Computer Physics Communications 235 (2019) 246–257

Contents lists available at ScienceDirect

Computer Physics Communications

journal homepage: www.elsevier.com/locate/cpc

# Application of the parareal algorithm to simulations of ELMs in ITER plasma

D. Samaddar [a,*], D.P. Coster [b], X. Bonnin [c], L.A. Berry [d], W.R. Elwasif [d], D.B. Batchelor [d]

[a] CCFE, Culham Science Centre, Abingdon, Oxon, OX14 3DB UK
[b] Max-Planck-Institut für Plasmaphysik, D-85748 Garching, Germany
[c] ITER Organization, Route de Vinon sur Verdon, 13115 St Paul Lez Durance, France
[d] Oak Ridge National Laboratory, Oak Ridge, USA

ARTICLE INFO

*Article history:*
Received 28 March 2018
Received in revised form 15 June 2018
Accepted 13 August 2018
Available online 26 September 2018

*Keywords:*
Time parallelization
Parareal algorithm
Scrape off layer (SOL)
SOLPS
Magnetically confined plasmas
Edge localized modes or ELMs
ITER
Tokamak

ABSTRACT

This paper explores the application of the parareal algorithm to simulations of ELMs in ITER plasma. The primary focus of this research is identifying the parameters that lead to optimum performance. Since the plasma dynamics vary extremely fast during an ELM cycle, a straightforward application of the algorithm is not possible and a modification to the standard parareal correction is implemented. The size of the time chunks also have an impact on the performance and needs to be optimized. A computational gain of 7.8 is obtained with 48 processors to illustrate that the parareal algorithm can be successfully applied to ELM plasma.

## 1. Introduction

The ever increasing demand for improving complexities in simulations requires maximizing the efficient use of computational resources. Traditional parallelization techniques (such as space parallelization) although reducing the wall clock time, often reach saturation on modern supercomputing machines. Parallelizing the time domain ushers in a new possibility of optimizing resource utilization. Parallel in time algorithms are not a replacement for, but are complementary to, other schemes of parallelization. This therefore allows utilization of the gain achieved from existing parallelization and adding a large improvement in computational gain.

Various parallel in time algorithms exist such as PITA [1], parareal [2], RIDC [3,4], and PFASST [5]. This work explores the application of the Parareal algorithm to a complex, non-linear simulation in fusion plasma. Of all the algorithms that achieve temporal parallelization, the Parareal algorithm has been the one to have been used the most in non-linear plasma physics simulations over the years [6–8]. The algorithm was introduced in [2] and its convergence and stability have been analysed in great detail in [9–17].

This work seeks to explore temporal parallelization of simulations of Edge Localized Modes (ELMs) that play a crucial role in the successful operation of any fusion device [18–22]. ELMs are simulated using the multi-fluid codes SOLPS 5.0 [23,24] and SOLPS-ITER [25,26]. ELMs have been observed in experiments on present machines [27,28]. These modes exhibit irregular or quasi-regular periodicity and lead to deposition of huge power fluxes on plasma facing walls. While ELMs are an area of strong active research in the fusion community and there is significant information collected from experimental databases, a complete understanding of ELMs is still elusive to scientists.

ELMs are an MHD phenomena and MHD codes such as JOREK [29] are used to investigate the cause of an ELM. ELMS can also be studied using full scale turbulence codes such as *BOUT* ++ [30]. However, investigating the impact of ELMs on the evolution of the plasma properties due to interactions between all ion species and electrons requires a different approach involving transport equations such as in SOLPS. A number of largely parallel codes like NIMROD [31], GEM [32], and M3D [33] are used to investigate ELMS but no simulation currently fully captures both aspects (cause and impact) of ELMs.

[*] Corresponding author.
*E-mail address:* Debasmita.Samaddar@ukaea.uk (D. Samaddar).

https://doi.org/10.1016/j.cpc.2018.08.007

SOLPS solves the fluid equations (which are not parallelized) for the ions and can be coupled with the parallel Eirene code to simulate the neutral particles in the plasma. However, simulating 1s of ELM can take 100 days of walltime. The cases discussed in [34] which involved a much smaller tokamak AUG than ITER typically took 200 days of wall clock time.

Since a major hindrance in studying the behaviour of ELMs using computer simulations is their demand for extremely large wall clock times. This is why adding a new dimension to the parallelization is of significance in ELM research. A successful application of the Parareal algorithm makes these simulations much more feasible, allowing the inclusion of more complex physics while still maintaining a reasonable computation time. However, due to the strong non-linear phenomena involved, the application of temporal parallelization to simulations of ELMs is unsurprisingly far from straightforward and this work attempts to identify the computational parameters and regimes that allow the best performance.

The present research is a follow-up of the work detailed in [8]. [8] demonstrated that the parareal algorithm is applicable to edge plasma simulations but relatively 'steady' states were used as test beds. This work extends that application to more strongly coupled non-linear phenomena called ELMs. Due to the complex non-linear behaviour of ELMs, the application poses new challenges. The characteristic time-scale of the plasma in [8] was ∼10 ms while the time scales involved in this present work are very different. ELMs involve time dependent behaviour with dramatic changes to the system that last for 1 ms. The plasma conditions continue to evolve after an ELM and before the onset of the next ELM. During an ELM, the power at the targets can vary by more than an order of magnitude for example.

A modification to the application of the parareal correction allows convergence and generates computational gain.

This paper is organized as follows: ELMs are described in Sections 2 and 3 details how they are implemented in the code to be observed in the simulations. Section 4 describes the Parareal Algorithm and the results are described in Section 5.

## 2. ELMs

Edge Localized Modes (ELMs) [35,36] are periodic bursts of instabilities at the plasma edge in a tokamak. A tokamak is a fusion device of toroidal geometry with complex magnetic field lines to confine the burning plasma of a few hundred million Kelvin within the vessel. As has been mentioned in [8], simulations of the plasma edge are particularly challenging due to the interactions with the materials of the walls of the vessel accompanied by both radial (perpendicular to the magnetic field lines) and parallel (parallel to the field lines) transports.

It is desirable to operate a tokamak in the H-mode (High Confinement Mode) [18,37–39] since that greatly enhances the energy confinement time of the plasma. The energy confinement time $\tau$ is the characteristic time scale in which the energy escapes the plasma. In fact, H-mode confinement time is typically twice that of L-mode (Low Confinement mode). The sharp transition from L-mode to H-mode occurs when the input heating power is above a certain threshold. A sharp increase in the profile gradients (such as of pressure, density or temperature) occurs over a very narrow region (typically a few cm) at the plasma edge. This results in the formation of the 'edge transport barrier' or 'H-mode pedestal' which generates the improved confinement. The suppression of turbulent transport at the plasma edge is believed to be the reason behind the improved confinement—although a complete understanding of the mechanisms at play is still an area of active research.

While the improved confinement makes the H-mode an attractive operating regime, the presence of instabilities makes it more challenging. Two MHD instabilities become prominent. The increasing pressure gradient leads to ballooning instabilities and the gradient in the edge current generates the peeling instability. These two instabilities are believed to be responsible for ELMs [18,19].

ELMs are characterized by sudden bursts of energy and a transient rise of heat loads on the divertor targets of the device. Depending on their amplitudes and frequencies, ELMs are classified into types I (large amplitudes—hence often termed as 'giant' ELMs) and II and III (smaller, hence often known as 'grassy' ELMs). Current predictions estimate [21,40,41] the heat flux due to ELMs on the divertor plates at ITER (the world's biggest experimental tokamak) will be up to 20 times larger than what can be tolerated for a reasonable lifetime of the target materials. This makes research into ELM control and mitigation of primary importance for making fusion a viable source of energy.

Despite the potential significant reduction of ITER's divertor target lifetime due to power loads by ELMs, the particle outflux associated with the ELMs has positive effects on H-mode plasma performance. Indeed ELMs are required in present experiments to exhaust impurities from the confined plasma and a similar role is expected for ITER. As a result, controlled ELMs in H-mode is the desired operational mode for ITER.

## 3. ELMs in the SOLPS code

The SOLPS 5.0 and SOLPS-ITER codes are described in detail in [23–26]. SOLPS stands for Scrape Off Layer Plasma Simulator which, as the name suggests, simulates the Scrape Off Layer (SOL). The SOL is the region between the wall of the device and the Last Closed Flux Surface (LCFS), characterized by open field lines. Both codes solve the same set of equations and SOLPS-ITER is considered to be an upgraded version of SOLPS5.0. Both codes use the B2.5 package that solves the Braginskii fluid equations for multiple species or all charge states of every individual element present in the plasma [20]. The density equation is given by Eq. (1) as an example of the set of equations solved, which are described in detail in [23].

$$\frac{\partial n}{\partial t}+\frac{1}{\sqrt{g}}\frac{\partial}{\partial x}\left(\frac{\sqrt{g}n}{h_x}\left(b_xV_\parallel+b_zV_\perp\right)\right)+\frac{1}{\sqrt{g}}\frac{\partial}{\partial y}\left(\frac{\sqrt{g}}{h_y}nV_y\right)=S^n \qquad (1)$$

In Eq. (1), n is density, V is the velocity and $S^n$ is the density source term. Following [23], the coefficients h and g are given by $h_x=\frac{1}{\|\nabla x\|}$, $h_y=\frac{1}{\|\nabla y\|}$, $h_z=2\pi R$ with R being the major radius and $\sqrt{g}=h_xh_yh_z$. The $\|$ and $\perp$ components are with respect to the direction of the magnetic field.

The regimes studied are sufficiently collisional to justify the use of fluid equations. In the present case, the fluid model is used for both the charged species as well as the neutrals in the plasma as in [20]. Ideally, it is desirable to use a kinetic model to simulate the neutral species in the plasma, but using the fluid model makes the computations tractable.

It is interesting to note that ELMs are believed to be intrinsically MHD phenomena while SOLPS is not an MHD code. It must be clarified that a code like SOLPS is used to study the impact of ELMs (and provide insight into the effects of individual species which is impossible to perform in an MHD code) rather than the mechanisms for their generation (which is typically studied using MHD codes [29]).

ELMs characteristically enhance the radial transport in a plasma [42]. This is typically simulated by increasing the diffusive or convective coefficient for a brief period (say 1 ms) at regular intervals. The impact of that increase successfully lasts much longer than

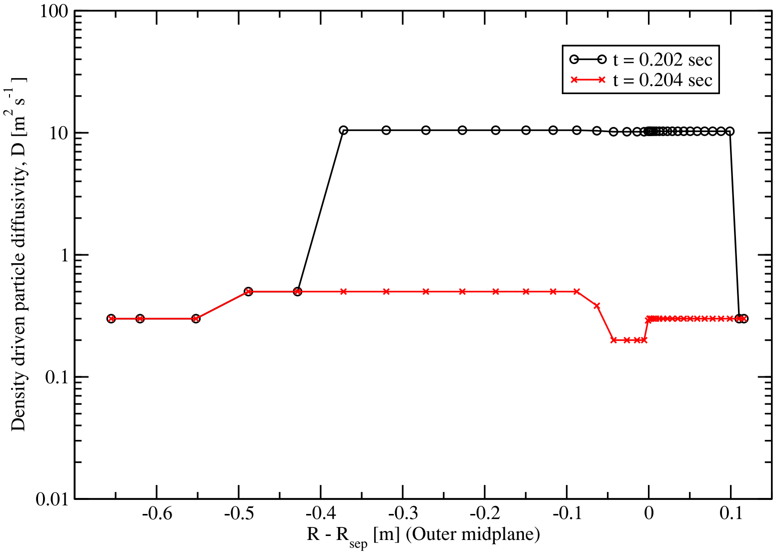


**Fig. 1.** The particle density gradient driven radial diffusivity is increased by 10 times to simulate ELMs.

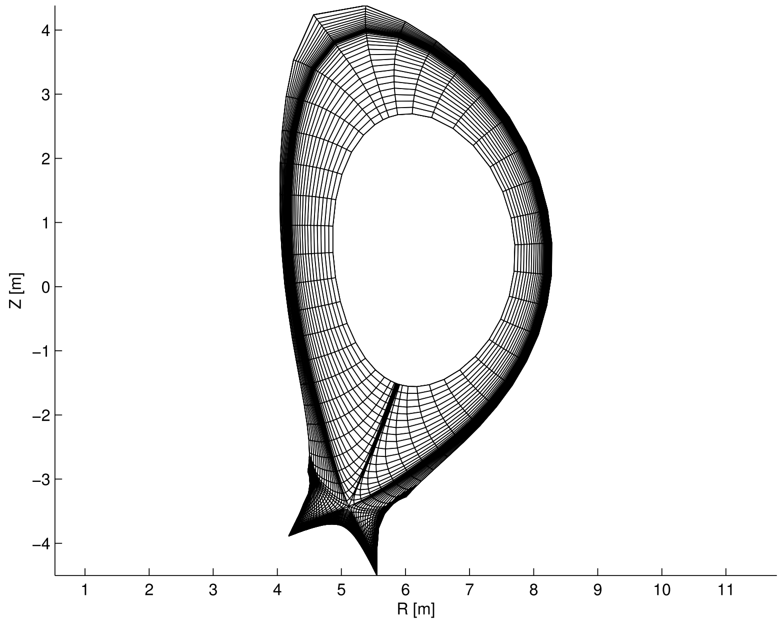
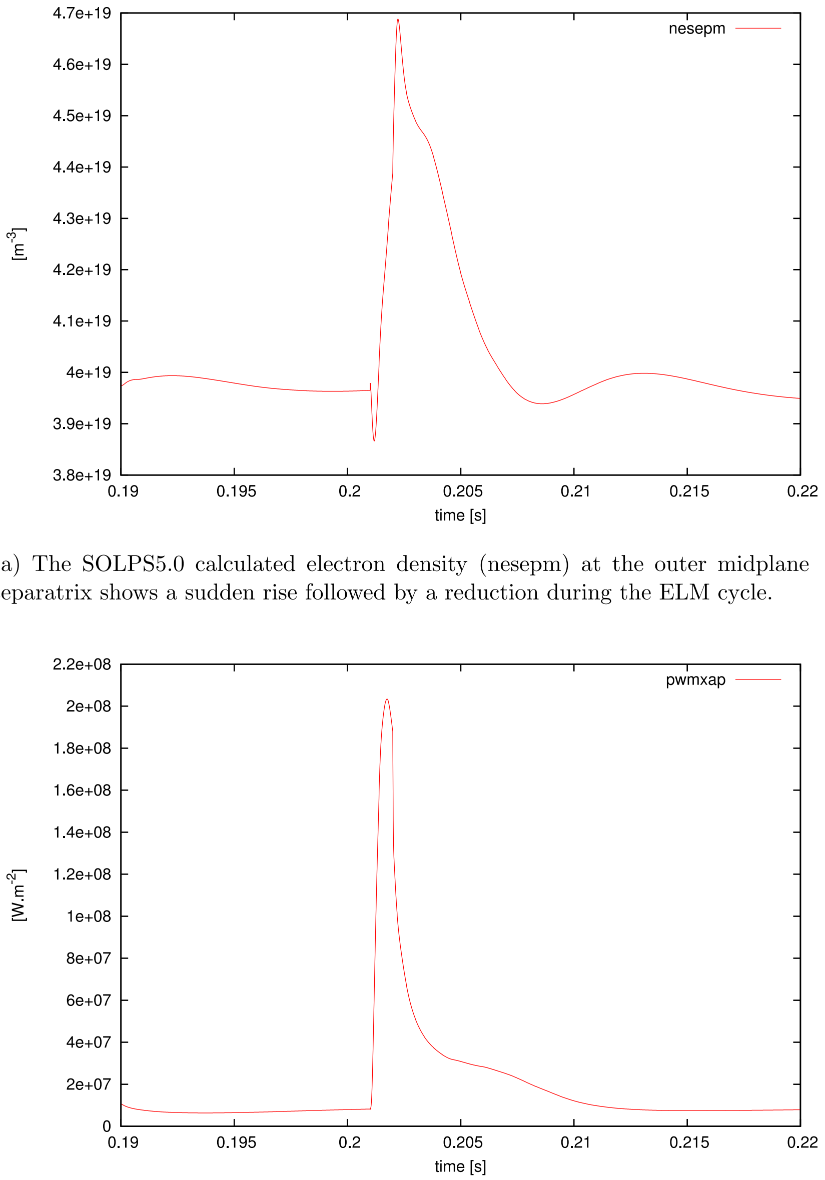


**Fig. 2.** A grid of size 96 × 36 was used for the fine or serial computation of the ELM plasma in the ITER device.

1 ms and simulates the behaviour of an ELM-plasma. These simulations have been successfully benchmarked against experiments in the JET, TCV and AUG tokamaks [21,22,34].

[8] studied relatively simpler cases in the tokamaks MAST and DIII-D. The present work simulates a significantly larger device – ITER – with a much higher number of species mix (98) in the plasma. The physics involved in these simulations are very similar to the cases described in [20]. The species include all ions and neutrals of Deuterium, Tritium, Helium, Beryllium, Neon and Tungsten and the complex interactions between the various charge states are included in the simulations. As in [20], prompt redeposition of Tungsten, which is otherwise released into the plasma as a result of sputtering at the divertor targets, is included in the calculations. It may be clarified that the redeposition is based on a simple approximation but a more accurate ELM based calculation is beyond the scope of this work. The density gradient driven diffusivity has been increased by 10 times for 1 ms in every 20 ms to simulate the ELM triggered plasma. This is illustrated in Fig. 1 where the diffusivity along the outer midplane is plotted for two different times. The ELM affected area on the outer midplane is estimated to be 40 cm for ITER. The ELM could also have been simulated by varying the convective coefficient instead of the diffusive coefficient. However, as discussed in [20], previous results have not demonstrated significant variation in the quantities of concern such as the maximum total power flux at the divertors. But, looking at the behaviour of parareal convergence where the ELMs are generated by varying the convective coefficient may be investigated in the future.

A computational grid of size 96 × 36 as shown in Fig. 2 was used as the fine model for the SOLPS computations of ELMs in the ITER plasma. The typical size of the timesteps for the fine computations



(a) The SOLPS5.0 calculated electron density (nesepm) at the outer midplane separatrix shows a sudden rise followed by a reduction during the ELM cycle.

(b) The maximum total power flux on the outer target for the same case as Fig.3a.

**Fig. 3.** A typical ELM cycle using the SOLPS5.0 code package with the Fine solver as a serial computation is demonstrated.

was $10^{-5}$ s. A typical ELM cycle is shown in Fig. 3 where the electron density (in $\mathrm{m}^{-3}$) at the separatrix on the midplane (henceforth will be referred to as nesepm) and the maximum total power flux (in $\mathrm{W\,m}^{-2}$) (henceforth referred to as pwmxap) on the outer target are plotted against time. It is seen that both these quantities are greatly increased during the ELM. The ELM-crash phase includes the peak values which then decrease through the remaining crash phase.

The variables that are commonly being referred to in the rest of the paper are summarized in Table 1.

## 4. Parareal algorithm

The parareal algorithm was first introduced in [2]. It is described in detail in [8]. The algorithm uses a predictor–corrector approach with an accurate fine (F) integrator and another coarse integrator (G). Identifying the optimum coarse predictor is generally the biggest challenge in all complex applications. The parareal correction is given by Eq. (2).

$$\lambda^k_{i+1} = F(\lambda_i^{k-1}) + G(\lambda_i^k) - G(\lambda_i^{k-1}) \qquad (2)$$

$\lambda^k_{i+1}$ is the initial state for the $(i+1)$th time slice at the kth iteration. At the kth iteration $\lambda_i^k$ is evolved to $\lambda^k_{i+1}$ using F($\lambda_i^k$) and G($\lambda_i^k$).

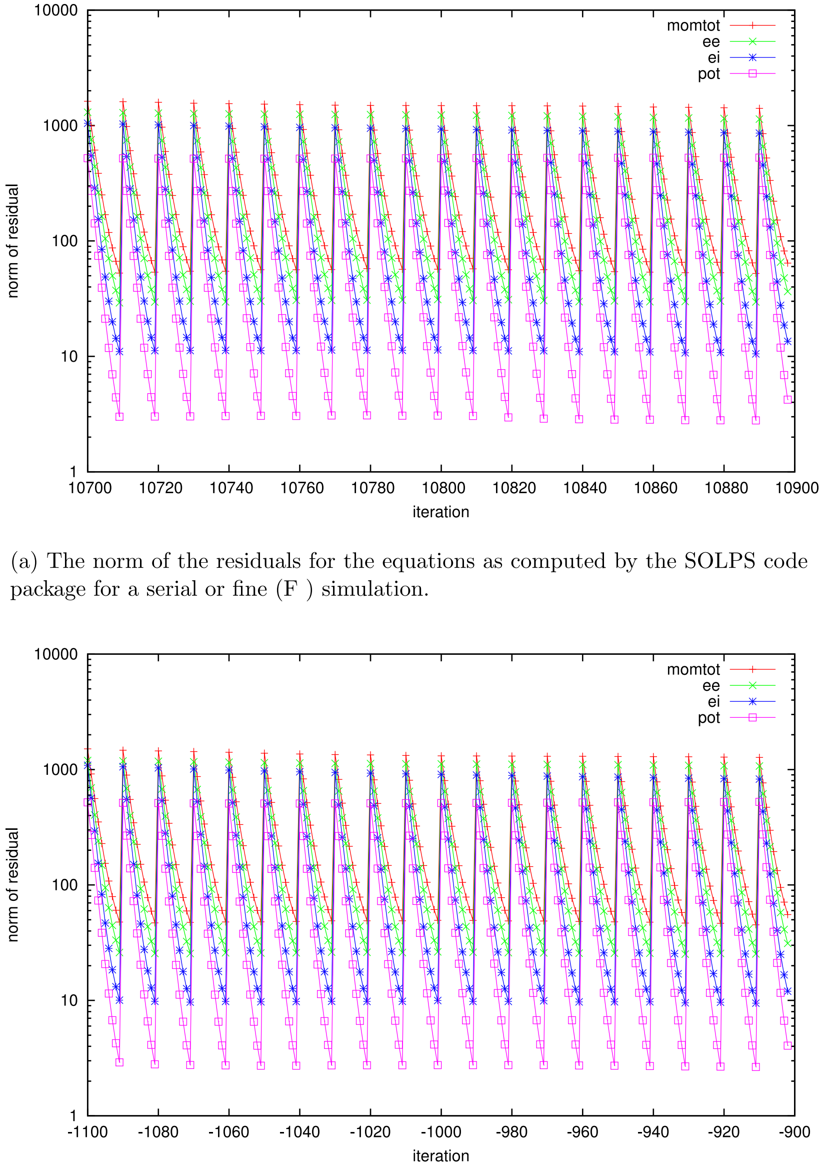
Just as in [8], this paper also utilizes the event-based approach given in detail in [43–45].

The coarse and fine calculations are repeated across a number of parareal iterations k until convergence is achieved. Parareal convergence across a time chunk or processor is said to have been achieved when the 'defect' between two successive fine calculations ($k$ and $k-1$) across that processor is below a tolerance value. The sum of this defect in solutions across a time chunk between $t_i$ and $t_{i-1}$ is defined by

$$\zeta_i^k = \int_{t_{i-1}}^{t_i} \frac{\left|\lambda^k(t) - \lambda^{k-1}(t)\right|}{\left|\lambda^{k-1}(t)\right|} dt. \qquad (3)$$

**Table 1**
Variables used in the text.

| Variable | Description | Units |
|---|---|---|
| nesepm | Electron density at the separatrix on outer midplane | $m^{-3}$ |
| tesepm | Electron temperature at the separatrix on outer midplane | eV |
| pwmxip | Maximum total power flux on inner divertor | $W\,m^{-2}$ |
| pwmxap | Maximum total power flux on outer divertor | $W\,m^{-2}$ |



(a) The norm of the residuals for the equations as computed by the SOLPS code package for a serial or fine (F ) simulation.

(b) The norm of the residuals for the equations as computed by the SOLPS code package for a converged case of the parareal formalism.

**Fig. 4.** The tolerance of the parareal computation is chosen in such a way so that the residual for the parareal calculation agrees with those of the fine (F) simulation.

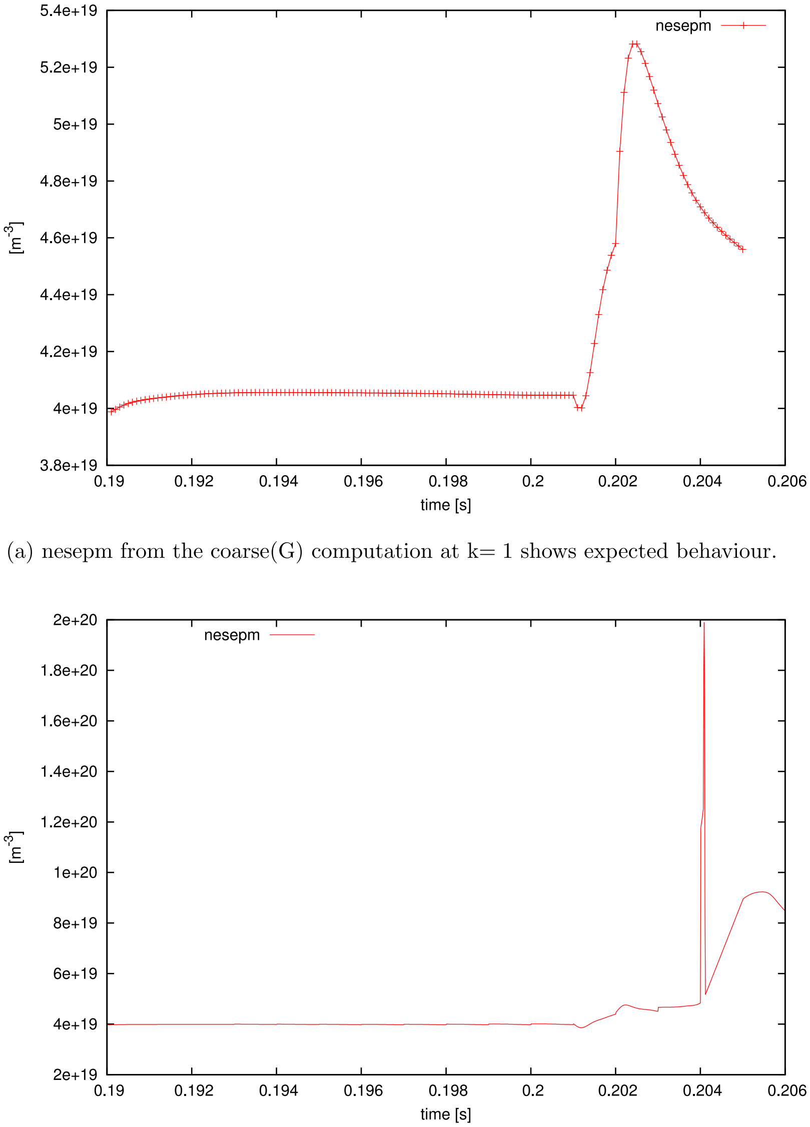
The solution is then converged for time slice $i$ if,

$$\zeta_i^k < \text{tol}, \quad \forall i \leq I. \tag{4}$$

The integration in Eq. (3) is treated as a sum of discrete errors at every point on the time axis, computed by the SOLPS integrator. For the purposes of parareal convergence, Eq. (3) is applied to two variables pwmxap and pwmxip. As already mentioned, pwmxap is the maximum total power flux (in $W.m^{-2}$) on the outer divertor. pwmxip is the same for the inner divertor.

## 5. Results

The cases explored in this work used both the SOLPS5.0 and SOLPS-ITER codes. In some cases, results from SOLPS-ITER were more stable than those from SOLPS5.0. The value of tol in Eq. (4) was 0.005. The choice of this value is based on the residual of the solved equations as described in [8] and is justified by Fig. 4 which illustrates that the residuals of converged parareal solutions and a serial fine solution are comparable.



(a) nesepm from the coarse(G) computation at k= 1 shows expected behaviour.

(b) nesepm from the parareal computation at k= 3 shows anomalous behaviour.

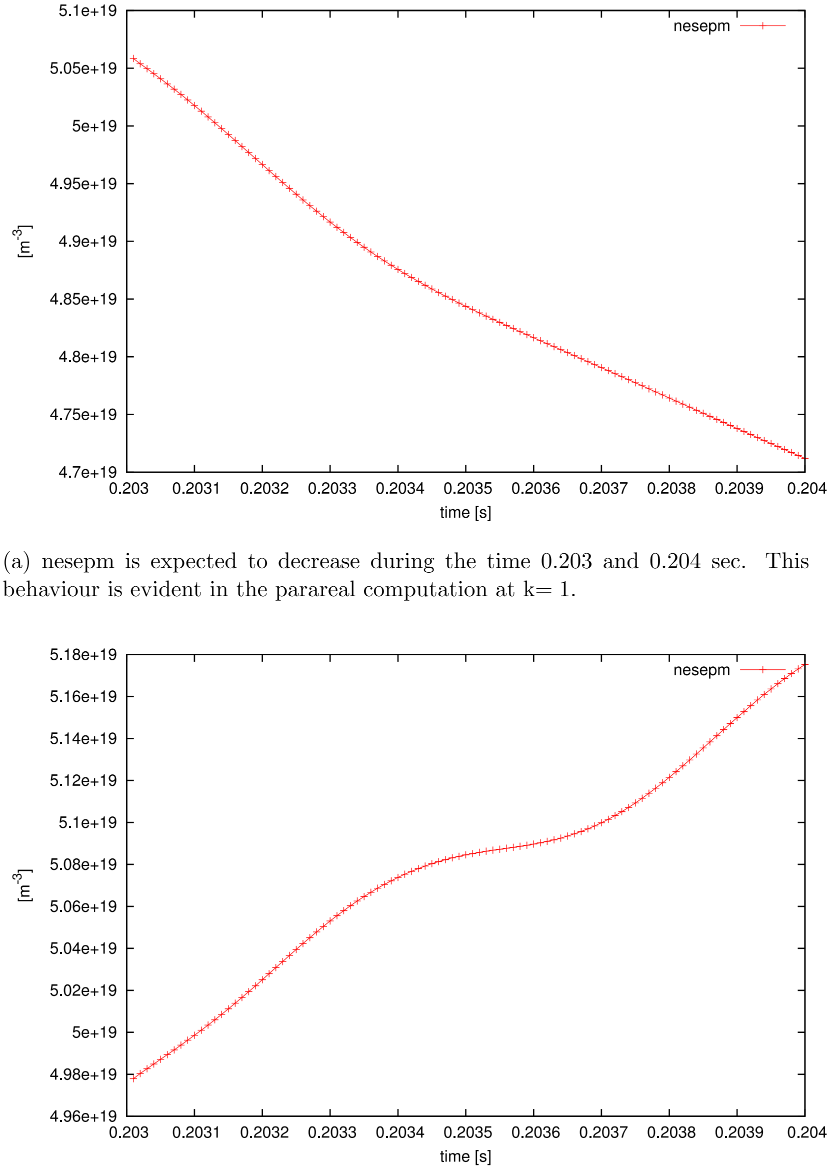
**Fig. 5.** A spurious rise in the electron density at the outer midplane obtained from the parareal computation using a grid of 48 × 36 and $dt_G = 10 * dt_F$ as the coarse solver. The fact that the coarse solution at $k = 1$ is stable indicates the problem is in the parareal method.

Following the work in [8], a reduced grid model of size 48 × 36 with bigger timesteps was used as a coarse predictor for the Parareal implementation. While this approach was relatively straightforward for the cases explored in [8], the treatment turned out to be much more challenging for a plasma with ELMs. The parareal solutions had spurious rises and falls in the plasma quantities such as temperature and density throughout the ELM cycles. One such ‘catastrophic’ case is illustrated in Fig. 5(b) where there is a rise in nesepm, the electron density at the separatrix on the midplane (it must be noted that other quantities like temperature and ion densities also behave similarly) when there is expected to be a decrease as is seen in Fig. 3(a). The fact that there is no unexpected instability in the coarse solution at $k = 1$ as shown in Fig. 5(a) implies that the root of the problem lies in the parareal correction. This was a simulation using 16 processors, with $ntim_F = 100$ and $dt_G = 10 * dt_F$. ntim is the number of timesteps solved per processor and dt represents the size of each time step. The subscripts F and G represent the fine and coarse cases, respectively. It may be noted that $ntim_F * dt_F = ntim_G * dt_G$ is the size of each ’time chunk’ solved on every processor.

A closer look at the parareal computation shows that in this case, the unwanted ELM rise starts at the 14th time chunk, which subsequently affects successive time chunks solved by other processors. Plotting nesepm against time for both $k = 1$ and $k = 2$ in Fig. 6 showed that the rise occurs at the parareal iteration $k = 2$, which somewhat implies that the parareal correction as given by Eq. (2) might be the cause for this behaviour.

A possible explanation might be obtained if we take into account that the radial profiles and the subsequent plasma dynamics vary extremely rapidly during an ELM cycle. This is illustrated for the electron density and temperature along the midplane in Figs. 7 and 8. With the fact that Eq. (2) is an algebraic correction at all grid points, it is not very surprising that after applying Eq. (2), the plasma profiles change to an ELM rise phase while it is expected to be in an ELM crash phase or vice versa.

This issue due to the rapidly varying profiles during an ELM is a cause for concern since the success of the parareal algorithm depends on Eq. (2). However, implementing a restriction to the parareal correction appeared to solve the problem. At the $k$th parareal iteration, the value of each primary variable was not



(a) nesepm is expected to decrease during the time 0.203 and 0.204 sec. This behaviour is evident in the parareal computation at k= 1.

(b) At the parareal iteration k= 2, nesepm starts to increase affecting subsequent time chunks solved using the parareal algorithm eventually leading to the (wrong) solution shown in Fig.5b.

**Fig. 6.** The parareal solutions are very sensitive to initial values in case of ELMs.

allowed to vary more than that of the fine value computed at the $(k-1)$th iteration by $X$%. To ensure that the study was restricted to the correction to the grid points and to rule out any impact of choosing a $dt_G$ greater than $dt_F$, $dt_G = dt_F = 10^{-5}$ s was chosen with a grid of 48 × 36 for the coarse predictor. A series of simulations were performed with different values of $X = 1, 5, 20, 40$. While the undesirable behaviour in the solutions was eradicated with a decreasing value of X, the number of iterations required for parareal convergence was minimum at $X = 1$.

This might raise the issue that the coarse predictor is not very different from the fine solution at all, and in fact, the necessity of the 'fine' integrator may be questioned. It must be noted here that the parareal correction is applied only to the primary variables and the measurement of convergence is based on the maximum total power fluxes (pwmxip and pwmxap) at the divertors. These plasma quantities can vary more significantly with very slight changes (even as small as 1%) to the primary variables. This is illustrated in Fig. 9 where for iteration $k = 1$ across a processor, with the same initial values of primary variables, a variation in pwmxip is observed. The fact that the parareal correction is required to achieve the solution otherwise obtained by a serial, fine run is shown in Fig. 10. Fig. 10 actually illustrates the evolution of the solution through parareal iterations, k with the original coarse solution having a different width for the peak (of pwmxip in this case) as well as a shift along the time axis. A comparison of the serial, fine solution and the parareal computation is discussed later in this paper.

With this promise, the next numerical parameter that was explored was the size of the time chunk solved per processor. This involved varying $ntim_F$ and subsequently varying $ntim_G$, keeping $dt_F$ and $dt_G$ constant. The size of the time chunk has been a factor influencing parareal performances in a large number of previous applications [6,8] and was found to be a strong one in the present case. Once again with restricting the time step sizes such that

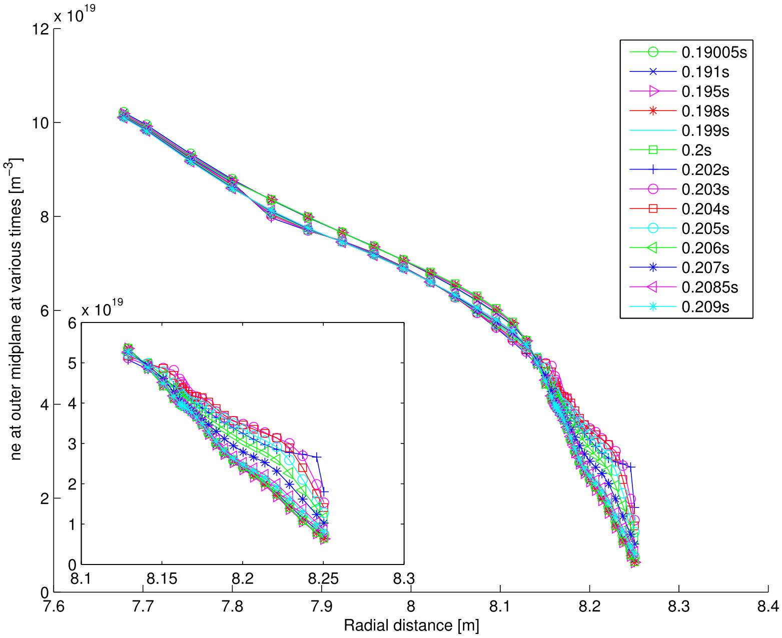


**Fig. 7.** The radial profile of the electron density, ne, very rapidly varies during an ELM cycle. The distance is in metres.

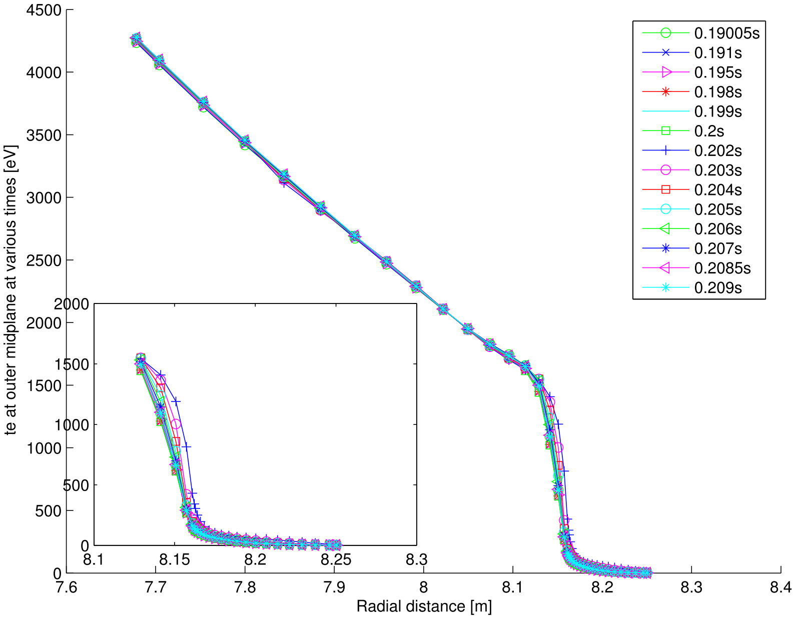


**Fig. 8.** The radial profile of the electron temperature, te, very rapidly varies during an ELM cycle. The distance is in metres.

$dt_G = dt_F = 10^{-5}$ s, a series of simulations were performed with 8 processors or time chunks varying the values of ntim. The results are listed in Table 2. It was observed that $ntim = 200$ allowed the best parareal performance. This dependence on ntim has been observed many times but a full mathematical understanding of the parareal algorithm with respect to it is still unclear. It may be argued that with a large ntim, the coarse solution deviates sufficiently such that the parareal correction can no longer recover the 'correct' solution. Interestingly, with $dtim = 10^{-5}$ s, $ntim = 2000$ actually is the time that includes an entire ELM, that is an ELM rise and crash phases. One might therefore expect, that parareal should be easier to implement when $ntim = 2000$, as most of the radical changes in the plasma dynamics are confined within a time chunk, and the breaks in the time line occur at 'quiet phases' in an ELM cycle. However, it was found that parareal convergence was not at all achieved with chunk size, $ntim = 2000$. This could indicate that the plasma dynamics were still varying rapidly during the 'quiet phase' and with such a long time chunk, the coarse predictor was deviating enough to hinder parareal convergence.

**Table 2**
The number of iterations were minimum at $ntim_F = ntim_G = 200$ with parareal correction restricted to $X = 1$.

| ntim | Parareal iteration k |
|---|---|
| 20 | 4 |
| 200 | 2 |
| 300 | 3 |
| 400 | No convergence |
| 2000 | No convergence |

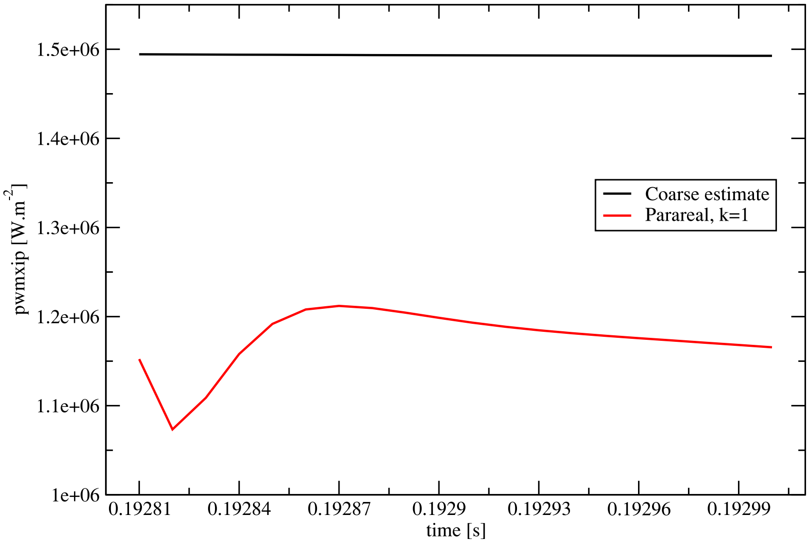


**Fig. 9.** The maximum total power flux at the inner divertor (pwmxip) varies with changes to the grid size and size of the timesteps.

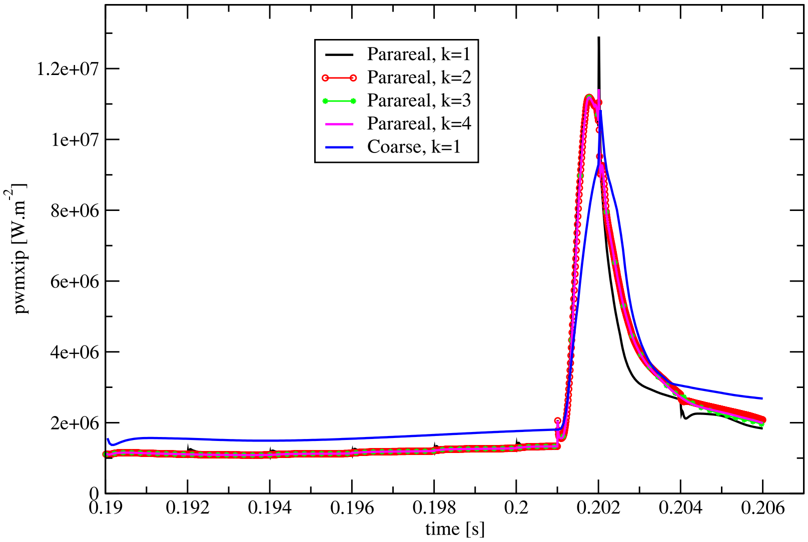


**Fig. 10.** The parareal solution is observed to converge over a number of iterations. The coarse solution has a different position of the peak as well as the width of the ELM is different between the cases.

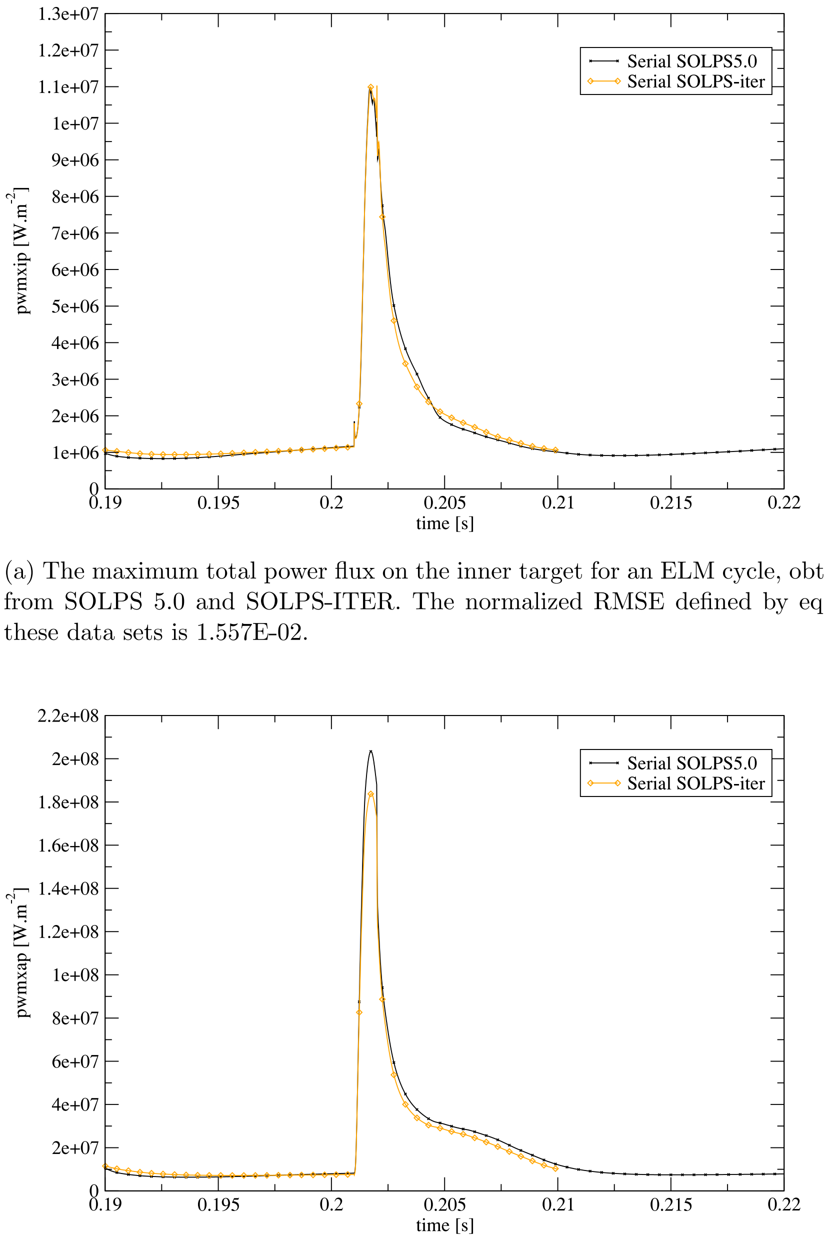
It is believed that SOLPS-like simulations with $dt = 10^{-5}$ s marginally resolve ELM physics with very rapidly varying plasma dynamics. A step size of $10^{-6}$ s is desired. But a serial computation of a single ELM cycle would then take about 16.25 days! However, once the optimum parameters ($ntim_F = 200$ & $X = 1$) were identified, a set of parareal simulations were performed with $dt_F = 10^{-6}$ s & $dt_G = 10^{-5}$ s. With this coarsening of the predictor with respect to the fine integrator resulted in an increase in parareal iteration (k) as in Table 2. The case of $ntim_F = 200$ & $X = 1$ now converged at $k = 3$. With the serial and parareal wall clock times defined as $T_{ser}$ and $T_{PR}$, respectively, the computational gain is defined as $gain = \frac{T_{ser}}{T_{PR}}$. A computational gain of 7.8 was achieved with 48 processors, which is expected to rise with increasing processor counts.

Before analysing the parareal solutions, it may be noted that very slight changes to the integrator or the code itself can result in slight variations in the solutions. This is evident in the results obtained from SOLPS 5.0 and SOLPS-ITER. Although both versions of the code solve the same sets of equations, the solutions (pwmxip and pwmxap) over a complete ELM cycle (2 ms) are illustrated in Fig. 11. The Root Mean Squared Errors (RMSE) for these two datasets are computed. RMSE is defined by the formula given in Eq. (5) and is normalized by Eq. (6), where N is the total number of datapoints, $v_n$ and $u_n$ are the $n$th variables of SOLPS-ITER and SOLPS 5.0, respectively. Comparing solutions from SOLPS 5.0 and SOLPS-ITER, the normalized RMSE for pwmxip is 1.557E−02 and for pwmxap it is 2.0203E−02.

$$RMSE = \sqrt{\frac{\sum_{n=1}^{N}(v_n - u_n)^2}{N}} \tag{5}$$

$$Normalized\ RMSE = \frac{RMSE}{u_{max} - u_{min}} \tag{6}$$

A similar comparison is performed using the parareal solutions with respect to the corresponding SOLPS 5.0 solutions in Fig. 12. As discussed above, although SOLPS 5.0 and SOLPS-ITER solve the same set of equations, variations in the integrator can result in finite, non-zero (although small) RMSE, due to the nature of the problem (ELMs). This fact is evident when comparing the fine, serial solutions with those obtained using the parareal algorithm.



(a) The maximum total power flux on the inner target for an ELM cycle, obtained from SOLPS 5.0 and SOLPS-ITER. The normalized RMSE defined by eq.6 for these data sets is 1.557E-02.

(b) The maximum total power flux on the outer target for the same case as Fig.11a.The normalized RMSE for these data sets is 2.0203E-02.

**Fig. 11.** The solutions obtained from serial versions of SOLPS 5.0 and SOLPS-ITER are compared to one another.

The error bars in Fig. 12 are plotted using the maximum relative error (given by Eq. (7)) between the SOLPS 5.0 and SOLPS-ITER solutions. The normalized RMSE for the parareal computation was determined to be 1.95E−02 for pwmxip and 6.43E−03 for pwmxap.

$$\textit{Relative error} = \frac{v_n - u_n}{u_n} \tag{7}$$

### 5.1. Parareal using the event based method

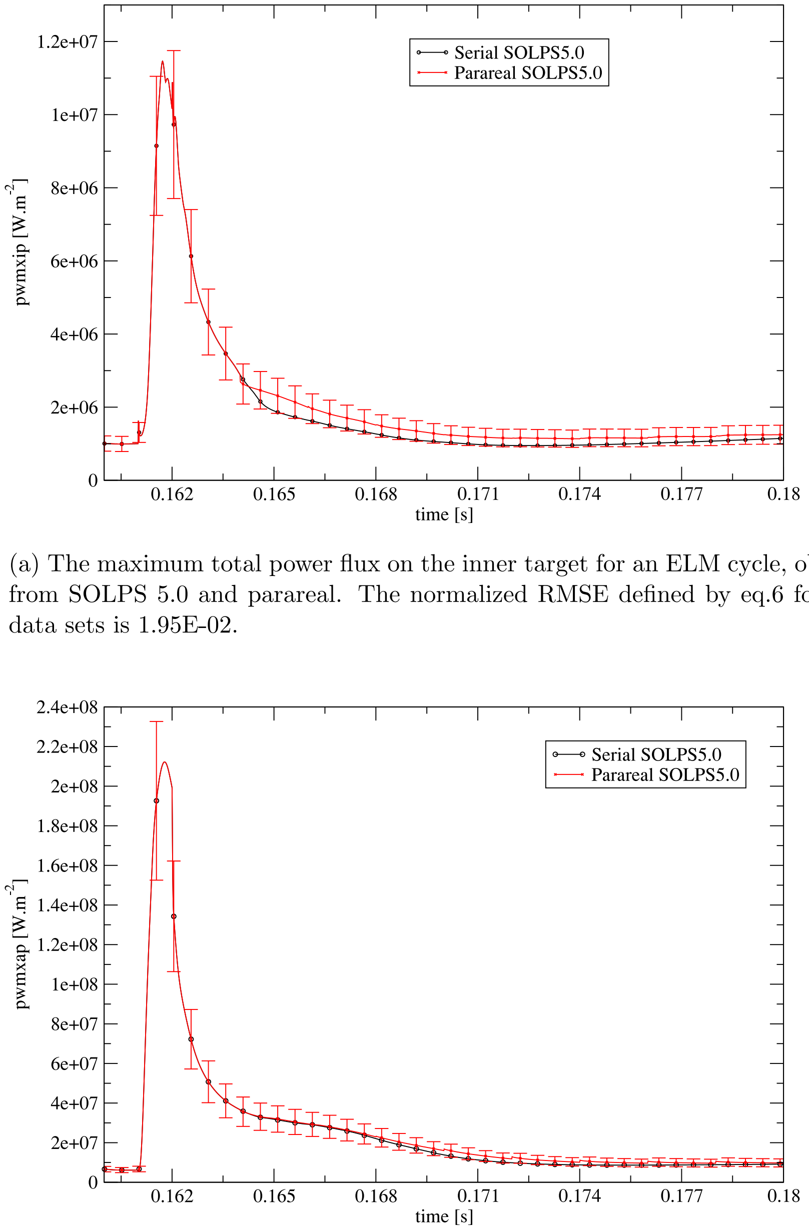
Similar to the work in [8], the event based parareal scheme was employed in this paper using the IPS (Integrated Plasma Simulator) framework [43] & [44]. Following the same approach as in [8] let $t_G$ be the wall clock time for the coarse (G) computation on a single time slice and let $t_F$ be the same for a fine computation also on a single time slice as well. Let N be the total number of time slices.

If at parareal iteration k, $n_c$ slices have converged then $n_k = N - n_c$ slices remain to perform a G and F calculation. In a traditional, sequential MPI implementation of parareal, with G being a serial process and F being performed in parallel, the wall clock time required per iteration k can then be stated as follows:

$$t_k = t_G * n_k + t_F \tag{8}$$

If K is the total number of iterations required for convergence of all N time slices, then summing across all iterations gives the total time for a traditional parareal implementation using MPI.

$$T_{trad} = \sum_{k=1}^{K} t_k \tag{9}$$



(a) The maximum total power flux on the inner target for an ELM cycle, obtained from SOLPS 5.0 and parareal. The normalized RMSE defined by eq.6 for these data sets is 1.95E-02.

(b) The maximum total power flux on the outer target for the same case as Fig.12a.The normalized RMSE for these data sets is 6.43E-03.

**Fig. 12.** The solutions obtained from serial versions of SOLPS 5.0 and the parareal computation are compared. The error bars represent the maximum relative error between SOLPS 5.0 and SOLPS-ITER calculations using a serial processor.

The computational gain for sequential implementation of the parareal algorithm using MPI may then be stated as follows:

$$gain_{trad} = \frac{T_F}{T_{trad}} = \frac{t_F * N}{T_{trad}} \qquad (10)$$

For the case with 48 processors yielding a computational gain of 7.8, the average values for $t_G$ and $t_F$ were $t_G = 922$ s & $t_F = 14040$ s. Using Eq. (10), which ignores communication overhead, the computational gain is 2.425 for a traditional, MPI version of the parareal algorithm.

## 6. Conclusion

The parareal algorithm is shown to work for a complex case of ELM simulations in ITER plasma with 98 species. The rapidly changing radial profiles during an ELM cycle pose unique challenges for this application. A modification to the application of the parareal algorithm alleviates the problem. The parareal performance is also found to be sensitive to the size of the time chunk solved per processor. An optimum value for it is identified, and a simulation with 48 processors yielded a computational gain of 7.8.

This application illustrates that ELM simulations can become more tractable using the parareal algorithm. As a result more complex physics can be incorporated into the model, such as kinetic neutrals, and long simulations of multiple ELM cycles may be performed within much shorter wall clock time.

## Acknowledgements

This work has been funded within the framework of the EUROfusion Consortium and has received funding from the Euratom research and training programme 2014–2018, EU under grant agreement no. 633053. The views and opinions expressed herein do not necessarily reflect those of the European Commission. This work has been part-funded by the RCUK Energy Programme, United Kingdom [grant number EP/I501045]. To obtain further information on the data and models underlying this paper please contact PublicationsManager@ccfe.ac.uk. Also, the views and opinions expressed herein do not necessarily reflect those of the ITER Organization. The computations have been performed on the EUROfusion Gateway Cluster.

## References

[1] C. Farhat, M. Chandesris, Internat. J. Numer. Methods Engrg. 58 (9) (2003) 1397.

[2] J. Lions, Y. Maday, G. Turinici, C. R. Acad. Sci. Ser. I Math. 332 (7) (2001) 661–668.

[3] A.J. Christlieb, B.W. Ong, J.M. Qiu, Math. Comp. 79 (2009) 761–783.

[4] A. Christlieb, C. Macdonald, B. Ong, SIAM J. Sci. Comput. 32 (2) (2010) 818–835.

[5] M. Emmett, M.L. Minion, Commun. Appl. Math. Comput. Sci. 7 (1) (2012) 105.

[6] D. Samaddar, D.E. Newman, R. Sánchez, J. Comput. Phys. 228 (2010) 6558.

[7] D. Samaddar, T. Casper, S. Kim, L.A. Berry, W.R. Elwasif, D.B. Batchelor, W. Houlberg, Appl. Math. Comput. (2018) submitted for publication.

[8] D. Samaddar, D. Coster, X. Bonnin, C. Bergmeister, E. Havlícková, L.A. Berry, W.R. Elwasif, D.B. Batchelor, Comput. Phys. Comm. 221 (2017) 19–27.

[9] L. Baffico, S. Bernard, Y. Maday, G. Turinici, G. Zérah, Phys. Rev. E 66 (5) (2002) 057706.

[10] G. Bal, Y. Maday, A Parareal Time Discretization for Non-linear PDEs with Application to the Pricing of an American Put, in: Lect. Notes Comput. Sci. Eng., vol. 23, Springer, Berlin, 2002, p. 189202.

[11] G. Bal, Parallelization in time of (stochastic) ordinary differential equations, 2003. URL http://www.columbia.edu/~gb2030/PAPERS/paralleltime.pdf.

[12] P.F. Fischer, F. Hecht, Y. Maday, Proceedings of Fifteen International Conference on Domain Decomposition Methods, Vol. 40, Springer Verlag, 2004, pp. 433–440.

[13] M.J. Gander, E. Hairer, Domain Decomposition Methods in Science and Engineering XVII, Vol. 60, Springer, 2008, p. 45.

[14] I. Garrido, M.S. Espedal, G.E. Fladmark, Proceedings of Fifteen International Conference on Domain Decomposition Methods, Vol. 40, Springer, Berlin, 2004, p. 469.

[15] G. Staff, G. Rønquist, Proceedings of Fifteen International Conference on Domain Decomposition Methods, Springer Verlag, 2003, pp. 449–456.

[16] J. Reynolds-Barredo, D.E. Newman, R. Sánchez, D. Samaddar, L.A. Berry, W.R. Elwasif, J. Comput. Phys. 231 (2012) 7851.

[17] A.-M. Baudron, J.-J. Lautard, Y. Maday, M.K. Riahi, J. Salomon, J. Comput. Phys. (2018) submitted for publication.

[18] J. Connor, Plasma Phys. Control. Fusion 40 (5) (1998) 531.

[19] J. Connor, R. Hastie, H. Wilson, R. Miller, Phys. Plasmas 5 (7) (1998) 2687–2700.

[20] D. Coster, A. Chankin, H.-J. Klingshirn, R. Dux, E. Fable, X. Bonnin, A. Kukushkin, A. Loarte, J. Nucl. Mater. 463 (2015) 620–623.

[21] B. Gulejová, R. Pitts, M. Wischmeier, R. Behn, D. Coster, J. Horacek, J. Marki, J. Nucl. Mater. 363 (2007) 1037–1043.

[22] B. Gulejová, R. Pitts, D. Coster, X. Bonnin, M. Beurskens, S. Jachmich, A. Kallenbach, et al., J. Nucl. Mater. 390 (2009) 412–416.

[23] R. Schneider, X. Bonnin, K. Borrass, D. Coster, H. Kastelewicz, D. Reiter, V. Rozhansky, B. Braams, Contrib. Plasma Phys. 46 (1–2) (2006) 3–191.

[24] X. Bonnin, A. Kukushkin, D. Coster, J. Nucl. Mater. 390 (2009) 274–277.

[25] X. Bonnin, W. Dekeyser, R. Pitts, D. Coster, S. Voskoboynikov, S. Wiesen, Plasma Fusion Res. 11 (2016) 1403102–1403102.

[26] S. Wiesen, D. Reiter, V. Kotov, M. Baelmans, W. Dekeyser, A. Kukushkin, S. Lisgo, R. Pitts, V. Rozhansky, G. Saibene, et al., J. Nucl. Mater. 463 (2015) 480–484.

[27] Resonant magnetic perturbation experiments on mast using external and internal coils for elm control.

[28] O. Gruber, A. Kallenbach, M. Kaufmann, K. Lackner, V. Mertens, J. Neuhauser, F. Ryter, H. Zohm, M. Bessenrodt-Weberpals, K. Büchl, et al., Phys. Rev. Lett. 74 (21) (1995) 4217.

[29] G. Huysmans, O. Czarny, Nucl. Fusion 47 (7) (2007) 659.

[30] B. Dudson, M. Umansky, X. Xu, P. Snyder, H. Wilson, Comput. Phys. Comm. 180 (9) (2009) 1467–1480.

[31] C.R. Sovinec, T. Gianakon, E. Held, S. Kruger, D. Schnack, N. Team, Phys. Plasmas 10 (5) (2003) 1727–1732.

[32] B.D. Scott, Phys. Plasmas 12 (10) (2005) 102307.

[33] W. Park, E. Belova, G. Fu, X. Tang, H. Strauss, L. Sugiyama, Phys. Plasmas 6 (5) (1999) 1796–1803.

[34] D. Coster, J. Nucl. Mater. 390 (2009) 826–829.

[35] J. Wesson, Vol. 149, Oxford University Press, 2011.

[36] P.C. Stangeby, The Plasma Boundary of Magnetic Fusion Devices, Vol. 7, Taylor & Francis, 2000.

[37] F. Wagner, G. Fussmann, T. Grave, M. Keilhacker, M. Kornherr, K. Lackner, K. McCormick, E. Müller, A. Stäbler, G. Becker, et al., Phys. Rev. Lett. 53 (15) (1984) 1453.

[38] M. Keilhacker, Plasma Phys. Control. Fusion 29 (10A) (1987) 1401.

[39] F. Wagner, Plasma Phys. Control. Fusion 49 (12B) (2007) B1.

[40] J. Gunn, S. Carpentier-Chouchana, F. Escourbiac, T. Hirai, S. Panayotis, R. Pitts, Y. Corre, R. Dejarnac, M. Firdaouss, M. Kočan, et al., Nucl. Fusion 57 (4) (2017) 046025.

[41] T. Eich, B. Sieglin, A. Thornton, M. Faitsch, A. Kirk, A. Herrmann, W. Suttrop, et al., Nucl. Mater. Energy 12 (2017) 84–90.

[42] A. Loarte, G. Saibene, R. Sartori, D. Campbell, M. Becoulet, L. Horton, T. Eich, A. Herrmann, G. Matthews, N. Asakura, et al., Plasma Phys. Control. Fusion 45 (9) (2003) 1549.

[43] W. Elwasif, D. Bernholdt, A. Shet, S. Foley, R. Bramley, D.B. Batchelor, L.A. Berry, Distributed and Network-based Processing (PDP), Pisa, Italy, 2010.

[44] W.R. Elwasif, S. Foley, D. Bernholdt, L.A. Berry, D. Samaddar, D.E. Newman, R. Sanchez, MTAGS '11 Proceedings of the 2011 ACM International Workshop on Many Task Computing on Grids and Supercomputers, ACM New York, NY, USA, 2011, p. 15.

[45] L.A. Berry, W.R. Elwasif, J. Reynolds-Barredo, D. Samaddar, R. Sanchez, D.E. Newman, J. Comput. Phys. 231 (17) (2012) 5945.