C. R. Acad. Sci. Paris, Ser. I 353 (2015) 843–847

Contents lists available at ScienceDirect

C. R. Acad. Sci. Paris, Ser. I

www.sciencedirect.com

Partial differential equations/Numerical analysis

Asynchronous numerical scheme for modeling hyperbolic systems

Schéma numérique asynchrone du second ordre pour la modélisation des systèmes hyperboliques

Asma Toumi [a], Guillaume Dufour [a], Ronan Perrussel [b], Thomas Unfer [b]

[a] *ONERA, 2, avenue Edouard-Belin, 31400 Toulouse, France*
[b] *Université de Toulouse, LAPLACE, CNRS/UPS/INPT, Toulouse, France*

ARTICLE INFO

Article history:
Received 16 March 2015
Accepted after revision 30 June 2015
Available online 23 July 2015

Presented by Oliver Pironneau

ABSTRACT

We present an asynchronous method for the explicit integration of multi-scale partial differential equations. This method is restricted by a local CFL condition rather than the traditional global CFL condition. First, we developed an upwind asynchronous forward Euler scheme for the transport equation and we proved that the asynchronous scheme is first order convergent. To improve the convergence rate of the asynchronous scheme, we derived an asynchronous Runge–Kutta 2 scheme from a standard explicit Runge–Kutta method.

© 2015 Académie des sciences. Published by Elsevier Masson SAS. All rights reserved.

RÉSUMÉ

Nous présentons une méthode asynchrone pour l'intégration explicite des équations aux dérivées partielles multi-échelles. Cette méthode est limitée par une condition CFL locale plutôt que par la condition CFL globale classique. Tout d'abord, nous avons développé un schéma d'Euler asynchrone pour la discrétisation de l'équation de transport et nous avons prouvé que le schéma asynchrone est convergent au premier ordre. Pour la montée en ordre, nous avons proposé un schéma Runge–Kutta 2 asynchrone, dérivé d'un schéma RK2 classique, pour obtenir un schéma numériquement d'ordre 2.

© 2015 Académie des sciences. Published by Elsevier Masson SAS. All rights reserved.

1. Introduction

Numerical time-stepping integration techniques used for modeling physical systems, such as combustion or atmospheric plasmas, generally fall into two categories: explicit and implicit schemes. In the explicit schemes, the time step is limited by the Courant–Friedrichs–Levy stability criterion (CFL condition). In contrast, an unconditionally stable implicit method is generally not suitable for strongly coupled problems. In the literature, many local time stepping (LTS) methods have

E-mail addresses: toumiasma@yahoo.fr (A. Toumi), Guillaume.Dufour@onera.fr (G. Dufour), perrussel@laplace.univ-tlse.fr (R. Perrussel), thomas.unfer@laplace.univ-tlse.fr (T. Unfer).

http://dx.doi.org/10.1016/j.crma.2015.06.010
1631-073X/© 2015 Académie des sciences. Published by Elsevier Masson SAS. All rights reserved.

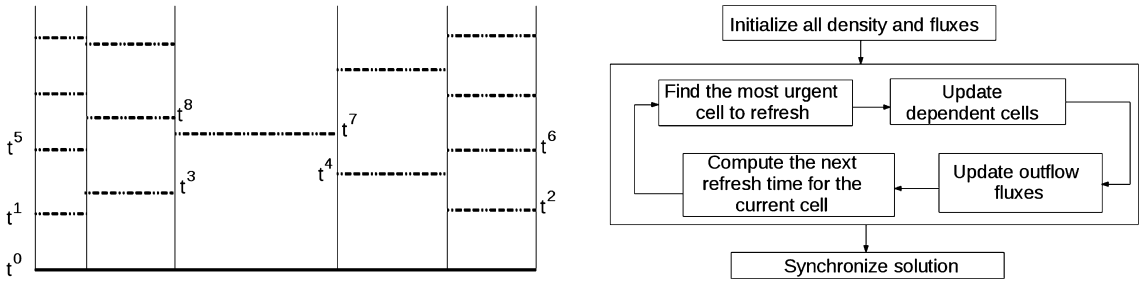

Fig. 1. Left: five elements and pre-computed update times t^i. Right: flow chart of the asynchronous algorithm.

been developed. Flaherty et al. [2] developed a parallel, adaptive discontinuous Galerkin method with a local forward Euler scheme that relies on interpolating values in time at interfaces between time steps of different sizes. However, they only presented first-order results. Various local time stepping schemes for hyperbolic conservation laws were proposed [6,7]. For these different algorithms, local time steps are selected to be fractions of the global time step. Grote et al. [3] derived an explicit LTS scheme from standard explicit Runge–Kutta (RK) methods. They have proved that their scheme preserves the accuracy of the original RK scheme. However, their approach is applicable only in the case of two distinct regions in the mesh: a "coarse" region with the larger elements and a "fine" region with the smaller elements. In the fine region, the time step must be a fraction of the coarse time step. Recently, a new time integration approach has been introduced in [4]. The proposed algorithm permits the selection of independent time steps in each mesh element. However, this approach is only applicable to Hamiltonian systems. In [5], Omelchenko and Karimabadi considered an asynchronous approach on a diffusion–advection–reaction equation in one dimension. Their method is based on a discrete-event simulation. However, they did not present results for higher dimensions and they did not discuss the convergence rate of their method. In this paper we focus on the asynchronous method proposed by one of the co-author, Thomas Unfer. His method uses local stability criteria. It was noticed that the asynchronous scheme reduces numerical diffusion and CPU time in comparison to the classical scheme [9]. In [8], Unfer has also proposed an extension of the asynchronous integration procedure to higher order. We proved that the asynchronous scheme as it was presented in [8] is at most of order one. Hence we propose a new asynchronous RK2 scheme.

This paper is outlined as follows: in Section 2, we apply the asynchronous method to discretize the transport equation and present theoretical results for the stability and first order convergence of the asynchronous scheme. In Section 3, we present the asynchronous Runge–Kutta 2 (ARK2) algorithm and we assess the order of convergence via numerical simulations.

2. Asynchronous scheme

The asynchronous algorithm permits the selection of independent time steps in each mesh element so that the local time steps do not bear an integral relation to each other. The time advance is performed by organizing computational tasks into a priority queue based on their pre-computed update times (see Fig. 1 left, where the superscript i in the update time t^i denotes the order in the priority queue). To illustrate the asynchronous method, we consider the advection equation

$$\frac{\partial u}{\partial t} + \nabla \cdot (vu) = 0, \quad \forall (x,t) \in \Omega \times]0, +\infty[, \qquad u(x,0) = u_0(x), \quad \forall x \in \Omega, \tag{1}$$

where v is a non-zero vector, Ω a polygonal domain in $\mathbb{R}^d$ $(d \geqslant 1)$ and u_0 a given initial condition. We suppose periodic boundary conditions.

Let $\mathcal{T} = \{K_i,\ i = 1, \ldots, N\}$ be a partition of the domain Ω in N polyhedral volumes K_i. For all $i \in [1, N]$, $\mathcal{N}(i) = \{j,\ |K_i \cap K_j| \neq 0\}$ is the set of indices of neighboring volumes of K_i; $|K_i \cap K_j|$ denotes the $(d-1)$ positive measure of $K_i \cap K_j$. If $j \in \mathcal{N}(i)$, we denote by $n_{i,j}$ the unit normal on $K_i \cap K_j$ that points out from K_i and by $\mathbf{N}_{i,j} = |K_i \cap K_j| n_{i,j}$. Let $\mathcal{N}^+(i) = \{j \in \mathcal{N}(i),\ v.n_{i,j} > 0\}$ and $\mathcal{N}^-(i) = \{j \in \mathcal{N}(i),\ v.n_{i,j} < 0\}$.

The spatial discretization of (1) is done by the first order upwind finite volume scheme

$$\frac{\partial u_i}{\partial t}(t) = -\frac{1}{|K_i|}\left(\sum_{j \in \mathcal{N}^+(i)} v \cdot \mathbf{N}_{i,j} u_i(t) + \sum_{j \in \mathcal{N}^-(i)} v \cdot \mathbf{N}_{i,j} u_j(t) \right) = -\frac{1}{|K_i|} \sum_{j \in \mathcal{N}(i)} F_{i,j}(t), \tag{2}$$

where the density $u_i(t)$ is an approximation of $\frac{1}{|K_i|}\int_{K_i} u(x,t)\mathrm{d}x$ and numerical fluxes $F_{i,j}$ are defined by: $F_{i,j}(t) = v \cdot \mathbf{N}_{i,j} u_i(t)$, $\forall j \in \mathcal{N}^+(i)$ (outflow fluxes) and $F_{i,j}(t) = v \cdot \mathbf{N}_{i,j} u_j(t)$, $\forall j \in \mathcal{N}^-(i)$ (inflow fluxes).

2.1. Asynchronous algorithm

Three main phases summarize the algorithm; see also Fig. 1 right. First, at the start-up time all densities and fluxes are initialized. Then, a second phase is carried out by continuously applying the following steps until the global simulation clock is advanced past the simulation finish time: find the most urgent cell to be refreshed by using the CFL condition, compute the values of the densities that are needed to compute the outflow fluxes (in the case of an upwind scheme only the current cell K_i and the neighbors K_j, $\forall j \in \mathcal{N}^+(i)$ are involved), update the values of the outflow fluxes and then compute the next refresh time of this cell. Finally, the third phase is to build the solution at the output time.

2.2. Properties of the asynchronous scheme

Classical estimations ensuring the order of accuracy (consistency) and stability are not directly compatible with the asynchronous aspect. It therefore becomes necessary to adapt the notion of consistency to this concept as well as to define a suitable norm to justify the stability of this approach.

In the case of an asynchronous integration in time, the solution in a given time depends on all previous times. Then, to prove the stability of this scheme we need to define the asynchronous infinity norm $\|\cdot\|_{\text{asyn},\infty}$ by $\|u^{t^n}\|_{\text{asyn},\infty} := \sup_{i,p\leqslant n} |u_i^{t^p}|$.

Proposition 2.1. *The asynchronous scheme is stable in the asynchronous infinity norm $\|.\|_{\text{asyn},\infty}$ under the local CFL condition on the time-step $\triangle t_i$ relative to the element K_i:*

$$\triangle t_i \leqslant \frac{|K_i|}{\sum_{j\in\mathcal{N}^+(i)} v\cdot \mathbf{N}_{i,j}}, \quad \forall i = 1,\dots,N. \tag{3}$$

Let us recall that, on a regular rectangular grid, the classical upwind explicit finite volume scheme is stable and strongly consistent. On irregular meshes, first-order finite volume schemes are not strongly consistent, their truncation error is only $O(1)$. These results can be easily extended to the case of the asynchronous integration in time. In the case of an explicit finite volume scheme on arbitrary meshes, the derivation of a priori optimal error estimates is still an open problem even for the scalar linear advection equation. Under some assumptions on the mesh and the scheme, an $O(h^{1/2})$ convergence rate has been established [10]. Bouche et al. [1] studied the convergence rate of the upwind finite volume method applied to the linear advection equation on a polygonal domain of $\mathbb{R}^d$. They construct first a set of geometric correctors that depend only on the mesh and on the advection vector, but not on the solutions to the advection equation. Then they proved an $O(h)$ convergence rate of the explicit upwind scheme.

To prove the convergence of the asynchronous scheme, we applied the method proposed by Bouche. We first considered a sequence of geometric correctors in $\mathbb{R}^d$, $(\Gamma_i)_{i=1,\dots,N}$ as in [1]. If there exists C such that $\|\Gamma\|_{\text{asyn},\infty} \leqslant Ch$ then, we have the following result:

Theorem 2.1. *Let u be the exact solution to (1), supposed of class C^2. Assume that the mesh is regular enough to ensure that $\triangle t_i - \triangle t_j = O(h^2)$ for all $K_i \in \mathcal{T}$ and $\forall j \in \mathcal{N}(i)$. Under the local CFL conditions, the asynchronous upwind scheme is convergent of order one.*

We proved that the asynchronous scheme is convergent in the first order. However, the approximation of order one is not sufficient for many applications. Hence the idea to define an asynchronous scheme of order 2. This will be discussed in the next section.

3. Asynchronous Runge–Kutta 2 scheme

We propose in this section an asynchronous Runge–Kutta 2 (ARK2) scheme. We suppose an arbitrary space discretization of a given linear partial differential equation with initial conditions but without a source term, for the sake of simplicity. This leads to a system of coupled ordinary differential equations

$$\frac{\mathrm{d}y_i}{\mathrm{d}t}(t) = B_i y(t), \quad \forall K_i \in \mathcal{T}, \tag{4}$$

where y_i and y are respectively vectors that contain the degrees of freedom (dofs) in the element K_i and in all the elements, and B_i a matrix coming from the space discretization.

For each cell, the source term depends only on the current cell, whereas the fluxes depend also on neighbors. So to apply the ARK2 method, we first need to split the term $B_i y$ in three contributions: V_i is a local "volume" contribution, $F^+_{i,j}$ is a local "flux" contribution and $F^-_{i,j}$ is a "flux" contribution from the neighbor. Then (4) becomes:

$$\frac{\mathrm{d}y_i}{\mathrm{d}t}(t) = V_i y_i(t) + \sum_{j\in\mathcal{N}(i)} F^+_{i,j} y_i(t) + F^-_{i,j} y_j(t), \quad \forall K_i \in \mathcal{T}. \tag{5}$$

We denote by y_i^m the approximation of $y_i(m\,\triangle\, t_i)$ and by $\left(\frac{\Delta y}{\Delta t}\right)_i^m$ the approximation of $\frac{\mathrm{d}y_i}{\mathrm{d}t}(m\Delta t_i)$.

3.1. Algorithm

Pre-computed update times correspond to multiples of $\triangle t_i$ in each K_i. The time advance is performed using a priority queue as explained in the beginning of Section 2; t_{cur} is the current simulation time.

At t_{cur} such that $t_{\text{cur}}/\triangle t_i$ is an integer p, we have to perform the following operations.

(i) Compute the tentative dofs at t_{cur}

$$\tilde{y}_i(t_{\text{cur}}) = \tilde{y}_i(t_{\text{pre},i}) + (t_{\text{cur}} - t_{\text{pre},i})\left(\frac{\Delta\tilde{y}}{\Delta t}\right)_i(t_{\text{mid},i}), \quad t_{\text{mid},i} = t_{\text{cur}} - \left(\frac{t_{\text{cur}} - t_{\text{pre},i}}{2}\right)$$

where $t_{\text{pre},i}$ is the last time when the tentative dofs has been updated. The tentative mid-point slope is defined as follows

$$\begin{aligned}\left(\frac{\Delta\tilde{y}}{\Delta t}\right)_i(t_{\text{mid},i}) = V_i\left(y_i^{p-1} + \frac{\Delta t_i}{2}\left(\frac{\Delta y}{\Delta t}\right)_i^{p-1}\right)\\ + \sum_{j\in N(i)}\left[F_{i,j}^+\left(y_i^{p-1} + \left(t_{\text{mid},i,j} - (p-1)\Delta t_i\right)\left(\frac{\Delta y}{\Delta t}\right)_i^{p-1}\right)\right.\\ \left. + F_{i,j}^-\left(y_j^{n_j} + \left(t_{\text{mid},i,j} - n_j\Delta t_j\right)\left(\frac{\Delta y}{\Delta t}\right)_j^{n_j}\right)\right]\end{aligned}$$

where $n_j = \left\lfloor \frac{t_{\text{pre},i}}{\Delta t_j} \right\rfloor$, $t_{\text{mid},i,j} = t_{\text{cur}} - \left(\frac{t_{\text{cur}} - t_{\text{pre},i,j}}{2}\right)$ with $t_{\text{pre},i,j} = \max(n_j\Delta t_j, (p-1)\Delta t_i)$.

(ii) For all the neighbors of the element i, *i.e.* $j \in \mathcal{N}(i)$, we update the tentative values

$$\tilde{y}_j(t_{\text{cur}}) = \tilde{y}_j(t_{\text{pre},j}) + (t_{\text{cur}} - t_{\text{pre},j})\left(\frac{\Delta\tilde{y}}{\Delta t}\right)_j(t_{\text{mid},j}),$$

$t_{\text{pre},j}$ is the last time where the tentative dofs have been updated and $t_{\text{mid},j}$ is the "mid-point" time.

(iii) Affect the dofs and update the last refresh times

$$y_i^p = \tilde{y}_i(t_{\text{cur}}), \quad t_{\text{pre},i} = t_{\text{cur}} \text{ and } t_{\text{pre},j} = t_{\text{cur}}, \quad \forall j \in \mathcal{N}(i).$$

(iv) After all elements K_i such that $t_{\text{cur}}/\Delta t_i$ is an integer have been updated, update their slopes

$$\left(\frac{\Delta y}{\Delta t}\right)_i^p = V_i y_i^p + \sum_{j\in N(i)} F_{i,j}^+ y_i^p + F_{i,j}^-\left(y_j^{n_j} + (p\Delta t_i - n_j\Delta t_j)\left(\frac{\Delta y}{\Delta t}\right)_j^{n_j}\right), \quad \text{where} \quad n_j = \left\lfloor \frac{t_{\text{cur}}}{\Delta t_j} \right\rfloor.$$

3.2. Numerical tests

In this subsection, we verify via numerical simulations the order of convergence of the ARK2 scheme. We consider the one-dimensional Maxwell equation:

$$\frac{\partial(\varepsilon E)}{\partial t} - \frac{\partial H}{\partial x} = 0, \quad \frac{\partial(\mu H)}{\partial t} - \frac{\partial E}{\partial x} = 0 \tag{6}$$

and we suppose periodic boundary conditions. Spatial discretization is performed with the classical P^1-discontinuous Galerkin scheme. We then apply the ARK2 algorithm for several meshes.

To study the convergence rate of the asynchronous scheme, we introduce several sequences of meshes. We first consider meshes where the local time steps are multiple of the smallest time step (see Fig. 2 Mesh 1 with $q = 4$ and Mesh 2 with $p = 4$ and $q = 8$) before more general meshes (see Fig. 3 Mesh 3 with $p = 3$, $q = 2$ and Mesh 4 where $p/q < 1$ with $p = 2$, $q = 3$). We finally test the algorithm on a mesh where the time steps are completely independent. The positions x_i of the vertices of this mesh follow a polynomial law with parameter α defining the slope at $x = 0.5$ and thus:

$$x_i = 4(1-\alpha)\left(\frac{i-1}{N-1} - \frac{1}{2}\right)^3 + \alpha\left(\frac{i-1}{N-1} - \frac{1}{2}\right) + \frac{1}{2}, \quad N \text{ is the number of cells.} \tag{7}$$

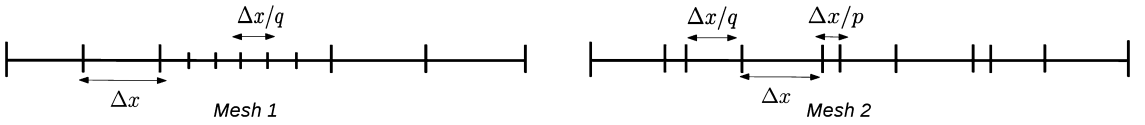

Fig. 2. In Meshes 1 (with $q = 4$) and 2 (with $p = 4$ and $q = 8$), the local time steps are multiple of the smallest time step.

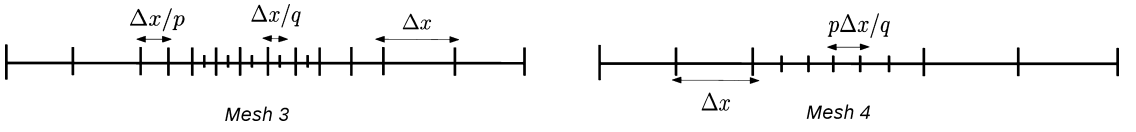

Fig. 3. In Meshes 3 (with $p = 3$, $q = 2$) and 4 (with $p = 2$, $q = 3$), the local time steps are not multiple of the smallest time step.

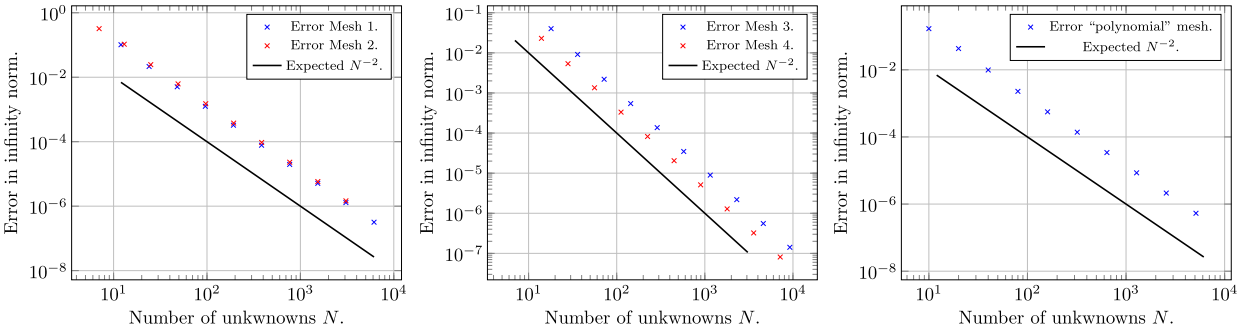

Fig. 4. (Color online.) Second order ARK2 scheme for several type of meshes.

In Fig. 4, the infinity norm of the error versus the number of cells is shown for each sequence of meshes. It can be observed that the ARK2 is a second-order convergent scheme independent of the type of mesh.

The presented asynchronous scheme has a little impact on precision but does not influence the order of convergence. Moreover, comparing with the classical integration, the asynchronous scheme is effective in terms of computation time. For example, in the case of the polynomial mesh (7) the estimated factor of acceleration is of the order of 11. Note that in practical implementation, the asynchronous speed-up is generally slower than theoretically expected, mainly because the asynchronous scheme access the data in memory in a more random manner. This overhead depends on the implementation, but in our experiments we observed that it is hard to reduce by more than by a factor of three. This means that for application with a theoretical speed-up exceeding 3, the asynchronous integration provides effective CPU speed-up. Application of the asynchronous scheme to nonlinear problems and theoretical study will be the subject of future works.

Acknowledgements

This research is granted by the project MACOPA (ANR-11-MONU-0019).

References

[1] D. Bouche, J.-M. Ghidaglia, F. Pascal, Error estimate and the geometric corrector for the upwind finite volume method applied to the linear advection equation, SIAM J. Numer. Anal. 43 (2) (2005) 578–603 (electronic).
[2] J.E. Flaherty, R.M. Loy, M.S. Shephard, B.K. Szymanski, J.D. Teresco, L.H. Ziantz, Adaptive local refinement with octree load-balancing for the parallel solution of three-dimensional conservation laws, J. Parallel Distrib. Comput. 47 (1997) 139–152.
[3] M.J. Grote, M. Mehlin, T. Mitkova, Runge–Kutta-based explicit local time-stepping methods for wave propagation, February 2014, preprint No. 2014-05.
[4] A. Lew, J.E. Mardsen, M. Ortiz, M. West, Asynchronous variational integrators, Arch. Ration. Mech. Anal. 167 (2003) 85.
[5] Y.A. Omelchenko, H. Karimabadi, Self-adaptive time integration of flux-conservative equations with sources, J. Comput. Phys. 216 (1) (2006) 179–193.
[6] S. Osher, R. Sanders, Numerical approximations to nonlinear conservation laws with locally varying time and space grids, Math. Comput. 41 (1983) 321–336.
[7] H.Z. Tang, Warnecke, A class of high-resolution schemes for hyperbolic conservation laws and convection-diffusion equations with varying time and space grids, preprint, 2003.
[8] T. Unfer, An asynchronous framework for the simulation of the plasma/flow interaction, J. Comput. Phys. 236 (2013) 229–246.
[9] T. Unfer, J.-P. Boeuf, F. Rogier, An asynchronous scheme with local time-stepping for multi-scale transport problems: application to gas discharges, J. Comput. Phys. 227 (1) (2007) 898–918.
[10] J.-P. Vila, P. Villedieu, Convergence of an explicit finite volume scheme for first order symmetric systems, Numer. Math. 94 (3) (2003) 573–602.