Bol. Soc. Esp. Mat. Apl.
n°42(2008), 5–19

ANALYSIS OF THE PARAREAL ALGORITHM APPLIED TO HYPERBOLIC PROBLEMS USING CHARACTERISTICS

MARTIN J. GANDER

Section de Mathématiques, Université de Genève.

martin.gander@math.unige.ch

Abstract

The parareal algorithm is a time domain decomposition algorithm for the time parallel approximation of solutions of evolution problems. It can be interpreted as a multiple shooting method for initial value problems with a particular choice of the approximate Jacobian on a coarse grid. The method can give significant speedup for non-linear systems of ordinary differential equations and discretized diffusive partial differential equations, but has been reported to be less effective for hyperbolic problems. We prove in this paper a convergence result for the advection equation using the technique of characteristics. Our analysis also reveals limitations of the method when applied to the second order wave equation.

Palabras clave: *Time parallel time integration methods, multiple shooting for initial value problems, parareal algorithm, hyperbolic problems, advection equation, second order wave equation.*
Clasificación por materias AMS: *65R20, 45L05, 65L20*

1 Introduction

Time domain decomposition methods have a long history: already Nievergelt proposed in [19] a parallel algorithm based on a decomposition of the time direction for the solution of ordinary differential equations. While his idea targeted large scale parallelism, Miranker and Liniger proposed a little later in [18] a family of naturally parallel Runge Kutta methods for small scale time parallelism. Waveform relaxation methods, introduced by Lelarasmee, Ruehli and Sangiovanni-Vincentelli in [14] for the large scale simulation in VLSI design, are another fundamental way to introduce time parallelism into the solution of evolution problems. A more recent time parallel algorithm which we will study in this paper is the parareal algorithm, see Lions, Maday and Turinici [15]. For an up to date historical review and further references, see [11].

The parareal algorithm is a time domain decomposition method, based on multiple shooting with an approximate Jacobian on a coarse grid. A detailed derivation of the algorithm and relations to other algorithms can be found in [11]. This reference also contains sharp convergence estimates for linear problems, including discretizations of the heat equation and the advection equation on unbounded domains, which show that while the method works well in the diffusive case, it is not effective in the advective case on unbounded domains. For nonlinear problems, the parareal algorithm has been analyzed in [9], and a sequence of numerical experiments showed that substantial speedup can be obtained for a problem from chemical reactions, the computation of satellite orbits, and for the Lorentz equations, which are a simplistic model for weather prediction, one of the key applications for time parallel algorithms, since computations need to be performed in real time, and thus any speedup is welcome, even if it is suboptimal, which is often the case for time parallel algorithms. More substantial numerical experiments can be found for fluid and structure problems in [6], for the Navier-Stokes equations in [8], and for reservoir simulation in [12]. Several variants of the method have been proposed, see for example [6, 13]. The algorithm has been further analyzed in [16, 17], and its stability is investigated in [3, 20].

For hyperbolic problems, the parareal algorithm can have performance problems, as it was pointed out in [6] and [11]. An interesting modification of the parareal algorithm was then proposed in [5] and [7], and further analyzed in the context of shooting methods in [10]. Numerical experiments in [11] however had shown that for advection equations on bounded domains, the parareal algorithm can be effective, and approximately linear convergence was observed, which could not be explained by the Fourier analysis used in [11]. We prove in this paper a convergence result for the advection equation on bounded domains. Our result is based on the technique of characteristics, which is the main novelty in the analysis of the parareal algorithm, and permits a generalization to non-constant coefficient and non-linear problems. We then show, using again the method of characteristics, the limitations of the parareal algorithm applied to the second order wave equation. We illustrate our results by numerical experiments.

2 Derivation of the Parareal Algorithm

The parareal algorithm is a time parallel algorithm for the solution of the general nonlinear system of ordinary differential equations

$$u'(t) = f(u(t)), \quad t \in (0,T), \quad u(0) = u^0, \tag{1}$$

where $f : \mathbb{R}^M \longrightarrow \mathbb{R}^M$ and $u : \mathbb{R} \longrightarrow \mathbb{R}^M$. To obtain a time parallel algorithm for (1), we follow the derivation in [9]: we decompose the time domain $\Omega = (0,T)$ into N time subdomains $\Omega_n = (T_n, T_{n+1})$, $n = 0, 1, \ldots N-1$, with $0 = T_0 < T_1 < \ldots < T_{N-1} < T_N = T$, and $\Delta T_n := T_{n+1} - T_n$, and consider on each time subdomain the evolution problem

$$u_n'(t) = f(u_n(t)),\ t \in (T_n, T_{n+1}),\ u_n(T_n) = U_n,\ n = 0, 1, \ldots, N-1, \tag{2}$$

where the initial values U_n need to be determined such that the solutions on the time subdomains Ω_n coincide with the restriction of the solution of (1) to

Ω_n, i.e. the U_n need to satisfy the system of equations

$$U_0 = u^0, \quad U_n = \varphi_{\Delta T_{n-1}}(U_{n-1}), \quad n = 1, \ldots, N-1, \tag{3}$$

where $\varphi_{\Delta T_n}(U)$ denotes the solution of (1) with initial condition U after time ΔT_n. This time decomposition method is nothing else than a multiple shooting method for (1), see [4]. Letting $U = (U_0^T, \ldots U_{N-1}^T)^T$, the system (3) can be written in the form

$$F(U) = \begin{pmatrix} U_0 - u^0 \\ U_1 - \varphi_{\Delta T_0}(U_0) \\ \vdots \\ U_{N-1} - \varphi_{\Delta T_{N-2}}(U_{N-2}) \end{pmatrix} = 0, \tag{4}$$

where $F : \mathbb{R}^{M \cdot N} \longrightarrow \mathbb{R}^{M \cdot N}$. System (4) defines the unknown initial values U_n for each time subdomain, and needs to be solved, in general, by an iterative method. For a direct method in the case where (1) is linear and the system (4) can be formed explicitly, see [1].

Applying Newtons method to (4) leads after a short calculation to

$$\begin{aligned} U_0^{k+1} &= u^0, \\ U_n^{k+1} &= \varphi_{\Delta T_{n-1}}(U_{n-1}^k) + \varphi'_{\Delta T_{n-1}}(U_{n-1}^k)(U_{n-1}^{k+1} - U_{n-1}^k), \end{aligned} \tag{5}$$

where $n = 1, \ldots, N-1$. In [4], it was shown that the method (5) converges quadratically, once the approximations are close enough to the solution. However in general, it is too expensive to compute the Jacobian terms in (5) exactly. An interesting recent approximation is the parareal algorithm, which uses two approximations with different accuracy: let $F(T_n, T_{n-1}, U_{n-1})$ be an accurate approximation to the solution $\varphi_{\Delta T_{n-1}}(U_{n-1})$ on time subdomain Ω_{n-1}, and let $G(T_n, T_{n-1}, U_{n-1})$ be a less accurate approximation, for example on a coarser grid, or a lower order method, or even an approximation using a simpler model than (1). Then, approximating the time subdomain solves in (5) by $\varphi_{\Delta T_{n-1}}(U_{n-1}^k) \approx F(T_n, T_{n-1}, U_{n-1}^k)$, and the Jacobian term by

$$\varphi'_{\Delta T_{n-1}}(U_{n-1}^k)(U_{n-1}^{k+1} - U_{n-1}^k) \approx G(T_n, T_{n-1}, U_{n-1}^{k+1}) - G(T_n, T_{n-1}, U_{n-1}^k),$$

we obtain as approximation to (5)

$$\begin{aligned} U_0^{k+1} &= u^0, \\ U_n^{k+1} &= F(T_n, T_{n-1}, U_{n-1}^k) + G(T_n, T_{n-1}, U_{n-1}^{k+1}) - G(T_n, T_{n-1}, U_{n-1}^k), \end{aligned} \tag{6}$$

which is the parareal algorithm, see [15] for a linear model problem, and [2] for the formulation (6). A natural initial guess is the coarse solution, i.e. $U_n^0 = G(T_n, T_{n-1}, U_{n-1}^0)$ for $n = 1, 2, \ldots, N$.

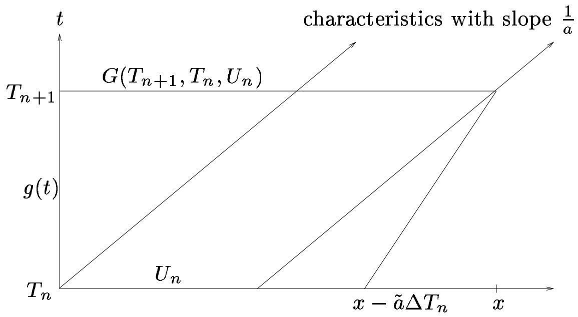

Figure 1: Propagation of the solution along the characteristics, and Assumption on the approximate solver G.

3 The Parareal Algorithm for the Advection Equation

We now study the convergence behavior of the parareal algorithm applied to the one dimensional linear advection equation on the domain $\Omega := (0, L)$, $L > 0$,

$$\begin{array}{rcll} u_t + a u_x & = & f & \text{in } \Omega \times (0, T), \\ u(x, 0) & = & u^0(x) & \text{in } \Omega, \\ u(0, t) & = & g(t) & t \in (0, T), \end{array} \tag{7}$$

where we assume that $a > 0$, so that a boundary condition on the left needs to be imposed. We also assume for simplicity that a is constant; we will indicate at the end how the analysis can be generalized to the case of variable advection speed a.

Applying the parareal algorithm to (7), we obtain the same iteration (6), where now however the iterates are functions, $U_n^k : \Omega \longrightarrow \mathbb{R}$. Numerical experiments in [11] showed that the parareal algorithm converges approximately linearly for this problem, which could not be explained by the Fourier analysis in [11]. We assume in what follows that the fine approximation $F(T_n, T_{n-1}, U_{n-1})$ is the exact solution of (7) at time T_n with initial condition U_{n-1} at time T_{n-1}, and that $G(T_n, T_{n-1}, U_{n-1})$ is an approximate solution of (7) at time T_n with initial condition U_{n-1} at time T_{n-1}. In addition, G needs to satisfy the assumption

Assumption 1 *There exists a positive constant $\tilde{a}$, $0 < \tilde{a} \leq a$, such that $G(T_{n+1}, T_n, U_n)$ at x only depends on the boundary condition g and on $U_n(x - \tilde{a}\Delta T_n)$ for $x - \tilde{a}\Delta T_n \geq 0$.*

Remark 1 *Assumption 1 is natural for the advection equation, since the exact solution follows the characteristics, as illustrated in Figure 1, and for convergence, the CFL condition of the scheme requires that $\tilde{a} \leq a$.*

We need two lemmas to prove a convergence result of the parareal algorithm (6) applied to problem (7). The first Lemma holds in general for the parareal algorithm (6) applied to any problem.

Lemma 1 *If F is exact, then at iteration k of the parareal algorithm (6), U_n^k is the exact solution for $n \leq k$.*

Demostración. The proof is by induction, in both k and n: for $k = 0$, we have $U_0^0 = u^0$ which is the initial condition and hence is exact. So assume that the result holds for k, i.e. U_n^k is exact for $n \leq k$. Then at iteration $k+1$, we still have for $n = 0$ that $U_0^{k+1} = u^0$, and we can now use induction on n: assuming that U_n^{k+1} is exact, algorithm (6) gives, since $U_n^{k+1} = U_n^k$,

$$\begin{aligned} U_{n+1}^{k+1} &= F(T_{n+1}, T_n, U_n^k) + G(T_{n+1}, T_n, U_n^{k+1}) - G(T_{n+1}, T_n, U_n^k) \\ &= F(T_{n+1}, T_n, U_n^k), \end{aligned}$$

which is exact by the assumption on F, and thus concludes the proof. $\square$

The next lemma shows a similar property going out from the left boundary, if the parareal algorithm (6) is applied to the advection equation (7).

Lemma 2 *Let F be exact and G satisfy Assumption 1, when the parareal algorithm (6) is applied to the advection equation (7). If $U_n^k(x)$ at iteration k satisfies $U_n^k(x) = u(x, T_n)$, for $x \in [0, \alpha]$ for some $\alpha \geq 0$ and for all $n = 0, 1, \ldots, N$, then $U_n^{k+1}(x) = u(x, T_{n+1})$ for $x \in [0, \alpha + \tilde{a}\Delta T]$ and all n, where $\Delta T = \min_{n\in\{0,1,\ldots,N-1\}} \Delta T_n$.*

Demostración. The proof is by induction on n. For $n = 0$, we have by definition in (6) that $U_0^{k+1} = u(x, 0)$ for all x and all k. So now we assume that for a given n, $U_n^{k+1}(x) = u(x, T_n)$ for $x \in [0, \alpha + \tilde{a}\Delta T]$. By assumption of the Lemma, we also have that $U_n^k(x) = u(x, T_n)$ for $x \in [0, \alpha]$, which implies that

$$U_n^{k+1}(x) = U_n^k(x) = u(x, T_n), \quad \text{for } x \in [0, \alpha].$$

Using now Assumption 1, the difference $G(T_{n+1}, T_n, U_n^{k+1}) - G(T_{n+1}, T_n, U_n^k)$ in algorithm (6) vanishes on the interval $[0, \alpha + \tilde{a}\Delta T_n]$, and thus the algorithm (6) gives on this interval $U_{n+1}^{k+1} = F(T_{n+1}, T_n, U_n^k)$, which implies that $U_{n+1}^{k+1} = u(x, T_{n+1})$ for $x \in [0, \alpha + \tilde{a}\Delta T_n]$, since F is the exact solution, $U_n^k(x) = u(x, T_n)$ for $x \in [0, \alpha]$ and $a \geq \tilde{a}$. Using now that $\Delta T = \min_{n\in\{0,1,\ldots,N-1\}} \Delta T_n$ completes the proof by induction. $\square$

We are now ready to prove a convergence estimate for the parareal algorithm (6) applied to the advection equation (7). Figure 2 is illustrating the argument graphically, where we assumed that the coarse grid is equi-spaced, i.e. $T_n - T_{n-1} = \Delta T$ for all $n = 1, 2, \ldots, N$.

Theorem 1 *Let U_n^k be the approximations computed by the parareal algorithm (6) applied to the advection equation (7), where F is exact and G satisfies Assumption 1, and let $\Delta T = \min_{n\in\{0,1,\ldots,N-1\}} \Delta T_n$. Then we have the convergence estimate*

$$\sum_{n=0}^{N} ||u(\cdot, T_n) - U_n^k||_1 \leq C \max(L - k\tilde{a}\Delta T, 0) \times \max(N - k, 0), \tag{8}$$

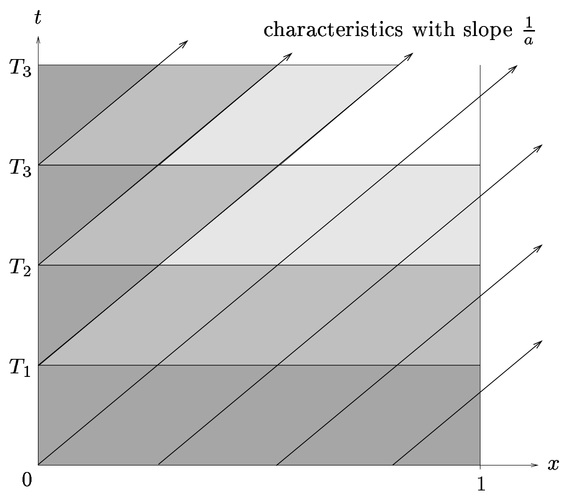

Figure 2: Convergence mechanism of the parareal algorithm applied to the advection equation.

where the constant C can be estimated by

$$C = \max_{n=1,2,\ldots,N} ||u(\cdot, T_n) - U_n^0||_\infty.$$

Demostracio´n. For $k = 0$, the estimate holds, since the sum of the L^1 norms is bounded by the maximum of the L^∞ norms multiplied by L and N. To obtain the decay estimate (8) for $k > 0$, Lemma 1 shows that $||u(\cdot, T_n) - U_n^k||_1 = 0$ for $n \leq k$, and using Lemma 2 inductively in k shows that $U_n^k(x) = u(x, T_n)$ for $x \in [0, k\tilde{a}\Delta T]$. Therefore, the difference $u(x, T_n) - U_n^k(x)$ is only non-zero for $n > k$ and $x > k\tilde{a}\Delta T$, which leads to the estimate (8). □

Remark 2 *The convergence result stated in Theorem 1 for the case of a constant advection term can be generalized to the case of variable advection, one simply needs to estimate the minimal distance over which the solution is transported. As long as this quantity remains positive, a similar convergence estimate holds.*

4 Numerical Experiments

We solve the advection equation (7) with $a = 1$ on the domain $\Omega = (0, L)$ with $L = 1$, and in time from zero up to $T = 2$, using for the initial condition $u^0(x) = e^{-100(x-\frac{L}{2})^2}$, for the boundary condition $g(t) = \operatorname{sen} 5t$, and for the source function $f(x, t) = 0$. We discretize the equation using the simple first order upwind scheme

$$\frac{u_{i+1}^j - u_i^j}{\Delta t} + a\frac{u_i^j - u_i^{j-1}}{\Delta x} = f_i^j,$$

with fine spatial and temporal discretization steps Δx and Δt to emulate F, and with coarser spatial and temporal discretization steps ΔX and ΔT to obtain the coarse approximation G. Note that we need to interpolate the solution from the coarse spatial grid to the fine spatial grid, and we use here linear interpolation.

In the first experiment, we chose for the coarse mesh $\Delta T = \frac{T}{12}$ and $\Delta X = \frac{1}{6}$, and for the fine mesh $\Delta t = \frac{T}{240}$ and $\Delta x = \frac{1}{120}$. We show in Figure 3 the initial guess and the first five iterations of the parareal algorithm. One can clearly see how the error is removed step by step, both from the initial line and also from the left boundary, as predicted by Theorem 8. We show in Figure 4 the convergence curve corresponding to this experiment, together with the theoretical estimate from Theorem 1. This shows that the rate estimate is quite sharp, the only overestimate is in this example the constant due to the use of the L^∞ and maximum norms in (8), but one could construct an example where this estimate is sharp as well.

We used in this first example a spatial and temporal discretization step which is precisely at the CFL condition, $a\frac{\Delta T}{\Delta X} = 1$, and thus Assumption 1 is verified with $\tilde{a} = a = 1$. In the next example, we change the coarse time step to $\Delta T = \frac{T}{13}$ and the fine time step to $\Delta t = \frac{T}{260}$, so that the discretization now stays below the CFL condition, $a\frac{\Delta T}{\Delta X} \approx 0.923$. In this case, Assumption 1 is not verified any more with a strictly positive $\tilde{a}$ for our discretization, since at each grid point, the scheme uses information from the same grid point one step earlier in time. We show in Figure 5 again the initial guess and the first five iterates. Even though our analysis does not apply any more, the behavior of the parareal algorithm is very similar to the case where Assumption 1 is verified: the error is still removed from the boundary as well, but now only approximately, as one can see in the error plots: on the left, at iteration two, and even more pronounced at iteration 3, the error is not identically zero any more in the corresponding spatial interval, it takes one more iteration to remove it there, as one can see in iteration 4. We show in Figure 6 the convergence curve for this case. Clearly the algorithm does not converge any more in the sixth step, but a more rapid convergence regime sets in, probably due to the slight diffusive nature of the discretized problem [11].

5 The Parareal Algorithm for the Wave Equation

It is tempting to try to generalize the convergence analysis using characteristics to the case of the second order wave equation,

$$\begin{array}{rcll} u_{tt} & = & c^2 u_{xx} & \text{in } \Omega \times (0,T), \\ u(x,0) & = & u^0(x) & \text{in } \Omega. \end{array} \qquad (9)$$

The property of the advection equation (7) which led to the convergence result in Theorem 1 was the fact that the solution later in time is only affected by the boundary condition, and not the initial condition. This is however not the case

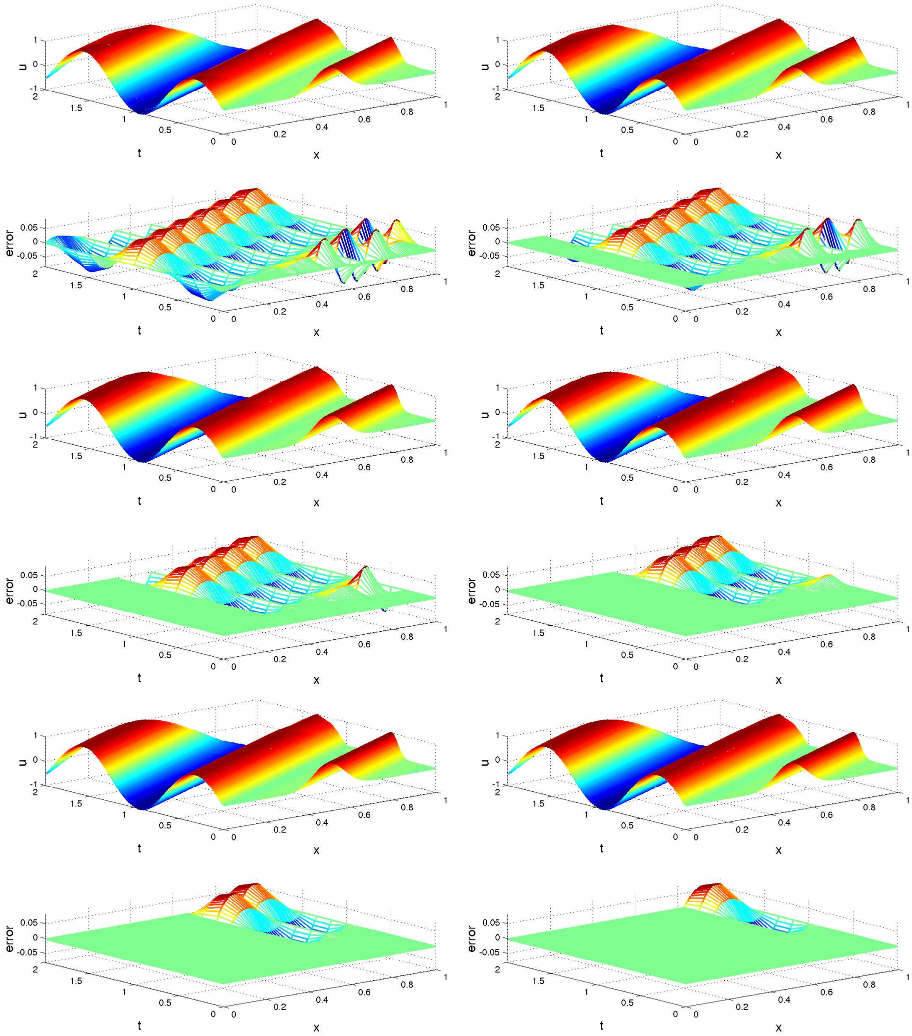

Figure 3: Initial guess and first five iterates of the parareal algorithm, on top the approximate solution, and underneath each time the error of the approximation, for the case where the discretization is precisely at the CFL condition.

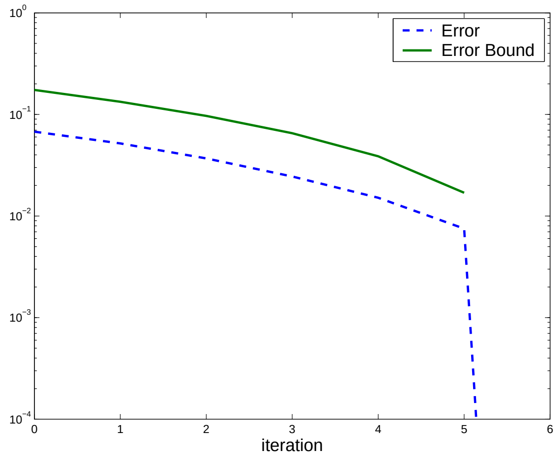

Figure 4: Convergence behavior of the parareal algorithm applied to the advection equation, together with the theoretical estimate.

for the wave equation (9) with Dirichlet boundary conditions

$$\begin{array}{rcll} u(0,t) & = & g_l(t) & t \in (0,T), \\ u(L,t) & = & g_r(t) & t \in (0,T), \end{array} \tag{10}$$

since with these conditions, the solution will be reflected on the boundary, and hence the initial condition can have an influence on the solution at an arbitrary later time.

The situation changes when one imposes transparent boundary conditions,

$$\begin{array}{rcll} cu_x(0,t) - u_t(0,t) & = & g_l(t) & t \in (0,T), \\ cu_x(L,t) + u_t(L,t) & = & g_r(t) & t \in (0,T). \end{array} \tag{11}$$

With these conditions, waves that arrive at the boundary are simply absorbed, and the only incoming information comes from the boundary functions $g_l(t)$ and $g_r(t)$. We show a numerical solution as an example in Figure 7, where we create an initial wave at $x = 1$, and also on each boundary a wave at $t = \frac{1}{2}$. A convergence result similar to the one for the advection equation could be shown for this case, as illustrated in Figure 8. There are now two propagation directions, so the algorithm can transport the correct boundary information both from the left and the right boundary, in addition to the initial line. As soon as in a region both the correct information from the left and from the right are available, the algorithm obtains the exact solution in that region. In order to prove such a result however, one will need a similar assumption on G as Assumption 1 for the advection equation, and unlike in the advection case, it is not natural for a discretization of the wave equation to solve the two propagation directions independently. For example the standard second order centered finite difference scheme for (9) is always using information from both

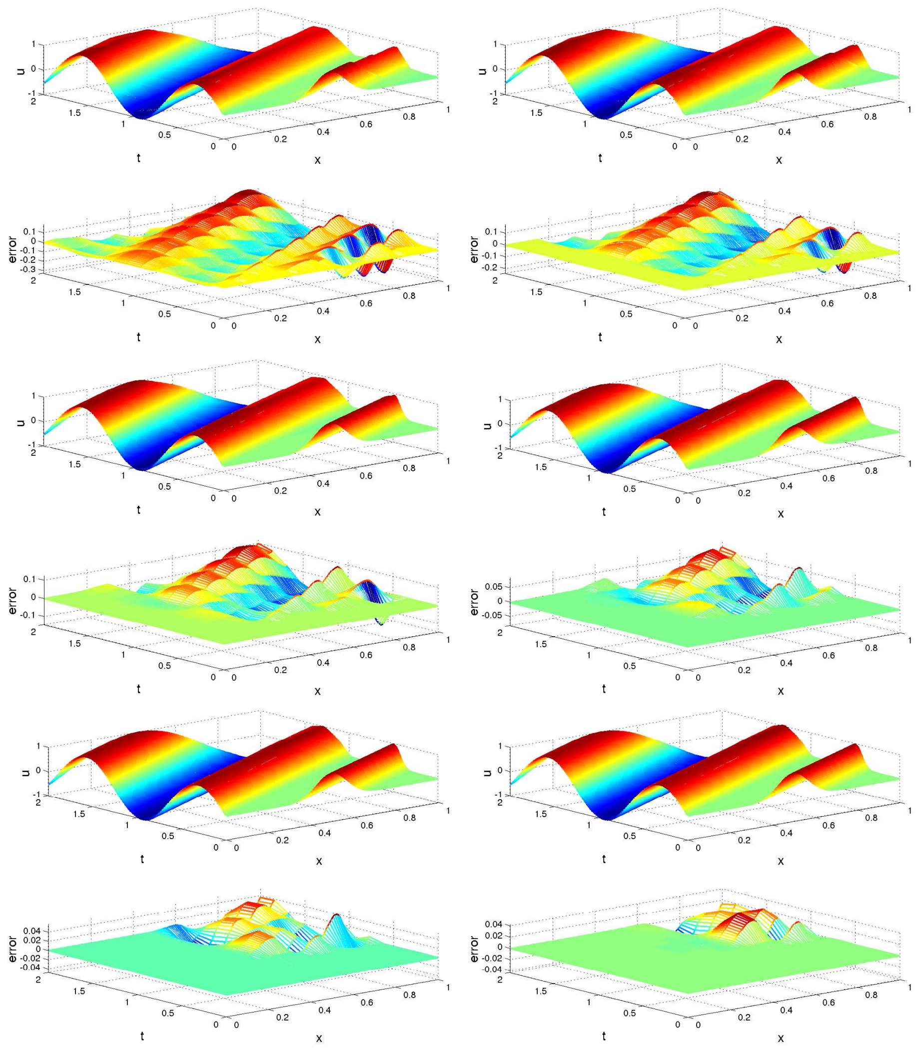

Figure 5: Initial guess and first five iterates of the parareal algorithm, on top the approximate solution, and underneath each time the error of the approximation, for the case where the discretization is below the CFL condition.

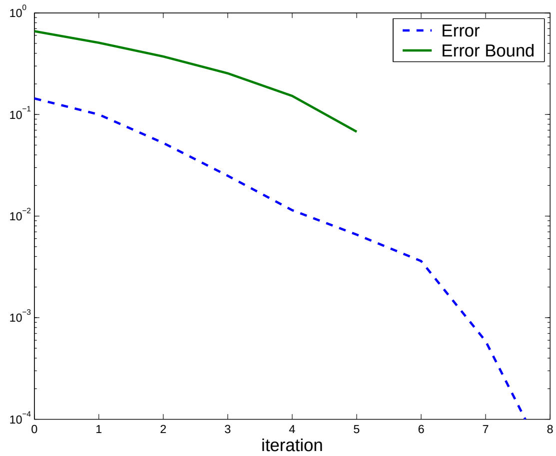

Figure 6: Convergence behavior of the parareal algorithm applied to the advection equation, together with the theoretical estimate, when Assumption 1 is violated.

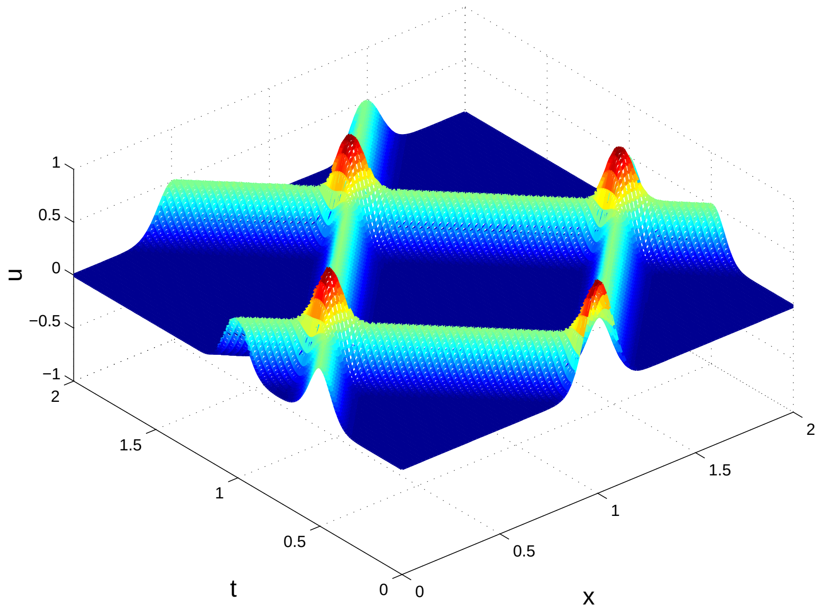

Figure 7: An example of a solution of the wave equation with transparent boundary conditions.

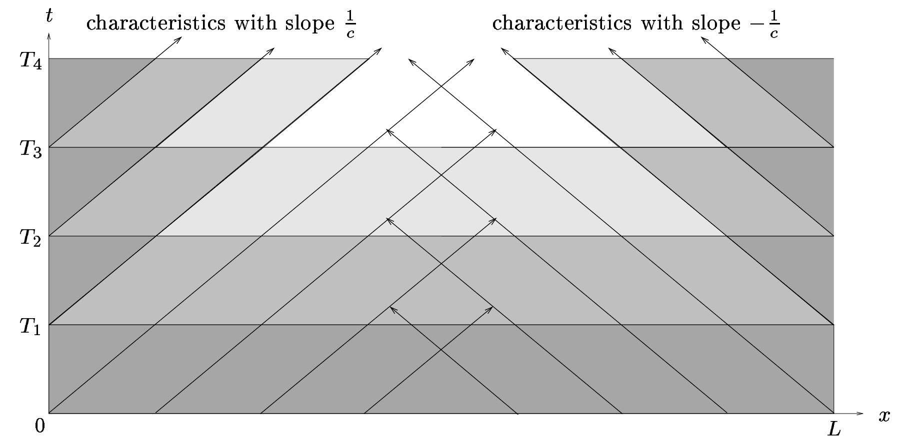

Figure 8: Idea how to generalize the convergence argument for the advection equation using characteristics to the wave equation.

directions, which prevents a convergence argument from going through. An illustrative example is shown in Figure 9. While the algorithm clearly proceeds to obtain the solution from the initial line in the time direction, as proved in Lemma 1, the solution at the two spatial boundaries is not obtained as in the case of the advection equation, only once the correct front from the initial condition reaches the point where the boundary condition is non-zero. Thus a convergence result like Theorem 1 does not hold when the parareal algorithm (6) is applied to an arbitrary discretization of the second order wave equation (9), not even with transparent boundary conditions (11).

6 Conclusions

Using characteristics, we have obtained a convergence result for the parareal algorithm applied to the advection equation with Dirichlet boundary condition. The algorithm computes in that case exact solution parts from the boundary inward, as it does usually from the initial line. This result can be generalized for variable coefficient advection problems, the only property needed in the proof is the transport of information along characteristics. Our analysis indicates however that the parareal algorithm is not the ideal tool to parallelize the solution of the advection equation: it would be much more efficient to solve such problems along characteristics, and then each characteristic can be solved independently, the problem becomes embarrassingly parallel.

In the case of the second order wave equation, the parareal algorithm can not obtain the exact solution from the boundary inward, even though the solution also has propagation directions, as in the case of the advection equation. For the algorithm to successfully do so, it would need a discretization which satisfies an assumption similar to the key assumption made for the advection equation, and

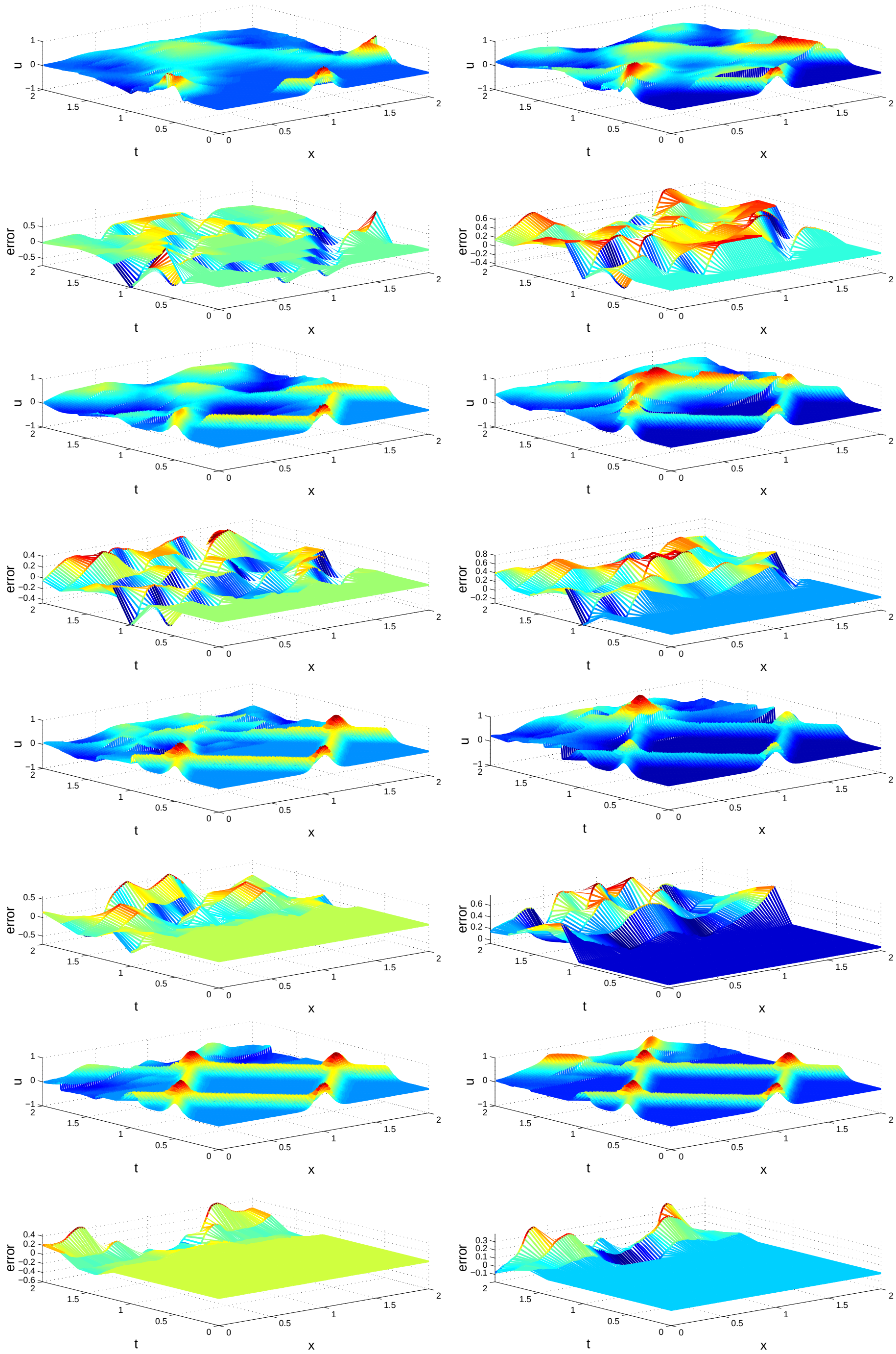

Figure 9: Initial guess and first seven iterates of the parareal algorithm, on top the approximate solution, and underneath each time the error of the approximation, for the case of the second order wave equation.

usual discretizations of the wave equation do not satisfy such an assumption. The algorithm is thus only converging from the initial line, which makes it not very useful for parallelizing the solution of the wave equation in time, since the number of iterations needed is then equal to the number of processors one can use in time.

If one needs to compute approximate solutions of the second order wave equation in a time parallel fashion, one therefore either needs to find discretizations of the wave equation which satisfy a propagation assumption, like in the case of the advection equation, or one needs to use a modified time parallel algorithm, a possibility being the modification of the parareal algorithm proposed in [5, 7, 10].

References

[1] P. Amodio and L. Brugnano. Parallel implementation of block boundary value methods for ODEs. *J. Comp. Appl. Math.*, 78:197–211, 1997.

[2] L. Baffico, S. Bernard, Y. Maday, G. Turinici, and G. Zérah. Parallel-in-time molecular-dynamics simulations. *Physical Review E*, 66:057706–1–4, 2002.

[3] G. Bal. On the convergence and the stability of the parareal algorithm to solve partial differential equations. In R. Kornhuber, R. H. W. Hoppe, J. Périaux, O. Pironneau, O. B. Widlund, and J. Xu, editors, *Proceedings of the 15th international domain decomposition conference*, pages 426–432. Springer LNCSE, 2003.

[4] P. Chartier and B. Philippe. A parallel shooting technique for solving dissipative ODEs. *Computing*, 51:209–236, 1993.

[5] J. Cortial and C. Farhat. A time-parallel implicit methodology for the near-real-time solution of systems of linear oscillators. In L. Biegler, O. Ghattaas, M. Heinkenschloss, D. Keyes, and B. van Bloemen Waanders, editors, *Real-Time PDE-Constrained Optimization*. SIAM, 2006.

[6] C. Farhat and M. Chandesris. Time-decomposed parallel time-integrators: theory and feasibility studies for fluid, structure, and fluid-structure applications. *Internat. J. Numer. Methods Engrg.*, 58(9):1397–1434, 2003.

[7] 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(5):697–724, 2006.

[8] P. F. Fischer, F. Hecht, and Y. Maday. A parareal in time semi-implicit approximation of the Navier-Stokes equations. In R. Kornhuber, R. H. W. Hoppe, J. Périaux, O. Pironneau, O. B. Widlund, and J. Xu, editors, *Proceedings of the 15th international domain decomposition conference*, pages 433–440. Springer LNCSE, 2003.

[9] M. J. Gander and E. Hairer. Nonlinear convergence analysis for the parareal algorithm. In U. Langer, O. Widlund, and D. Keyes, editors, *Proceedings of the 17th international domain decomposition conference*. Springer LNCSE, 2007.

[10] M. J. Gander and M. Petcu. Analysis of a modified parareal algorithm for second-order ordinary differential equations. In *AIP Conference Proceedings of the International Conference of Numerical Analysis and Applied Mathematics ICNAAM*, volume 936, pages 233–236, 2007.

[11] M. J. Gander and S. Vandewalle. Analysis of the parareal time-parallel time-integration method. *SIAM J. Sci. Comp.*, 29(2):556–578, 2007.

[12] I. Garrido, M. S. Espedal, and G. E. Fladmark. A convergence algorithm for time parallelization applied to reservoir simulation. In R. Kornhuber, R. H. W. Hoppe, J. Périaux, O. Pironneau, O. B. Widlund, and J. Xu, editors, *Proceedings of the 15th international domain decomposition conference*, pages 469–476. Springer LNCSE, 2003.

[13] I. Garrido, B. Lee, G. E. Fladmark, and M. E. Espedal. Convergent iterative schemes for time parallelization. *Mathematics of Computation*, 75:1403–1428, 2006.

[14] 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. on CAD of IC and Syst.*, 1:131–145, 1982.

[15] J.-L. Lions, Y. Maday, and G. Turinici. A parareal in time discretization of pde's. *C.R. Acad. Sci. Paris, Serie I*, 332:661–668, 2001.

[16] Y. Maday and G. Turinici. A parareal in time procedure for the control of partial differential equations. *C.R.A.S. Sér. I Math*, 335:387–391, 2002.

[17] Y. Maday and G. Turinici. The parareal in time iterative solver: a further direction to parallel implementation. In R. Kornhuber, R. H. W. Hoppe, J. Périaux, O. Pironneau, O. B. Widlund, and J. Xu, editors, *Proceedings of the 15th international domain decomposition conference*, pages 441–448. Springer LNCSE, 2003.

[18] W. L. Miranker and W. Liniger. Parallel methods for the numerical integration of ordinary differential equations. *Math. Comp.*, 91:303–320, 1967.

[19] J. Nievergelt. Parallel methods for integrating ordinary differential equations. *Comm. ACM*, 7:731–733, 1964.

[20] G. A. Staff and E. M. Rønquist. Stability of the parareal algorithm. In R. Kornhuber, R. H. W. Hoppe, J. Périaux, O. Pironneau, O. B. Widlund, and J. Xu, editors, *Proceedings of the 15th international domain decomposition conference*, pages 449–456. Springer LNCSE, 2003.