A novel, semilagrangian, coarse solver for the Parareal technique and its aplication to 2D drift-wave (BETA) and 5D gyrokinetic (GENE) turbulence simulations

J. M. Reynolds-Barredo[1], D. E. Newman[1,2], R. Sanchez[1], F. Jenko[3]

[1] *Universidad Carlos III de Madrid, Leganes, Spain*

[2] *University of Alaska Fairbanks, Fairbanks, USA*

[3] *Max-Planck-Institut fur Plasmaphysik EURATOM-IPP, Garching, Germany*

Abstract

In this work, we apply the Parareal time-parallelization technique [1] to convection-dominated problems. In particular, to a 2D drift-waves case using the BETA code and to a 5D gyro-kinetic ITG simulation using GENE code. Although partial success was previously reported applying parareal to the drift-wave BETA runs [2], the speed-up of the process was limited by the compromise that must be reached between having a sufficiently fast coarse serial solver for the problem, and how far from the actual solution the coarse solver is pushed by the approximations made. This limitation becomes more dramatic in the case of GENE simulations. Here, we propose and test a new and promising coarse solver based on a semi-lagrangian time advance. Its advantage comes from the fact that a significant part of the coarse solver can be solved in parallel. As a result, it can be made faster without paying the penalty of excessive simplifications.

Introduction to Parareal

Generally, physical problems simulated on a computer can be efficiently parallelized only up to a certain number of processors. With the supercomputers that are available nowadays, there are often many more processors available then the number that can be efficiently utilized with standard techniques. It is therefore clear that new parallelization techniques that permit to increase the number of processors are of great interest. Parareal [1] is a new such parallelization technique that focuses on the time coordinate. It is based on an iterative process with two stages for every iteration: 1) a fast coarse time propagator that gives an approximated solution for all time; 2) an accurate time propagator that is used to correct the solution. The first stage is fast but must be computed sequentially. The second one is expensive but can be computed in parallel. The key to success depends on choosing an adequate coarse solver: it must be much faster than the fine one but, at the same time, not diverge too much from the actual solution.

We introduce next the algorithm with more detail, but we refer the reader to [3] for a more in-depth description. In general, we will look for the solution represented by the state vector

$\lambda(\mathbf{r},t)$ that satisfies the PDE:

$$\frac{\partial \lambda(\mathbf{r},t)}{\partial t} = R[\lambda(\mathbf{r},t)] \tag{1}$$

Being $R[\cdot]$ a linear or non-linear operator. The total simulation time, T, is split in N chunks, each of size ΔT, so that $T = N\Delta T$. Next, two propagators are defined:

- $F[\lambda(\mathbf{r},t)]$ is the fine propagator that evolves $\lambda(\mathbf{r},t)$ exactly during a chunk time ΔT.
- $G[\lambda(\mathbf{r},t)]$ is the (fast) coarse propagator that gives an approximation to $F[\lambda(\mathbf{r},t)]$.

As previously commented, parareal is an iterative algorithm so, for every iteration k and chunk s, it offers the approximation $\lambda^{k,s}(\mathbf{r}) \simeq \lambda(\mathbf{r},T_s)$. Every iteration k is composed of two stages:

1. For all chunks not converged (labeled by s), compute serially $\lambda^{k,s}(\mathbf{r}) = G[\lambda^{k,s-1}(\mathbf{r})] + F[\lambda^{k-1,s-1}(\mathbf{r})] - G[\lambda^{k-1,s-1}(\mathbf{r})]$. This is fast, because G has a low computational cost and $F[\lambda^{k-1,s-1}(\mathbf{r})]$ was already computed in previous iterations.
2. Compute in parallel $F[\lambda^{k,s}(\mathbf{r})]$ for all not-converged chunks. This quantity will be used in the next iteration. F is expensive computationally, but it is computed in parallel.
3. Check for convergence of new chunks.

If the total number of required iterations K to converge is small relative to N, and if the coarse solver is fast enough compared with the fine solver so that its contribution to the walltime is negligible, the total wallclock time of the simulation is reduced by a factor of the order of N/K. It can be seen that K is small if the coarse solver is a not-too-bad approximation of the fine solver. This is a rather imprecise statement but, in most practical cases, this can only be checked by trial-and-error. Be as it may, the key of a good parareal implementation is to find a fast coarse solver that wanders not to far from the fine one. But finding such a solver is not easy.

Proposed new coarse solver

We propose a new coarse solver that can be made faster without making excessive simplifications. It is based on using a semi-lagrangian technique that keeps the solver stable and allows to parallelize parts of the coarse solver thus reducing the wall-clock time. To illustrate it, we will start with a simple convection problem with fixed velocity, that can be written as:

$$\frac{\partial \lambda(\mathbf{r},t)}{\partial t} + \mathbf{v}(\mathbf{r}) \cdot \nabla \lambda(\mathbf{r},t) = 0 \tag{2}$$

As Fig. 2 shows, to compute $\lambda(\mathbf{r_0},t_0)$ at node r_0 at time t_0, a semi-lagrangian time advance requires:

1. Convect "back in time" the node up to the time $t_0 - dT$ and obtain the position r'_0.

2. Because the problem is convective, $\lambda(\mathbf{r_0}, T_0)$ is exactly equal to $\lambda(\mathbf{r'_0}, T_0 - dt)$. Thus, it can be obtained through interpolation from the λ solution in the previous time step.

At first sight, the semilagrangian time advance may not appear to be faster than say, an explicit one. But because the convection velocity has been kept fixed in Eq. 2, the first stage of the semilagrangian time advance (the convection of the nodes position) could be precomputed and stored previously to the application of the parareal algorithm. In addition, because the interpolation positions are known, all the interpolation coefficients could also be precomputed. The result is that, to solve the convection problem in parareal, the computation of the coarse solver would consist only in applying the interpolation coefficients to λ, which is just sparse matrix-vector multiplication. Thus, this implementation can become a fast and very accurate coarse solver.

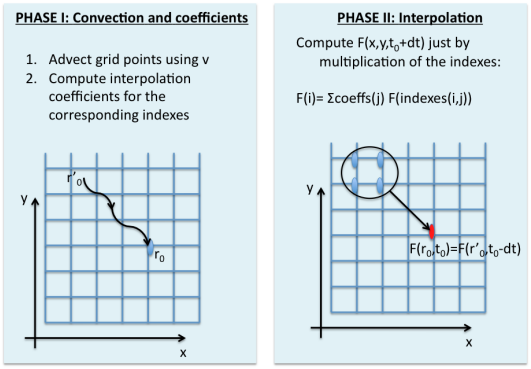

Figure 1: *Semilagrangian two stages algorithm.*

Of course, the difficulty in applying the proposed coarse solver to real world problems is that, in general, the convection velocity changes in time as a function of λ. This converts the problem in non-linear and more complicated to solve:

$$\frac{\partial \lambda(\mathbf{r},t)}{\partial t} + \mathbf{v}[\lambda(\mathbf{r},\mathbf{t})] \cdot \nabla \lambda(\mathbf{r},t) = 0 \tag{3}$$

However, it is possible to apply an extension of the proposed coarse solver by **obtaining the convection velocity field from the previous parareal iteration results**. This allows that, at the end of one parareal iteration, the convection of the nodes for the semilagrangian time advance could be computed based in that iteration results and prepare the interpolation coefficient for the next iteration, all of that **done in parallel**. Thus, one of the highlights of this work is to realize that, even if standard parareal states that the coarse solver must be computed serially, *part of the work could be computed in parallel if semilagrangian time advance is used*. Only the sparse matrix - vector multiplication required to effectuate the interpolation should be computed serially. As the parareal algorithm converges to the correct solution, the convection velocity field will also converge.

Applications to BETA and GENE codes

The proposed coarse solver has been tested on BETA and GENE. BETA implements a 2D plasma turbulence model based on dissipative-trapped-electron modes in a uniform magnetic field [4]. GENE is a state-of-the-art 5D gyrokinetic continuum code [5], that solves the non-linear gyrokinetic equation on a fixed eulerian grid. In both cases, the coarse solver proposed here has shown good convergence properties (10-16 chunks per iteration), nice ideal efficiencies for a parareal simulation ($15-25\%$) and is stable. Fig. 2 shows that even at the high spatial modes the method shows good convergence properties and that the parareal solution accurately reproduces the serial solution for several turnover times.

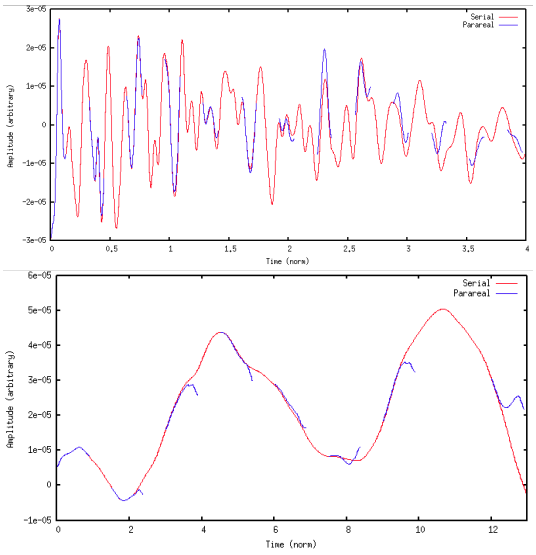

Figure 2: *Time evolution of the amplitude of an spatial mode for both BETA (top) and GENE (bottom). The red line represent the serial solution and the blue one some of the iterations of the parareal one. Good convergence is found for several turbulence times.*

Acknowledgements

This work was also supported in part by the Spanish projects MICINN ENE2009-12213-C03-03 and MINECO ENE2012-33219. Part of the research was carried out at the University of Alaska Fairbanks, funded by the DOE Office of Science Grant No. DE-FG02-04ER54741. The authors are grateful for grants of supercomputing resources at the University of Alaska Arctic Region Supercomputing Center (ARSC) in Fairbanks. DEN gratefully acknowledges the support of a *Catedra de Excelencia* from Universidad Carlos III - Banco de Santander Project.

References

[1] J. Lions, Y. Maday, G. Turinici, C. R. Acad. Sci. Paris-Serie **332** (7) (2001)

[2] D. Samaddar, D.E. Newman, R. Sanchez, J. Comput. Phys. **229**, 6558 (2010)

[3] J.M. Reynolds-Barredo, D. E. Newman, R. Sanchez, D. Samaddar, L.A. Berry and W.R. Elwasif, J. Comp. Phys. **231** 7851 (2012).

[4] D.E. Newman, P.W. Terry, P.H. Diamond, Phys. Fluids B **4** (3) 599 (1992)

[5] F. Jenko, B. Dorland, M. Kotschenreuther, B. Rogers, Phys. Plasmas **7** 1904 (2000)

[6] A.M. Dimits et al, Phys. Plasmas **7** 969 (2000)