K iterations of the parareal algorithm, and denote

$$U_n:=\left(u_n^0,u_n^1,\cdots,u_n^K\right).$$

https://doi.org/10.1051/m2an/2012046 Published online by Cambridge University Press

The symmetric parareal algorithm is given by (17) (or equivalently (18)–(19)), that we here write as

$$U_{n+1} = \Psi^{\rm sym}_{\Delta T}(U_n),$$

where the map $\Psi^{\rm sym}_{\Delta T}$ is symmetric in the classical sense. We next introduce K energies, namely, for any $1 \le k \le K$,

$$H_k(U_n) := H(u_n^k)$$

and we consider a symmetric algorithm, based on the symmetric map $\Psi^{\rm sym}_{\Delta T}$ and on a symmetric projection on the manifold where all the energies H_k are preserved. Following (29), this algorithm is given by

$$\begin{aligned}
\widetilde{U}_n &= U_n + \sum_{k=1}^{K} \mu_k \nabla H_k(U_n) \\
\widetilde{U}_{n+1} &= \Psi^{\rm sym}_{\Delta T}(\widetilde{U}_n) \\
U_{n+1} &= \widetilde{U}_{n+1} + \sum_{k=1}^{K} \mu_k \nabla H_k(U_{n+1}),
\end{aligned}$$

where the Lagrange multipliers μ_k are such that $H_k(U_{n+1}) = H_0$ for any $1 \le k \le K$.

Since $\nabla H_k(U_n) = (0, \dots, 0, \nabla H(u_n^k), 0, \dots, 0)$, the above equations read

$$\begin{aligned}
\widetilde{u}_n^0 = u_n^0 \quad &\text{and} \quad \widetilde{u}_n^k = u_n^k + \mu_k \nabla H(u_n^k) \text{ for any } 1 \le k \le K, \\
\widetilde{U}_{n+1} &= \Psi^{\rm sym}_{\Delta T}(\widetilde{U}_n), \\
u_{n+1}^0 = \widetilde{u}_{n+1}^0 \quad &\text{and} \quad u_{n+1}^k = \widetilde{u}_{n+1}^k + \mu_k \nabla H(u_{n+1}^k) \text{ for any } 1 \le k \le K.
\end{aligned}$$

This yields the following algorithm, that we call the *symmetric parareal algorithm with symmetric projection.*

Algorithm 1. *Let u_0 be the initial condition.*

1. *Initialization: set $u_0^0 = u_0$, and sequentially compute $\left\{u_{n+1/2}^0\right\}_{0\le n\le N-1}$ and $\left\{u_{n+1}^0\right\}_{0\le n\le N-1}$ as*

$$u_{n+1/2}^0 = \mathcal{G}^{-1}_{-\Delta T/2}\left(u_n^0\right), \quad u_{n+1}^0 = \mathcal{G}_{\Delta T/2}\left(u_{n+1/2}^0\right).$$

   Set $\widetilde{u}_{n+1/2}^0 = u_{n+1/2}^0$.

2. *Assume that, for some $k \ge 0$, the sequence $\left\{\widetilde{u}_{n+1/2}^k\right\}_{0\le n\le N-1}$ is known. Compute the sequences $\left\{u_n^{k+1}\right\}_{0\le n\le N}$ and $\left\{\widetilde{u}_{n+1/2}^{k+1}\right\}_{0\le n\le N-1}$ at iteration $k+1$ by the following steps:*
   (a) For all $0 \le n \le N-1$, compute in parallel $\mathcal{F}_{-\Delta T/2}(\widetilde{u}_{n+1/2}^k)$, $\mathcal{F}_{\Delta T/2}(\widetilde{u}_{n+1/2}^k)$, $\mathcal{G}_{-\Delta T/2}(\widetilde{u}_{n+1/2}^k)$ and $\mathcal{G}_{\Delta T/2}(\widetilde{u}_{n+1/2}^k)$;
   (b) Set $u_0^{k+1} = u_0$;
   (c) For $0 \le n \le N-1$, sequentially compute $\widetilde{u}_{n+1/2}^{k+1}$ and u_{n+1}^{k+1} as

$$\begin{aligned}
\widetilde{u}_n^{k+1} &= u_n^{k+1} + \mu_{k+1} \nabla H(u_n^{k+1}) \\
\widetilde{u}_{n+1/2}^{k+1} &= \mathcal{G}^{-1}_{-\Delta T/2}\left[\widetilde{u}_n^{k+1} - \mathcal{F}_{-\Delta T/2}\left(\widetilde{u}_{n+1/2}^k\right) + \mathcal{G}_{-\Delta T/2}\left(\widetilde{u}_{n+1/2}^k\right)\right] \\
\widetilde{u}_{n+1}^{k+1} &= \mathcal{G}_{\Delta T/2}\left(\widetilde{u}_{n+1/2}^{k+1}\right) + \mathcal{F}_{\Delta T/2}\left(\widetilde{u}_{n+1/2}^k\right) - \mathcal{G}_{\Delta T/2}\left(\widetilde{u}_{n+1/2}^k\right) \\
u_{n+1}^{k+1} &= \widetilde{u}_{n+1}^{k+1} + \mu_{k+1} \nabla H\left(u_{n+1}^{k+1}\right)
\end{aligned}$$

   where μ_{k+1} is such that $H(u_{n+1}^{k+1}) = H_0$.

https://doi.org/10.1051/m2an/2012046 Published online by Cambridge University Press

We observe that all the expensive computations (involving the fine propagator) can again be performed in parallel. Like in the symmetric parareal algorithm, the map that defines u_{n+1}^{k+1} from u_n^{k+1} is not symmetric in the classical sense, but the map that defines $(u_{n+1}^0, u_{n+1}^1, \cdots, u_{n+1}^K)$ from $(u_n^0, u_n^1, \cdots, u_n^K)$ is symmetric.

Remark 7. Instead of using the symmetric projection scheme (29), we can use the quasi-symmetric projection scheme (32). In the above algorithm, this amounts to replacing the last line in 2c), namely

$$u_{n+1}^{k+1} = \widetilde{u}_{n+1}^{k+1} + \mu_{k+1} \nabla H(u_{n+1}^{k+1}),$$

by

$$u_{n+1}^{k+1} = \widetilde{u}_{n+1}^{k+1} + \mu_{k+1} \nabla H(\widetilde{u}_{n+1}^{k+1}),$$

where again μ_{k+1} is such that $H(u_{n+1}^{k+1}) = H_0$. We call this algorithm the *symmetric parareal algorithm with quasi-symmetric projection*.

For both the symmetric and the quasi-symmetric algorithms, the algorithmic complexity is the same as for the plain parareal algorithm.

6.3. Numerical results for the Kepler problem

We first consider Algorithm 1, namely the symmetric projection algorithm, with a projection on the constant energy manifold. The iterative projection procedure is stopped using the three criteria described above, with the parameters $\texttt{tol} = 10^{-7}$ and $N_{\text{iter}}^{\max} = 2$. Results are shown on Figures 7, 8 and 9, for the errors (12) on the trajectory, the relative errors (11) on the energy preservation and the relative errors (27) on the angular momentum preservation, respectively.

We observe that, after only 5 iterations, the errors on the trajectory are comparable to the errors on the trajectory of the fine scheme used sequentially on the whole interval $[0, 10^4]$. As expected, the energy is extremely well preserved at any parareal iteration $k \geq 1$, although, as above, we limit the number of iterations to solve the nonlinear projection step. Otherwise stated, convergence in this nonlinear procedure is reached for a very small number of iterations. We also observe that, for all parareal iterations $k \geq 7$, the angular momentum is preserved with a relative accuracy of 5×10^{-4} over the complete time range.

These results are better than those reported in Section 5.2, for algorithms (25)–(26) (namely the standard parareal algorithm coupled with a standard, *not symmetrized*, projection step). First, trajectory convergence is obtained after 5 parareal iterations, rather than 11 (compare Figs. 4 and 7). Second, numerical results illustrate the fact that the nonlinear projection step on the constant energy manifold converges more rapidly. With our newly constructed Algorithm 1, the relative error on the energy preservation is smaller than the tolerance 10^{-7} for any parareal iteration $k \geq 1$, whereas it is the case only for $k \geq 6$ for algorithms (25)–(26) (compare Figs. 3 and 8). Similarly, the angular momentum is better preserved, at any parareal iteration (compare Figs. 5 and 9).

We next consider the quasi-symmetric projection algorithm, with projection on the same manifold, with the same parameters. We observe comparable performances in terms of trajectory accuracy and angular momentum preservation as with Algorithm 1 (which uses a *symmetric*, rather than quasi-symmetric, projection). However, the preservation of energy is not as good. We refer to [12] for complete numerical results.

The final time in the simulations discussed above is $T = 10^4$. This represents 1600 periods of the system (we work with the initial conditions given in (13), for which the period of the system is 2π). This time range can hence be considered as a *large* time range. On this particular example, we have also checked that Algorithm 1 gives good results on even longer time ranges. Setting $T = 10^5$, $\texttt{tol} = 10^{-11}$ and $N_{\text{iter}}^{\max} = 2$ (and keeping the same time steps as above), we obtain *e.g.* the results shown on Figure 10 for the errors (12) on the trajectory.

7. An example in higher dimension: the outer solar system

We now consider a system in a dimension higher than those considered previously. We study here the evolution of the outer solar system, which is composed of 6 three-dimensional particles, representing the Sun and the

https://doi.org/10.1051/m2an/2012046 Published online by Cambridge University Press

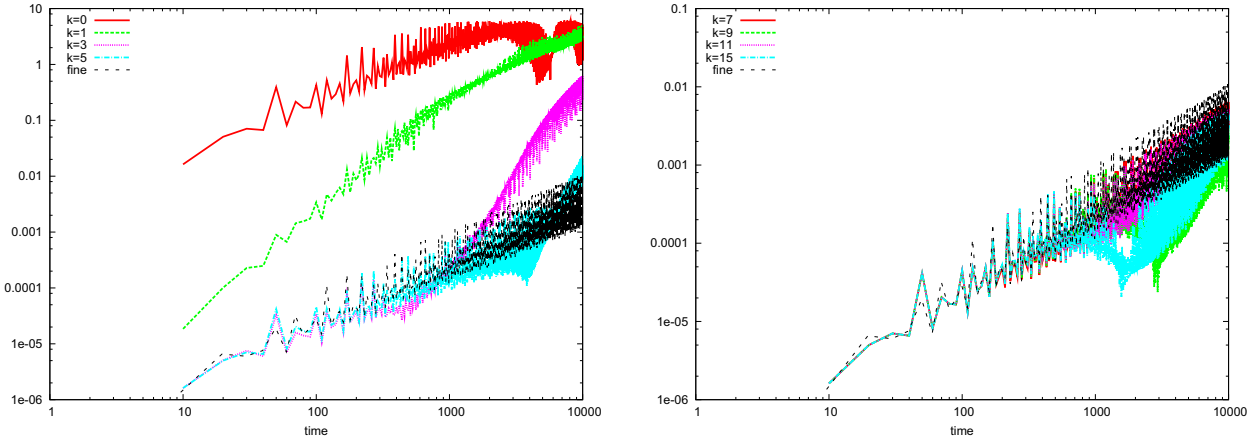


FIGURE 7. Errors on the trajectory for Kepler problem obtained by Algorithm 1 ($\delta t = 10^{-4}$, $\mathrm{d}T = 0.01$, $\Delta T = 0.2$).

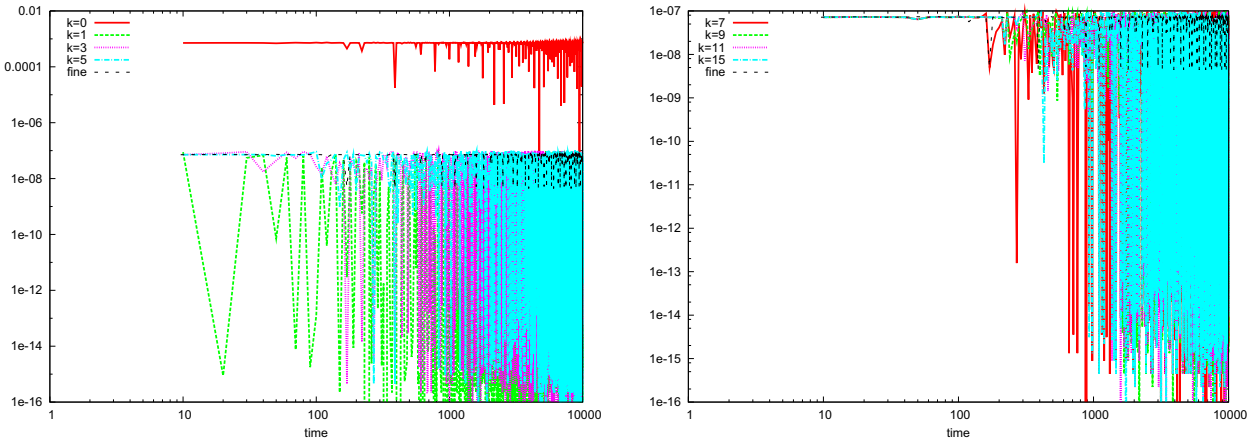


FIGURE 8. Errors on the energy for Kepler problem obtained by Algorithm 1 ($\delta t = 10^{-4}$, $\mathrm{d}T = 0.01$, $\Delta T = 0.2$).

planets from Jupiter to Pluto. The Hamiltonian reads

$$H(q,p) = \frac{1}{2} p^T M^{-1} p + V(q), \quad V(q) = -\sum_{1 \le i < j \le 6} \frac{G\, m_i\, m_j}{\|q_i - q_j\|}, \quad q \in \mathbb{R}^{18}, \quad p \in \mathbb{R}^{18}, \tag{33}$$

where M is the diagonal mass matrix $M = \mathrm{diag}(m_i)$. The values for the initial conditions, the gravitational constant G and the masses m_i of the particles are taken from ([22] Sect. I.2.4). By convention, the first particle is the Sun, the mass of which is 1000 times as large as the mass of the heaviest planet we consider.

In the sections above, we pointed out that Algorithm 1 yields good results for the Kepler problem, and converges in few iterations to the results obtained using the fine propagator in a purely sequential manner. We show here that (i) this algorithm also behaves well to simulate the outer solar system, whose dimensionality is higher, and (ii) that the coarse solver can be driven by a simplified dynamics without damaging the good properties of the algorithm. The advantage of choosing a simplified coarse integrator is that the complexity of the sequential computations using the coarse propagator is now negligible with respect to the complexity of the

https://doi.org/10.1051/m2an/2012046 Published online by Cambridge University Press

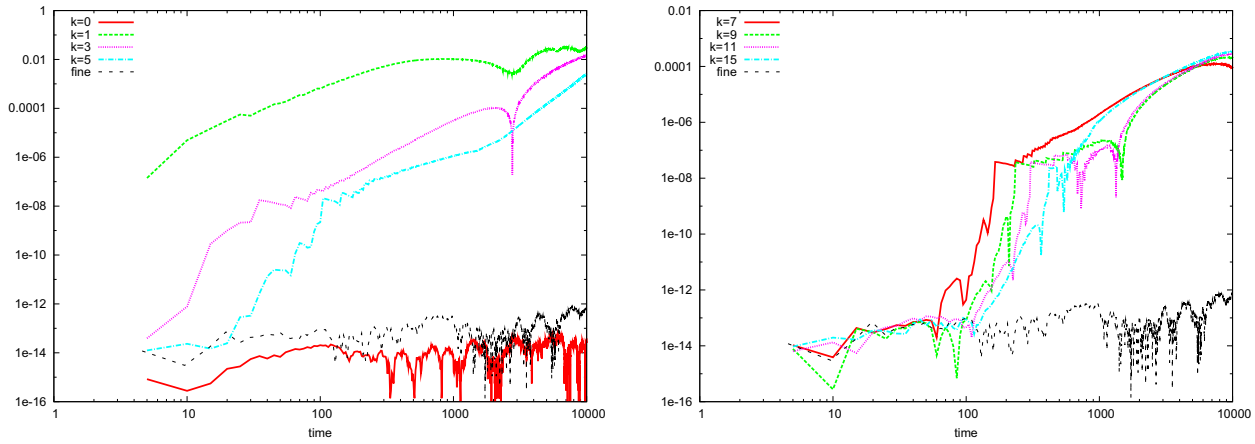


FIGURE 9. Errors on the angular momentum for Kepler problem obtained by Algorithm 1 ($\delta t = 10^{-4}$, $\mathrm{d}T = 0.01$, $\Delta T = 0.2$).

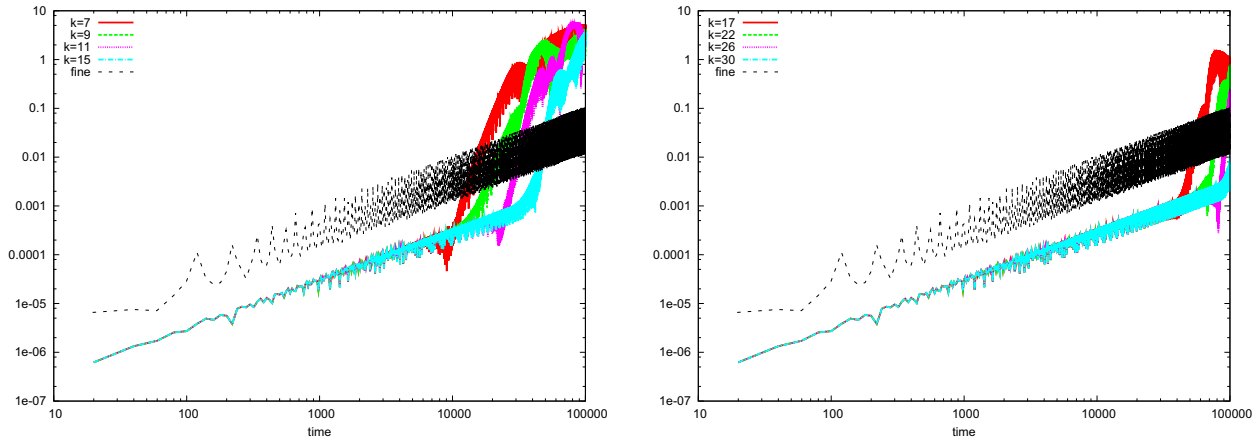


FIGURE 10. Errors on the trajectory for Kepler problem obtained by Algorithm 1 ($\delta t = 10^{-4}$, $\mathrm{d}T = 0.01$, $\Delta T = 0.2$) on a long time range.

parallel computations using the fine integrator on the interval $[n\Delta T, (n+1)\Delta T]$, in opposition to what was the case for the Kepler system (see Rem. 6).

The dynamics governed by the Hamiltonian (33) can indeed be considered as a perturbation of the dynamics driven by

$$H(q,p) = \frac{1}{2} p^T M^{-1} p + V_{\mathrm{simp}}(q), \quad V_{\mathrm{simp}}(q) = -\sum_{2 \le j \le 6} \frac{G\, m_1\, m_j}{\|q_1 - q_j\|}, \quad q \in \mathbb{R}^{18}, \quad p \in \mathbb{R}^{18}. \tag{34}$$

In V_{simp}, we only take into account the gravitational interaction between the Sun (at position q_1) and the other planets (at position q_j, $2 \le j \le 6$), and we ignore the interaction between pairs of planets. The fact that (33) is a small perturbation to (34) owes to the huge discrepancy between the mass m_1 of Sun and the masses of the planets. The latter system is less expensive to simulate, since V_{simp} is a sum of 5 terms, whereas V is a sum of 15 terms.

https://doi.org/10.1051/m2an/2012046 Published online by Cambridge University Press

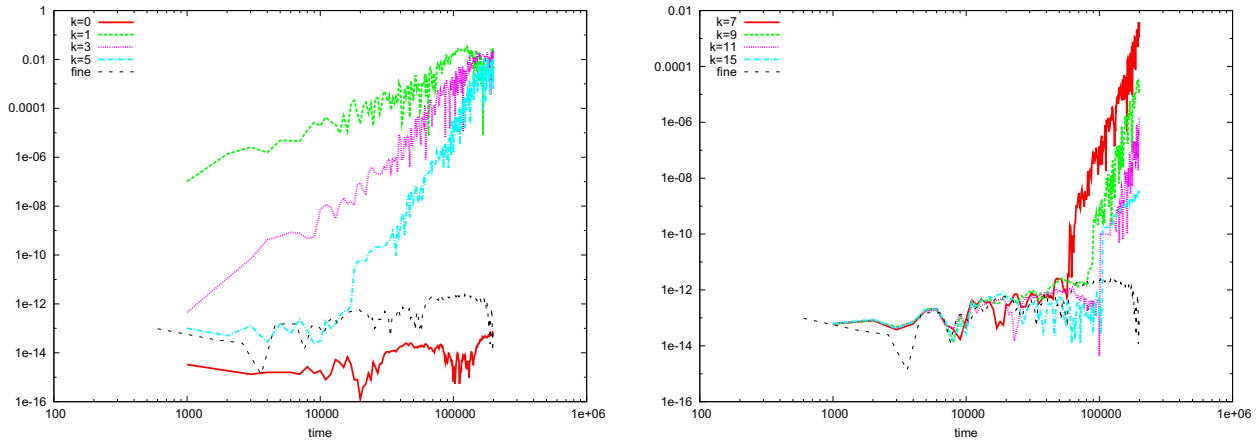


FIGURE 11. Errors (27) on the angular momentum first component for the outer solar system obtained using Algorithm 1 ($\delta t = 10^{-2}$, $\mathrm{d}T = 50$, $\Delta T = 200$).

We set $\delta t = 10^{-2}$, $\mathrm{d}T = 50$ and $\Delta T = 200$, and study Algorithm 1 over the time range $[0, T]$ with $T = 2 \times 10^5$ (same time range as in [22]). As above, we project on the manifold $\mathcal{M}$ of constant energy, and we use the three criteria C1, C2, C3 to stop the nonlinear projection step, with the parameters $N_{\mathrm{iter}}^{\max} = 2$ and $\mathtt{tol} = 10^{-11}$ (which again corresponds to the error on the energy preservation of the fine scheme).

For $k \geq 8$, the energy is preserved up to the tolerance prescribed by criterion C1 on the whole time range (result not shown). The first convergence criterion that is fulfilled is thus C1 (hence convergence in the projection step is achieved with less than $N_{\mathrm{iter}}^{\max} = 2$ iterations). Note that a larger value of $N_{\mathrm{iter}}^{\max}$ would be required to reach convergence for $k \leq 7$. We also obtain a good preservation of the angular momentum when $k \geq 5$ (see Figure 11: the error is then always smaller than 1 %). Trajectory errors are shown on Figure 12. We see that, at $k = 15$, the trajectory error is comparable to the error made by the fine propagator. Note that the error on the trajectory is already small when $k \geq 9$ (the absolute error (12) is smaller than 0.01, and the relative error is even smaller, as $\|q\|$ is of order 1 or larger, see Fig. 13).

Remark 8. We have checked that, if we set $N_{\mathrm{iter}}^{\max} = 5$, we obtain a better energy preservation (the error on the energy is then smaller than the tolerance prescribed by criterion C1 over the complete time range, at any parareal iteration $k \geq 1$), and comparable results for the angular momentum and the trajectory. However, the speed-up is then a little smaller.

Algorithm 1 also provides very good qualitative results in terms of trajectory. On Figure 13, we plot the actual trajectories of the different planets obtained at parareal iterations $k = 1$ and 5. At $k = 5$ (and actually for any $k \geq 5$, see [12]), trajectories are qualitatively similar to the ones obtained with a symplectic or symmetric integration of the solar system. We also note that, even though the trajectory is quantitatively wrong for $k = 5$, it is qualitatively correct. This is a standard observation in geometric integration. For $k \geq 10$, the trajectory is quantitatively correct.

We conclude this section by discussing the complexity of the algorithm. Note that the coarse solver integrates a simpler system than the fine solver. Its complexity, for an equal time step, is hence smaller. We are in the case of assumption (14). In addition, the average number of iterations in the nonlinear projection problem is

$$m_{\mathrm{sym-proj}} = 1.12.$$

With these choices, we need

$$K_{\mathrm{SP/sym-proj}} = 15 \text{ parareal iterations}$$

https://doi.org/10.1051/m2an/2012046 Published online by Cambridge University Press

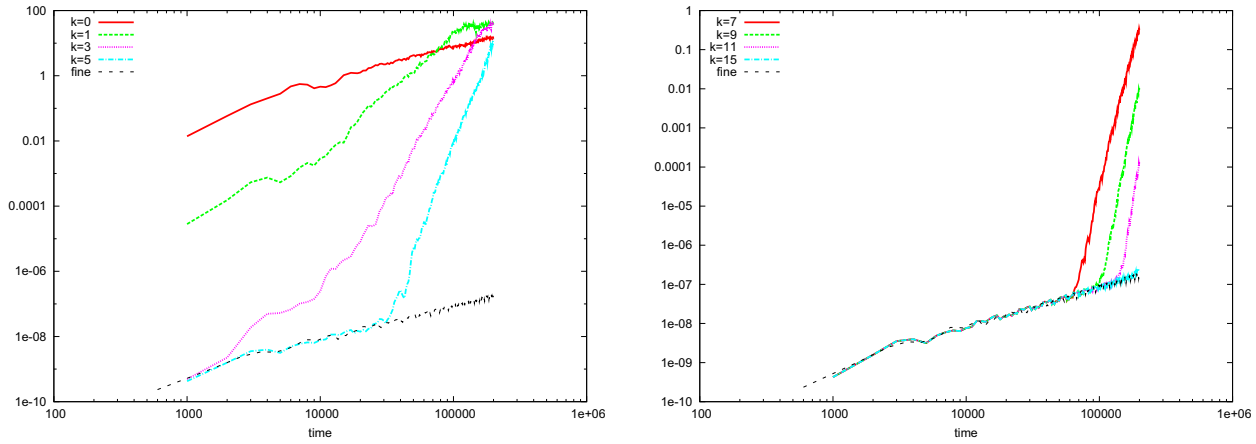


FIGURE 12. Errors (12) on the trajectory for the outer solar system obtained using Algorithm 1 ($\delta t = 10^{-2}$, $\mathrm{d}T = 50$, $\Delta T = 200$).

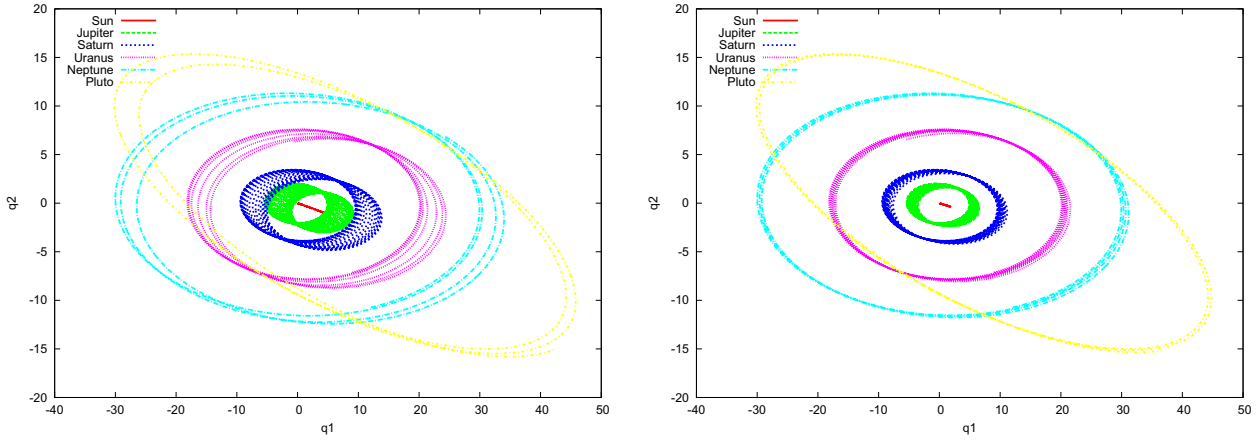


FIGURE 13. Trajectory of the outer solar system obtained using Algorithm 1 ($\delta t = 10^{-2}$, $\mathrm{d}T = 50$, $\Delta T = 200$), at iteration $k = 1$ (left) and $k = 5$ (right).

to reach convergence, namely to obtain an error on the trajectory as small as that obtained with the fine solver used sequentially over the complete time range (see Fig. 12).

Using our algorithm, and assuming that we have $T/\Delta T = 1000$ processors at our disposal, we thus obtain a speed-up of the order (see Sect. 3.3) of $T/(K_{\mathrm{SP}}\Delta T) = 1000/15 \approx 66$: for a similar accuracy, the computational time is 66 times smaller.

8. CONCLUSION

The parareal algorithm allows one to take benefit from a parallel architecture to speed up the numerical integration of systems of ODEs or time-dependent PDEs. It consists in the iterative use of a coarse and a fine propagator.

In case the system is Hamiltonian, discretization schemes that preserve some geometrical features of the underlying Hamiltonian dynamics are known to yield the best results, in terms of preservation of energy, but also preservation of other first invariants, trajectory accuracy, ... These properties are not compatible with

https://doi.org/10.1051/m2an/2012046 Published online by Cambridge University Press

the plain parareal algorithm, which has some issues when used for extremely long simulations of Hamiltonian systems.

We have identified in this article two ingredients that, when combined, yield an algorithm that is better adapted to the Hamiltonian context. The first ingredient consists in making the scheme *symmetric*. The idea is simple, but the adaptation to the parareal context is not so simple, because the original parareal algorithm cannot naturally be written as a one-step method. We have shown here how to obtain a symmetric algorithm that retains the parallel features of the original, plain, parareal algorithm. This is yet not sufficient to provide an improvement on the geometric properties of the resulting scheme and thus on the long time simulations, because of resonance issues. The second ingredient consists in *projecting* the solution on the constant energy manifold. Here, this projection is not realized at every time step but only at the coarse level. This may be the reason why such a projection, which is generally not recommended as a viable solution for a general solver, appears here to be the right complement (at least in the form of the symmetric or quasi symmetric projector) to the symmetrization of the parareal algorithm. Note also that the projection is inexpensive, as only one or two iterations are needed to solve this nonlinear procedure.

The conjunction of both ingredients above, in a case where the fine and coarse solvers have good geometric properties, is shown here to be the best choice. The symmetrized parareal algorithm has, as expected, good geometric features, apart from the resonance problem. The projection at the end of each propagation interval $[T_n, T_{n+1}]$ takes care of this resonance problem. After a limited number of parareal iterations, we obtain a trajectory which is as accurate as the one obtained using the sequential fine solver over the whole time range $[0, T]$. In turn, this convergence stabilizes the values of the invariants, besides the energy.

This allows for a substantial speed-up. Let us recall that the full efficiency offered by the parallel architecture with the parareal in time algorithm can very scarcely be obtained on systems of ODE's. In order to get a full speed-up, the parareal algorithm (in its symmetric version or not) has to be combined with other iterative procedures (such as domain decomposition approaches for PDEs, see *e.g.* [34]).

Finally, we emphasize that, in this article, in addition to energy conservation, we have chosen to illustrate the quality of the integrators by the accuracy with which the trajectories are computed. In many examples of Hamiltonian systems, for even longer simulations, the trajectories cannot be well approximated and the interest is more in *e.g.* averages of some quantities along the trajectories. The problem should then be considered from a different viewpoint as the one adopted here.

Acknowledgements. The work of Xiaoying Dai and Yvon Maday is supported in part by the Agence Nationale de la Recherche, under the grant ANR-06-CIS6-007 (PITAC). The work of Claude Le Bris and Frédéric Legoll is supported by INRIA under the grant "Action de Recherche Collaborative" HYBRID and by the Agence Nationale de la Recherche, under the grant ANR-09-BLAN-0216-01 (MEGAS).

References

[1] H.C. Andersen, Rattle: a velocity version of the Shake algorithm for molecular dynamics calculations. *J. Comput. Phys.* **52** (1983) 24–34.

[2] L. Baffico, S. Bernard, Y. Maday, G. Turinici and G. Zérah, Parallel in time molecular dynamics simulations. *Phys. Rev. E* **66** (2002) 057701.

[3] 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*, edited by R. Kornhuber, R. Hoppe, J. Périaux, O. Pironneau, O. Widlund and J. Xu. Springer Verlag, *Lect. Notes Comput. Sci. Eng.* **40** (2005) 425–432.

[4] G. Bal and Y. Maday, A parareal time discretization for nonlinear PDE's with application to the pricing of an American put, in *Recent developments in domain decomposition methods*, edited by L.F. Pavarino and A. Toselli. Springer Verlag, *Lect. Notes Comput. Sci. Eng.* **23** (2002) 189–202.

[5] G. Bal and Q. Wu, *Symplectic parareal*, in *Domain decomposition methods in science and engineering*, edited by U. Langer, M. Discacciati, D.E. Keyes, O.B. Widlund and W. Zulehner. Springer Verlag, *Lect. Notes Comput. Sci. Eng.* **60** (2008) 401–408.

[6] A. Bellen and M. Zennaro, Parallel algorithms for initial value problems for nonlinear vector difference and differential equations. *J. Comput. Appl. Math.* **25** (1989) 341–350.

[7] G. Benettin and A. Giorgilli, On the Hamiltonian interpolation of near to the identity symplectic mappings with application to symplectic integration algorithms. *J. Stat. Phys.* **74** (1994) 1117–1143.

https://doi.org/10.1051/m2an/2012046 Published online by Cambridge University Press

[8] L.A. Berry, W. Elwasif, J.M. Reynolds-Barredo, D. Samaddar, R. Sanchez and D.E. Newman, Event-based parareal: A data-flow based implementation of parareal. *J. Comput. Phys.* **231** (2012) 5945–5954.

[9] K. Burrage, *Parallel and sequential methods for ordinary differential equations*, Numerical Mathematics and Scientific Computation, Oxford Science Publications, The Clarendon Press, Oxford University Press, New York (1995).

[10] K. Burrage, Parallel methods for ODEs. *Advances Comput. Math.* **7** (1997) 1–3.

[11] P. Chartier and B. Philippe, A parallel shooting technique for solving dissipative ODE's. *Computing* **51**(3-4) (1993) 209–236.

[12] X. Dai, C. Le Bris, F. Legoll and Y. Maday, *Symmetric parareal algorithms for Hamiltonian systems*, `arXiv:preprint 1011.6222.`

[13] C. Farhat, J. Cortial, C. Dastillung and H. Bavestrello, Time-parallel implicit integrators for the near-real-time prediction of linear structural dynamic responses. *Int. J. Numer. Meth. Engng.* **67** (2006) 697–724.

[14] P. Fischer, F. Hecht and Y. Maday, A parareal in time semi-implicit approximation of the Navier Stokes equations, in *Domain decomposition methods in science and engineering*, edited by R. Kornhuber, R. Hoppe, J. Périaux, O. Pironneau, O. Widlund and J. Xu. Springer Verlag *Lect. Notes Comput. Sci. Eng.* **40** (2005) 433–440.

[15] D. Frenkel and B. Smit, *Understanding molecular simulation, from algorithms to applications*, 2nd ed., Academic Press (2002).

[16] M. Gander and S. Vandewalle, *On the superlinear and linear convergence of the parareal algorithm*, in Proceedings of the 16th International Conference on Domain Decomposition Methods, January 2005, edited by O. Widlund and D. Keyes. Springer, *Lect. Notes Comput. Sci. Eng.* **55** (2006) 291–298.

[17] M. Gander and S. Vandewalle, Analysis of the parareal time-parallel time-integration method. *SIAM J. Sci. Comput.* **29** (2007) 556–578.

[18] I. Garrido, B. Lee, G.E. Fladmark and M.S. Espedal, Convergent iterative schemes for time parallelization. *Math. Comput.* **75** (2006) 1403–1428.

[19] W. Hackbusch, *Parabolic multigrid methods*, Computing methods in applied sciences and engineering VI (Versailles, 1983), North-Holland, Amsterdam (1984) 189–197.

[20] E. Hairer, Symmetric projection methods for differential equations on manifolds. *BIT* **40** (2000) 726–734.

[21] E. Hairer and C. Lubich, The life span of backward error analysis for numerical integrators. *Numer. Math.* **76** (1997) 441–462.

[22] E. Hairer, C. Lubich and G. Wanner, Geometric numerical integration: structure-preserving algorithms for ordinary differential equations. *Springer Ser. Comput. Math.* **31** (2002).

[23] P. Joly, Numerical methods for elastic wave propagation, in *Waves in nonlinear pre-stressed materials*, edited by M. Destrade and G. Saccomandi. Springer-Verlag (2007) 181–281.

[24] P. Joly, The mathematical model for elastic wave propagation. Effective computational methods for wave propagation, in *Numer. Insights, Chapman & Hall/CRC* (2008) 247–266.

[25] J. Laskar, A numerical experiment on the chaotic behavior of the Solar system. *Nature* **338** (1989) 237–238.

[26] J. Laskar, Chaotic diffusion in the Solar system. *Icarus* **196** (2008) 1–15.

[27] B. Leimkuhler and S. Reich, Simulating Hamiltonian dynamics. *Cambridge University Press* (2004).

[28] B.J. Leimkuhler and R.D. Skeel, Symplectic numerical integrators in constrained Hamiltonian systems. *J. Comput. Phys.* **112** (1994) 117–125.

[29] E. Lelarasmee, A.E. Ruehli and A.L. Sangiovanni-Vincentelli, The waveform relaxation method for time-domain analysis of large scale integrated circuits. *IEEE Trans. CAD of IC Syst.* **1** (1982) 131–145.

[30] J.-L. Lions, Y. Maday and G. Turinici, A parareal in time discretization of PDE's. *C. R. Acad. Sci. Paris, Ser. I* **332** (2001) 661–668.

[31] Y. Maday, *The parareal in time algorithm*, in Substructuring Techniques and Domain Decomposition Methods, edited by F. Magoulès. Chapt. 2, Saxe-Coburg Publications, Stirlingshire, UK (2010) 19–44. `doi:10.4203/csets.24.2`

[32] Y. Maday and G. Turinici, A parareal in time procedure for the control of partial differential equations. *C. R. Acad. Sci. Paris Ser. I* **335** (2002) 387–392.

[33] Y. Maday and G. Turinici, A parallel in time approach for quantum control: the parareal algorithm. *Int. J. Quant. Chem.* **93** (2003) 223–228.

[34] Y. Maday and G. Turinici, The parareal in time iterative solver: a further direction to parallel implementation, in *Domain decomposition methods in science and engineering*, edited by R. Kornhuber, R. Hoppe, J. Périaux, O. Pironneau, O. Widlund and J. Xu. Springer Verlag, *Lect. Notes Comput. Sci. Eng.* **40** (2005) 441–448.

[35] A. Quarteroni and A. Valli, *Domain decomposition methods for partial differential equations*, Numerical Mathematics and Scientific Computation, Oxford Science Publications, The Clarendon Press, Oxford University Press, New York (1999).

[36] S. Reich, Backward error analysis for numerical integrators. *SIAM J. Numer. Anal.* **36** (1999) 1549–1570.

[37] J.-P. Ryckaert, G. Ciccotti and H.J.C. Berendsen, Numerical integration of the cartesian equations of motion of a system with constraints: molecular dynamics of n-alkanes. *J. Comput. Phys.* **23** (1977) 327–341.

[38] P. Saha, J. Stadel and S. Tremaine, A parallel integration method for Solar system dynamics. *Astron. J.* **114** (1997) 409–414.

[39] P. Saha and S. Tremaine, Symplectic integrators for solar system dynamics. *Astron. J.* **104** (1992) 1633–1640.

[40] J.M. Sanz-Serna and M.P. Calvo, *Numer. Hamiltonian Problems*. Chapman & Hall (1994).

[41] G.A. Staff and E.M. Rønquist, Stability of the parareal algorithm, in *Domain decomposition methods in science and engineering*, edited by R. Kornhuber, R. Hoppe, J. Périaux, O. Pironneau, O. Widlund and J. Xu. Springer Verlag, *Lect. Notes Comput. Sci. Eng.* **40** (2005) 449–456.

[42] A. Toselli and O. Widlund, Domain decomposition methods–algorithms and theory. *Springer Ser. Comput. Math.* **34** (2005).

https://doi.org/10.1051/m2an/2012046 Published online by Cambridge University Press