An Adaptive Parallel-in-Time Method with application to a membrane problem

Noha Makhoul Karam[1], Nabil Nassif[2], and Jocelyne Erhel[3]

1 Introduction

Assuming global existence on $[0,\infty)$ (and uniqueness) for a solution to the initial value problem:

$$(\mathscr{S}) \qquad \begin{cases} \frac{dY}{dt} = F(t,Y), & 0 < t \leq T < \infty, \\ Y(0) = Y_0, \end{cases}$$

we seek in this paper, computing its solution $Y : [0,\infty] \to \mathbb{R}^k$ using a parallel-in-time method, for a given function $F : \mathbb{R} \times \mathbb{R}^k \to \mathbb{R}^k$.

There is no natural parallelism across time since the solution on a time level must be known before the computation of the solution at subsequent time levels can start. However, it could be possible to compute simultaneously on many time levels by providing a multi-processor architecture some initial guesses for the solution at later time levels. Such time-parallel computations may be superposed to parallelism in space variables whenever $(\mathscr{S})$ results from a semi-discretization of a time-dependent partial differential equation. Several parallel-in-time algorithms have been proposed to tackle $(\mathscr{S})$. One of the first has been suggested by Nievergelt [12] in 1964 and led to *multiple shooting methods* of which many variants were developed [2], [3], ... In the eighties and nineties, parabolic multigrid methods and multigrid waveform relaxation have been devised. In 2001, Lions, Maday & Turinici proposed in [5] the *parareal algorithm* that marked a turning point: since its introduction, it was subject to many contributions ([6], [4], [1], ...), in particular during Domain Decomposition Conferences. All those methods are based on the principle of combining coarse and fine resolutions in time, starting with the choice of a most often *regular coarse grid* for the time domain, followed by prediction of starting seed values at the lower ends of the coarse grid intervals, then iteratively proceed with parallel computations on a fine grid within each time-interval yielding updated values at their upper ends. Evaluation of the resulting gaps between predicted and updated values on the coarse grid provides corrections for new seed values. An iterative process is thus pursued until convergence occurs.

In this work, we give a parallel-in-time method that has been first introduced in [10] and experimented on a reaction-diffusion problem having a bounded solution. Two main features are used in this method: (i) the use of an end-of-slice function, strongly related to the behavior of the solution, that permits the automatic generation of a non-uniform coarse grid; (ii) rescaling, within each of the generated slices, the time and the solution variables thus obtaining a sequence of rescaled initial value problems with uniformity properties. Such approach has been used (in its two components) in [8] and [11] for getting sequentially accurate solutions for stiff and ex-

Université Saint Joseph, Beyrouth noha.makhoulkaram@usj.edu.lb · American University of Beirut nn12@aub.edu.lb · INRIA, Rennes Jocelyne.Erhel@inria.fr

plosive systems and has been exploited in [10] for parallel time integration of several types of initial value problems. The resulting parallel in time integration is done without numerical integration over the coarse grid as it is the case in the parareal method: instead, a concept of similarity between the rescaled systems allows the prediction of starting values at the onset of future slices. We refine here the similarity concepts in order to tackle more problems (having non-bounded solutions) and to increase the accuracy of the predictions thus enhancing speed-ups.
After giving, in section 2, an overview of the automatic coarse grid generation, we define in section 3 some similarity properties for the rescaled systems. This yields a prediction model which is at the core of the adaptive parallel-in-time (APTI) algorithm presented in section 4. Numerical results on a membrane problem are then given in section 5.

2 Automatic Coarse Grid generation

The basic principle of the method is in breaking $(\mathscr{S})$ into a sequence of **shooting values problems**. Specifically, we assume the existence of a shooting-value function $E:\mathbb{R}^k\times\mathbb{R}^k\to\mathbb{R}$ that permits the initiation of a recurrence process, starting with a **first slice** of the coarse grid, obtained by seeking $\{T_1,\{Y(t)\in\mathbb{R}^k, 0\le t\le T_1\}\}$ such that:

$$(\mathscr{S}_1)\qquad \begin{cases} \frac{dY}{dt}=F(t,Y),\ 0<t<T_1,\\ Y(0)=Y_0,\\ E(Y(t),Y_0)\neq 0,\, 0<t<T_1,\ E(Y(T_1),Y_0)=0. \end{cases}$$

$Y_1=Y(T_1)$ becomes the initial condition for a 2^{nd} slice of the coarse grid. More generally, we let for $n>1$, $Y_{n-1}=Y(T_{n-1})$ and define the system on the n^{th} slice:

$$(\mathscr{S}_n)\qquad \begin{cases} \frac{dY}{dt}=F(t,Y),\ T_{n-1}<t<T_n,\\ Y(T_{n-1})=Y_{n-1},\\ E(Y(t),Y_{n-1})\neq 0,\, T_{n-1}<t<T_n,\ E(Y(T_n),Y_{n-1})=0. \end{cases}$$

Based on the *End-Of-Slice (EOS) function* $E(.,.)$, one gets **the coarse grid**:

$$\{0=T_0<T_1<...<T_n<...T_{N-1}<T\le T_N\},$$

with the corresponding sequence of starting values of the solution:

$$\{Y_n=Y(T_n)|n=0,1,...,N\}.$$

Two cases of existence of a function $E(.,.)$ have so far been identified ([7]).

a. Case of Explosive solutions
Let $||.||=||.||_{\infty,\mathbb{R}^k}$ and assume $\lim_{\mathbf{t}\to\infty}\|\mathbf{Y}(\mathbf{t})\|=\infty$. In that case, given $U,W\in\mathbb{R}^k$, and $D(W)\in\mathbb{R}^{k\times k}$ an invertible matrix depending on W with $||(D^{-1}(W))(V)||\ge c(W)||V||$, we then let for $S>0$: $E(U,W)=||D^{-1}(W)(U-W)||-S$.
When applied to $(\mathscr{S}_n)$, such function $E(.,.)$ determines the size of the n^{th} slice $[T_{n-1},T_n]$ by:

$$-S\le E(Y(t),Y_{n-1})<0,\, T_{n-1}\le t<T_n \text{ and } E(Y(T_n),Y_{n-1})=0. \qquad (1)$$

b. Case of Oscillatory Problems
When the behavior of the solution is oscillatory, over a long period of time, in the sense that there exists a two-dimensional plane $\mathscr{P}$ in $\mathbb{R}^k$ on which the projection of the solution's trajectory rotates about a fixed center ω, then a slice is ended when the solution completes a full, or almost full, rotation in that plane about ω.

3 Parallelizing the shooting values problems $\{\mathscr{S}_n\}$

The sequence of shooting values problems $\{(\mathscr{S}_n)|n = 1,...,N\}$ can be computed in a parallel way, provided one is able to predict *accurately*, the coarse grid $\{0,T_1,T_2,...,T_N\}$ and the values of the solution $Y(t)$ on that grid, i.e. $\{Y_0,Y_1,Y_2,...,Y_N\}$.

Rescaling and use of local time and solution:
Dealing uniformly with $\{(\mathscr{S}_n)\}$ is then done through a rescaling technique that changes the variables $\{t,Y(t)\}$ on each time-slice $[T_{n-1},T_n]$, into a new pair $\{s,Z_n(s)\}$:

$$\begin{array}{ll} t = T_{n-1} + \beta(Y_{n-1})s, & (2.1) \\ Y(t) = Y_{n-1} + D(Y_{n-1})Z_n(s). & (2.2) \end{array} \qquad (2)$$

where $\beta(Y_{n-1}) \equiv \beta_n > 0$ and $D(Y_{n-1}) \equiv D_n \in \mathbb{R}^{k\times k}$ is an invertible matrix. Thus, each $(\mathscr{S}_n)$ is now equivalent to a shooting value problem, whereby one seeks the pair $\{s_n, \{Z_n(s) \in \mathbb{R}^k, 0 \le s \le s_n\}\}$, such that:

$$(\mathscr{S}'_n) \qquad \begin{cases} \frac{dZ_n}{ds} = G_n(s,Z_n),\ 0 < s < s_n \\ Z_n(0) = 0, \\ H_n(Z_n(s)) \neq 0,\ 0 < s < s_n\ H_n(Z_n(s_n)) = 0, \end{cases}$$

where:
$G_n(s,Z_n) = \beta_n D_n^{-1} F(T_{n-1} + \beta_n s, Y_{n-1} + D_n Z_n)$ and $H_n(Z_n) = E(Y_{n-1} + D_n Z_n, Y_{n-1})$.
Note the following:
- The rescaled time $s = \frac{t-T_{n-1}}{\beta_n}$ and solution $Z_n(s)$ are *set to* 0 *at the beginning of every slice*.
- The functions G_n and H_n depend on the starting values T_{n-1} and Y_{n-1}.
- The solution function $Z_n(.)$ depends on β_n, on each n^{th} slice, in the sense that *different choices of β_n lead to different functions $Z_n(.)$*. However, *independently of β_n and D_n*, one has the following identities:

$$\forall \beta_n, \qquad \begin{cases} \beta_n s_n = T_n - T_{n-1}, & (3.1) \\ Z_n(s_n) = D_n^{-1}(Y_n - Y_{n-1}). & (3.2) \end{cases} \qquad (3)$$

These identities are at the core of our prediction model, whereas, if the choice of $\beta(Y_{n-1})$ and $D(Y_{n-1})$ are such that the behavior of the pair $\{s_n, Z_n(s_n)\}$ can be accurately predicted, then the coarse grid $\{T_n\}$ and the values $\{Y_n\}$ of $Y(t)$ on that grid can also be obtained from (3).

Similarity concepts:
The change of variables (2) and the consequent rescaled problems $(\mathscr{S}'_n)$ have been originally proposed in [8] and [11] to handle initial value problems $(\mathscr{S})$ which solutions explode in a finite time. As the computation of these problems present a high sensitivity to the sharp variations of the solution on a short time, one way to circumvent this issue is through appropriate choices of $\{\beta_n, D_n, H_n(.)\}$, so that one inherits "uniformity" on the rescaled systems $\{(\mathscr{S}'_n)\}$. This is done by selecting appropriately the rescaling parameter β_n so as to insure uniform boundedness, independently of n, of $\{s_n\}$, $\|Z_n\|$, $\|G_n\|$ and $\|J_{G_n}\|$ (where J_{G_n} is the jacobian of G_n), thus controlling the stiffness of the problem. In that way, placing the same fine solver on each of the $(\mathscr{S}'_n)$, provides a robust algorithm for solving $(\mathscr{S})$, as proved in [9].
Using this approach for parallel in time solving was done first in [10] and more extensively in [7] on the basis of properties satisfied by the pair $\{s_n, Z_n(s_n)\}$.

Definition 1. Invariance: If the rescaling parameters $\{\beta_n, D_n\}$ are such that $\forall n,\ \ G_n(.,.) = G_1(.,.)$ and $H_n(.) = H_1(.)$, then the rescaled systems $(\mathscr{S}'_n)$ are **invariant** and are all equivalent to the shooting Problem $(\mathscr{S}'_1)$.
In that case one has $\forall n$, $Z_n(.) = Z_1(.)$, $s_n = s_1$ and $Z_n(s_n) = Z_1(s_1)$. Invariance is an ideal and rare case: one unique time-slice allows getting the solution on all time-slices through a simple change of variables. A weaker property is given as follows.

Definition 2. Asymptotic similarity: it occurs when the rescaling parameters $\{\beta_n, D_n\}$ are such that $\lim_{n\to\infty}\{s_n, Z_n(s_n)\} = \{s_L, Z_L(s_L)\}$, where $\{s_L, Z_L(s_L)\}$ are obtained from a limit shooting value problem:

$$(\mathscr{S}_L) \qquad \begin{cases} \frac{dZ_L}{ds} = G_L(s, Z_L),\ 0 < s < s_L \\ Z_L(0) = 0, \\ H_L(Z_L(s)) \neq 0,\ 0 < s < s_L\ H_L(Z_L(s_L)) = 0. \end{cases}$$

In this case, the use of (3) for a prediction purpose is possible after a sequential run on a number of slices n_s, at which point one has:

$$\max_{n > n_s}\{\max\{|s_n - s_{n-1}|, ||Z_n(s_n) - Z_{n-1}(s_{n-1})||\}\} \leq tol, \tag{4}$$

where tol is a user's computation tolerance. We finally consider, based on (4), a weak case of similarity, which can be used in spite of the lack of any evidence of invariance or asymptotic similarity.

Definition 3. Numerical Similarity is considered to be reached, whenever, there exists 2 integers, $n_0 \geq 1$ and n_r sufficiently large, such that:

$$\max_{n_0 \leq n \leq n_0 + n_r}\{\max\{|s_n - s_{n-1}|, ||Z_n(s_n) - Z_{n-1}(s_{n-1})||\}\} \leq tol, \tag{5}$$

In that case, as in (4), one lets $n_s = n_0 + n_r$.

Remark: in the case where all components of Y_n are *distinct from 0*, then (3.2) is equivalent to $Y_n = D_n(e + Z_n(s_n))$, where $e \in \mathbb{R}^k$ is a vector of 1's, and $Z_n(s_n) = D_n^{-1}Y_n - e$ can be expressed in terms of the vector $R_n = D_n^{-1}Y_n = \{\frac{Y_{n,i}}{Y_{n-1,i}}\}$ (***ratio-vector***). The behavior of $\{Z_n(s_n)\}$ is then equivalent to that of $\{R_n\}$.

Data analysis and prediction model:
The similarity properties determine the behavior of the ordered pairs $\{\{s_n, Z_n(s_n)\}\}$ or $\{\{s_n, R_n\}\}$ and allow the prediction of the pairs $\{\{T_n, Y(T_n)\}\}$, *without any integration on the coarse grid*. Hence, *on the basis of Asymptotic or Numerical Similarity*, let n_s be the number of slices on which a sequential run has been conducted with (5) being reached. We seek a prediction data model on the pairs $\{\{s_n, R_n\}|n > n_s\}$. For that purpose, data analysis is carried out on the sequence: $\mathscr{D}^{(0)} = \{\{s_n, R_n\}|n = n_0, ..., n_s\}$. It leads to the model:

$$\{\{s_n, R_n\}|n > n_s\} = Fit(\mathscr{D}^{(0)}), \tag{6}$$

extrapolating best onto next slices. In case of asymptotic similarity, the data model should also take into consideration the convergence of $\{s_n, R_n\}$ to $\{s_L, R_L\}$ (see [7]). Besides, this model allows to get an estimate on N^0, least number of slices such that:

$$\sum_{n=n_s}^{N^0-1} \beta_n s_n < T \leq \sum_{n=n_s}^{N^0} \beta_n s_n. \tag{7}$$

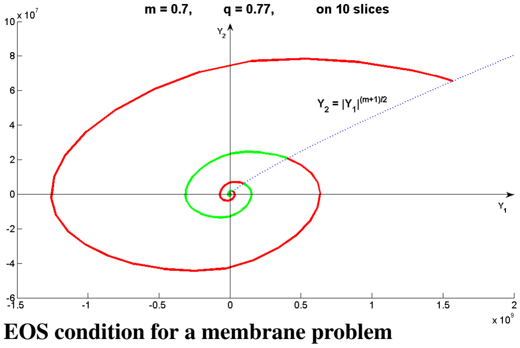
The case of a membrane problem:
Consider the second order IVP where one seeks $y : [0, T] \longrightarrow \mathbb{R}$ $(T \leq \infty)$ such that:

$$\begin{cases} y'' - b|y'|^{q-1}\, y' \ +\ |y|^{m-1}\, y = 0,\ t > 0, & (8.1) \\ y(0) = y_{1,0}, \qquad y'(0) = y_{2,0}. & (8.2) \end{cases} \tag{8}$$

This model describes the motion of a membrane element linked to a spring. When $b > 0$, the speed-up of the motion causes a "blow-up" of the solution, case that has been studied by Souplet et al in [13]. In [11], the rescaling method was applied to the case $m > 1$ and $q = \frac{2m}{m+1}$ where the solution explodes in finite time. We consider now the case $0 < m \leq q \leq \frac{2m}{m+1} \leq 1$. Carrying numerical integration of (8) has shown global existence of the solution on $[0,\infty)$ with **(a)** $\lim_{t\to\infty}|y(t)| = \lim_{t\to\infty}|y^{'}(t)| = \infty$, **(b)** $y(t)$ and $y^{'}(t)$ admit an infinite number of roots in the interval $[0,\infty)$.
Such behavior makes the solution, in the phase-plane (y,y'), spiral outwards about the origin toward infinity. The first step for solving (8) is to write it as a system of first order ODE's. Letting $Y_1(t) = y(t)$ and $Y_2(t) = y'(t)$ makes problem (8) equivalent to an initial value problem of the form $(\mathscr{S})$ where:

$$Y_0 = \begin{pmatrix} Y_{1,0} \\ Y_{2,0} \end{pmatrix} \text{ and } Y(t) = \begin{pmatrix} Y_1(t) \\ Y_2(t) \end{pmatrix}, \text{ with } F(t,Y) = F(Y) = \begin{pmatrix} Y_2 \\ b|Y_2|^{q-1}\,Y_2 \;-\; |Y_1|^{m-1}\,Y_1 \end{pmatrix}.$$

Because of the oscillatory behavior of the solution, one possible way to end the n^{th} slice could be whenever the trajectory of the solution, in the Y_1Y_2 phase plane, intersects the curve $Y_2 = |Y_1|^{\frac{m+1}{2}}$ in the first quadrant, thus completing an almost full rotation. The oscillating behavior of the solution makes such EOS condition guaranteed to be reached. Thus, one chooses:



EOS condition for a membrane problem

$$\forall\, W = \begin{pmatrix} W_1 \\ W_2 \end{pmatrix} \in \mathbb{R}^2, \quad H(W) = W_2 - |W_1|^{\frac{m+1}{2}}, \text{ and } \beta_n = |Y_{n-1,1}|^{\frac{1-m}{2}} = |Y_{n-1,2}|^{\frac{1-m}{m+1}}.$$

This yields the rescaled systems:

$$\begin{cases} \frac{dZ_{n,1}}{ds} = 1 + Z_{n,2}, \\ \frac{dZ_{n,2}}{ds} = b\gamma_n|1+Z_{n,2}|^{q-1}(1+Z_{n,2}) - |1+Z_{n,1}|^{m-1}(1+Z_{n,1}),\, 0 < s \leq s_n, \\ Z_{n,1}(0) = Z_{n,2}(0) = 0 \\ H(Z_n(s)) \neq 0,\, 0 < s < s_n \text{ and } H(Z_n(s_n)) = 0, \end{cases} \tag{9}$$

with $\gamma_n = |Y_{n-1,1}|^{\frac{m+1}{2}\left(q - \frac{2m}{m+1}\right)} \leq 1$. Thus, one checks the following [7]:
1. If $q = \frac{2m}{m+1}$, $\forall m \leq 1$, the rescaled systems (9) are invariant and equivalent to finding $Z(s) = (Z_1(s), Z_2(s))$, such that:

$$\begin{cases} \frac{dZ_1}{ds} = 1 + Z_2, \\ \frac{dZ_2}{ds} = b|1+Z_2|^{q-1}(1+Z_2) - |1+Z_1|^{m-1}(1+Z_1),\, 0 < s \leq s_1, \\ Z_1(0) = Z_2(0) = 0 \\ H(Z(s)) \neq 0,\, 0 < s < s_1 \text{ and } H(Z(s_1)) = 0, \end{cases} \tag{10}$$

2. If $0 < m \leq q < \frac{2m}{m+1} \leq 1$, then the rescaled systems (9) are asymptotically similar to the limit problem:

$$\begin{cases} \frac{dZ_{L,1}}{ds} = 1 + Z_{L,2}, \\ \frac{dZ_{L,2}}{ds} = -|1 + Z_{L,1}|^{m-1}(1 + Z_{L,1}), 0 < s \leq s_L, \\ Z_{L,1}(0) = Z_{L,2}(0) = 0 \\ H(Z_L(s)) \neq 0, 0 < s < s_L \text{ and } H(Z_L(s_L)) = 0, \end{cases} \tag{11}$$

4 Adaptive Parallel in Time (APTI) algorithm

The superscripts ^{p} and ^{c} denote predicted and calculated values respectively.
At the core of parallel in time algorithms, one must have a fine solver $\mathscr{F}$ that uniformly handles each of the rescaled problems $(\mathscr{S}_n')$. It is a software function defined by:
$$(\mathscr{F}) \qquad [Y_n^c, T_n^c] = \mathscr{F}(Y_{n-1}^p, T_{n-1}^p, \beta_n, D_n, F, E, tol),$$
on the basis of the functions F and E, given in $(\mathscr{S}_n)$, with $D_n = D(Y_{n-1})$ and β_n selected to insure obtaining a prediction model on the pairs $\{s_n, Z_n(s_n)\}$; tol is a **global** user's tolerance, the same as that used to check (4) or (5). The function $\mathscr{F}$ is designed so that:

$$\max\left\{\frac{||Y_{n-1} - Y_{n-1}^p||}{||Y_{n-1}||}, \frac{|T_{n-1} - T_{n-1}^p|}{|T_{n-1}|}\right\} = O(tol) \Rightarrow \max\left\{\frac{||Y_n - Y_n^c||}{||Y_n||}, \frac{|T_n - T_n^c|}{|T_n|}\right\} = O(tol). \tag{12}$$

Such fine solver $\mathscr{F}$ is discussed in [9], with a proof of (12) in the case when E is given by $E(U,W) = ||D^{-1}(W)(U - W)|| - S$; $\mathscr{F}$ takes in charge changing $(\mathscr{S}_n)$ to $(\mathscr{S}_n')$, then uses a high order explicit Runge-Kutta method with a **local** tolerance $tol_1 << tol$ to insure (12).

Theorem 1. *Assuming (12) is satisfied, then:*

$$\begin{cases} \max\left\{\frac{||Y_{n-1} - Y_{n-1}^p||}{||Y_{n-1}||}, \frac{|T_{n-1} - T_{n-1}^p|}{|T_{n-1}|}\right\} = O(tol) \\ \max\left\{\frac{||Y_n^p - Y_n^c||}{||Y_n^p||}, \frac{|T_n^p - T_n^c|}{|T_n^p|}\right\} = O(tol) \end{cases} \Longrightarrow \max\left\{\begin{matrix} \frac{||Y_n - Y_n^p||}{||Y_n||} \\ \frac{|T_n - T_n^p|}{|T_n|} \end{matrix}\right\} = O(tol).$$

An **iterative process** can now be initiated using a parallel architecture with P processors. For increasing the speed-up, we adopt a strategy of duplication of sequential tasks on all processors (that reduces communications and avoids idle time).

Initialization step *duplicated on all P processors:*
- Set the iteration index l to 0.
- Solve sequentially problem $\{(\mathscr{S}_n')\}$ on $m^{(0)} = n_s$ time-slices using $\mathscr{F}$.
- Obtain $\{(T_j^{(0)}, Y_j^{(0)}) | j = 0, ..., m^{(0)}\}$ and let $T^{(0)} = \max\{T_j^{(0)}\}$.
- Compute N^0 according to estimate (7).

Allocation of tasks on the P **processors:** At this point, the remaining time-slices $(n > m^{(0)})$ are statically allocated, based on a cyclic distribution: processor pr will be assigned slices number n where $(n - m^{(0)})$ is congruent to pr mod P. This provides an optimized synchronization and a *load balanced distribution* of the work.

While $T^{(l)} < T$ **(Iterative steps):**
1. All P processors duplicate the task of predicting $\{(T_j^p, Y_j^p) | j = m^{(l)} + 1, ..., N^l\}$, using $Fit(\mathscr{D}^{(l)})$ from (6).
2. Every processor $pr \in \{1, ..., P\}$ executes in parallel the following:

a. pr solves its first slice n ($n \geq m^{(l)}+1$) and computes Y_n^c and T_n^c using $\mathscr{F}$.
b. Processor pr computes $\max\{||Y_n^p - Y_n^c||/||Y_n^p||, |T_n^p - T_n^c|/|T_n^p|\}$.
c. While $\max\{||Y_n^p - Y_n^c||/||Y_n^p||, |T_n^p - T_n^c|/|T_n^p|\} \leq tol$ and $n < N^l$, processor pr takes on its assigned next slice, based on theorem 1, and repeats 2(a) (b).
d. If $\max\{||Y_n^p - Y_n^c||/||Y_n^p||, |T_n^p - T_n^c|/|T_n^p|\} > tol$, processor pr stops the execution (*the remaining time-slices need not to be solved*). It sends to the master processor (processor 1) the index $\mathscr{I}(pr)$ of the last slice having converged, together with the new $\{T_n^c, Y_n^c\}_{n>m^{(l)}}$.

3. Master processor synthesizes the received data and updates the following:
   a. Iteration number $l := l+1$ and number of so far solved slices $m^{(l)} := \max_{pr} \mathscr{I}(pr)$.
   b. $\left\{\left(T_j^{(l)}, Y_j^{(l)}\right) | j = 0, ..., m^{(l)}\right\}$ with $\left(T_j^{(l)}, Y_j^{(l)}\right) := \left(T_j^{(l-1)}, Y_j^{(l-1)}\right), \forall j = 0, ..., m^{(l-1)}\}$.
   c. $T^{(l)} := \max\{T_j^{(l)}\}$, and $N^{(l)}$ from estimate (7), and the set $D^{(l)}$ to be used by the function Fit (as set in (6)).

   Then, the master processor sends $T^{(l)}$, $N^{(l)}$ and $D^{(l)}$ to all other processors.

End While

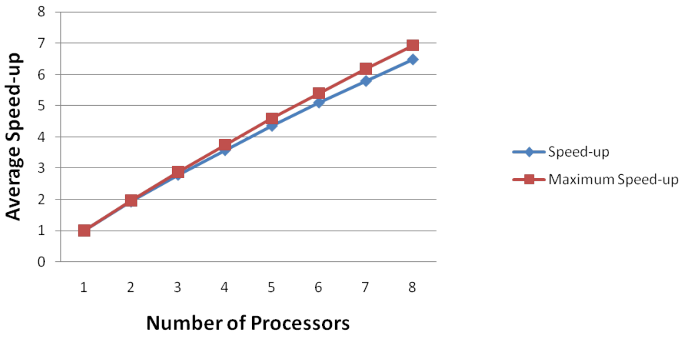
Remark: In case of autonomous problems $F(t,Y) \equiv F(Y)$, one needs not to predict the starting values $\{T_n^p\}$ of the time. Given $\{Y_n^p\}$ only, the rescaling technique allows solving (S_n') in a local time s, thus providing in parallel $\{s_n\}$ and the size $T_n^c - T_{n-1}^c = \Delta T_n^c = \beta_n s_n$ of time-slices. Then, $\{T_n^c\}$ is reconstituted from received $\{\Delta T_n^c\}$.

5 Numerical results

The table below summarizes some results obtained by the above APTI algorithm on the membrane problem, in the case of asymptotic similarity when $0 < m \leq q < \frac{2m}{m+1}$ and for 8 combinations of the problem parameters m and q, with $b = 1$. The total number of slices N, and therefore the interval of integration $[0,T]$, corresponds to the maximum (or almost) number preventing the explosive solution from exceeding the machine capacity. The total number of iterations vary from one case to another, but in all cases, the results show how small is this number compared to the total number of slices. This ascertains the fast convergence of the method when applied to this type of problems. S_i represents the speed-up obtained when using i processors (compared to the sequential run time of the same rescaling method) and S_i^{max} is the corresponding maximum speed-up stated by Amdhal's law. The following tolerances have been used: $tol = 5 \times 10^{-6}$ (global) and $tol_1 = 10^{-14}$ (local).

Case	1	2	3	4	5	6	7	8
m	**0.8**	**0.7**	**0.7**	**0.6**	**0.6**	**0.6**	**0.5**	**0.5**
q	**0.84**	**0.74**	**0.77**	**0.66**	**0.69**	**0.72**	**0.55**	**0.60**
T	$\approx 10^{14}$	$\approx 10^{29}$	$\approx 10^{28}$	$\approx 10^{14}$	$\approx 10^{18}$	$\approx 10^{30}$	$\approx 10^{17}$	$\approx 10^{25}$
N	**65000**	**65000**	**50000**	**65000**	**65000**	**65000**	**65000**	**65000**
n_s	1499	1143	1471	1156	1414	1993	1053	1385
$\mathbf{n_I}$	**6**	**11**	**12**	**35**	**28**	**23**	**5**	**5**
$\mathbf{S_2}$	**1.88**	**1.93**	**1.93**	**1.96**	**1.94**	**1.91**	**1.96**	**1.94**
S_2^{max}	1.95	1.97	1.94	1.97	1.96	1.94	1.97	1.96
$\mathbf{S_4}$	**3.57**	**3.66**	**3.50**	**3.59**	**3.56**	**3.44**	**3.68**	**3.63**
S_4^{max}	3.74	3.80	3.68	3.80	3.75	3.66	3.81	3.76
$\mathbf{S_8}$	**6.47**	**6.76**	**6.23**	**6.57**	**6.38**	**6.05**	**6.82**	**6.59**
S_8^{max}	6.89	7.12	6.63	7.11	6.94	6.59	7.19	6.96

Actually, the method has been tested on the previous 8 cases, using 2, 3, 4, 5, 6, 7, and 8 processors. The opposite figure shows how the values of speed-up, averaged on the 8 cases above, vary with the number of processors and how close it is to the maximum speed-up.



Conclusion

The application of the adaptive parallel in time algorithm we have presented is not unconditional and requires the prior knowledge of the solution behavior and the existence of an EOS condition inducing the predictability of the end-of-slice values. However, when applicable, APTI algorithm yields a fast convergence due to accurate predictions that do not require any sequential integration on the coarse grid. Besides, not all the remaining time-slices are solved at each iteration and communications are minimized in number and size. Our future work aims at experimenting the method on additional application problems.

References

1. Bal, G., Wu, Q.: Symplectic parareal. In: M. Bercovier, M. Gander, R. Kornhuber, O. Widlund (eds.) DD08, Lecture Notes in computational Science and Eng., pp. 189–202. Springer (2008)
2. Chartier, P., Philippe, B.: A parallel shooting technique for solving dissipative ode's. Computing **51**, 209–236 (1993)
3. Erhel, J., Rault, S.: Algorithme parallèle pour le calcul d'orbites. Techniques et Sciences Informatiques **19**, 649–673 (2000)
4. Farhat, C., Chandesris, M.: Time-decomposed parallel time-integrators. Int. J. Numer. Meth. Engng **58**, 1397–1434 (2003)
5. Lions, J., Maday, Y., Turinici, G.: Résolution d'edp par un schéma en temps "pararéel". C.R.Acad.Sci.Paris **332**, 661–668 (2001)
6. Maday, Y., Bal, G.: A parareal time discretization for non-linear pde's with application to the pricing of an american put. In: I.H. et al (ed.) DD02, Comp. Sc., pp. 189–202. Springer (2002)
7. Makhoul-Karam, N.: Time-slicing, rescaling & ratio-based parallel time integration. TEL (2012). URL http://tel.archives-ouvertes.fr/tel-00743132
8. Nassif, N., Fayad, D., Cortas, M.: Sliced-time computations with rescaling for blowing-up solutions to init. val. pbs. In: S.S. et al. (ed.) ICCS 05, Comp. Sc., pp. 58–65. Springer (2005)
9. Nassif, N., Makhoul-Karam, N., Erhel, J.: Globally adaptive explicit numerical methods for exploding systems of ordinary differential equations. APNUM (2011). URL http://dx.doi.org/10.1016/j.apnum.2011.09.009
10. Nassif, N., Makhoul-Karam, N., Soukiassian, Y.: A new approach for solving evolution problems in time-parallel way. In: V.A. al (ed.) ICCS 06, Comp. Sc., pp. 148–155. Springer (2006)
11. Nassif, N., Makhoul-Karam, N., Soukiassian, Y.: Computation of blowing-up solutions for second-order differential equations using re-scaling techniques. JCAM **227**, 185–195 (2009)
12. Nievergelt, J.: Parallel methods for integration of ode's. Comm. ACM **7**, 731–733 (1964)
13. Souplet, P.: Critical exponents, special large-time behavior and oscillatory blow-up in nonlinear ode's. Differential and Integral Equations **11**, 147–167 (1998)