IMPROVED MODELING OF TURBULENT TRANSPORT: FROM NOISE IN TRANSPORT MODELS TO THE PARAREAL ALGORITHM APPLIED TO FULL TURBULENCE CODES

By

Debasmita Samaddar

RECOMMENDED:

[illegible]

[illegible]

[illegible]

[illegible]

Advisory Committee Chair

[illegible]

Chair, Department of Physics

APPROVED:

[illegible]

Dean, College of Natural Science and Mathematics

Lawrence K Duffy

Dean of the Graduate School

April 8, 2010

Date

IMPROVED MODELING OF TURBULENT TRANSPORT: FROM NOISE IN TRANSPORT MODELS TO THE PARAREAL ALGORITHM APPLIED TO FULL TURBULENCE CODES

A

THESIS

Presented to the Faculty

of the University of Alaska Fairbanks

in Partial Fulfillment of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

By

Debasmita Samaddar, B.Sc., M.S.

Fairbanks, Alaska

May 2010

Abstract

Turbulence and turbulent transport are ubiquitous in nature and are of fundamental importance in everything from the spread of pollution to confinement in fusion plasmas. In order to study this, tubulence models need to be as realistic as possible and one must also be able to evolve the turbulence and the profiles of the quantities of interest on transport (long) time scales. Improving turbulence simulations by the introduction of new techniques forms the basis of this research.

One part of this work involved improving the performance of a 1D transport model by the addition of noise.

On a more fundamental level, studying long time dynamics for turbulence simulations is very difficult even with the fastest computers available now or in the near future. To help overcome this difficulty, a new way of simulating turbulence has been presented, namely parallelizing in time. Time parallelization of a fully developed turbulent system is a new application. Parallelizing the space domain to computationally solve partial differential equations has been extensively used and is one of the most common forms of parallelization. In contrast, the Parareal Algorithm parallelizes the time domain and has been found to significantly reduce the computational wall time in many simpler systems. Despite its success in other less complex problems, it has not yet been successfully applied to a turbulent system (to the best of our knowledge). If efficiently applied, this algorithm will allow study of the turbulent transport dynamics on transport time scales - something that has heretofore been very difficult.

In this work, the results of applying the Parareal Algorithm to simulations of drift wave turbulence in slab geometry in which the relative dominance of the polarization and $\mathbf{E} \times \mathbf{B}$ nonlinearities are tuned artificially, are presented. These turbulent systems are in many ways similar to neutral fluid turbulence models, so success of the Parareal scheme in them expands the prospect of a broader range of application to many other turbulent problems.

This thesis also presents the results of a modification to the algorithm. A model to study and predict the parameters governing the convergence of the scheme is also explored.

Table of Contents **Page**

List of Figures

List of Tables

Acknowledgements

A very big thank you to my thesis advisor, David E Newman, for never losing your cool when I did my best to try your patience (and still continue to do so) in innumerable ways. Although you tried to teach me a lot of things and most of them never sank into me, this was the best thing I learnt from you. As a result, I can now retain my composure for a few extra seconds compared to the original few mili seconds.

Another fact about David that I greatly appreciate - you are a great advisor. In fact, the best! The depth of your knowledge of physics, and science in general, as well as your love for it, are admirable. You are not only an exceptional physicist but a remarkably gifted teacher as well, and you make great use of that gift. Without your ideas, inputs, enthusiasm and constant support - this thesis would never have happened. I must make a special mention of the time when my code wasn't working a few days before the APS conference, and I had almost decided I wasn't going despite having registered. It was your unwavering support that got me through the whole thing! I did fix the code (right before getting on the aeroplane), and did print the poster (at the conference venue, on the day of the presentation) and did present it - all of which happened because you never ran short of encouragement. I can cite numerous such anecdotes of David's unending support, but then the acknowledgement might become longer than the rest of the thesis and the grad school might complain.

Another person without whose ideas, support, enthusiasm and encouragement my research would pose to be impossible is Raúl Sánchez. Raúl's unique sense of humor, wonderful knowledge of physics and mathematics, as well as an extremely helpful nature made my PhD research seem much easier than what it actually should have been. Thank you Raúl!

Raúl along with Maria Varela also made my stay at Oak Ridge exceptionally enjoyable! Thank you Raúl for helping me so much with my talk at ORNL.

The rest of my committee - Renate Wackerbauer and Davis Sentman - have also rendered their valuable inputs as well as encouragement, often fighting deadlines alongside me, and I greatly appreciate that! I am also thankful to Renate for helping me practice my presentation before a Journal Club talk, updated versions of which I later presented at a seminar in Oak Ridge National Laboratory, and finally at my defense. I would also like to thank Heinz Wiechen who had originally been on my committee, and with whom I did a semster of independent study when I first arrived in Alaska. In addition to having deep respect for his knowledge as well as his supportiveness, I also have great admiration for his exemplary wit.

Discussions with Benjamin Carreras, Luis Chacon, Lee A. Berry and Jean Noel Leuboff have also been helpful for my research and I acknowledge that.

I am also specially indebted to the consultants and staff at ARSC (Arctic Region Super Computing Center). Don Bahls, Alec Bennett, Lawrence Murakami, Oralee Nudson, Craig Stephenson, Tom Logan, Edward Kornkven, Anton Kulchitsky - thanks for always fixing my ugliest computational problems and never hating me (and even if you did, which was legitimate and normal, you never let me feel it!).

Dave Covey's help with numerous issues of linux administration is gratefully acknowledged.

I am also thankful to John Olson for the support and encouragement.

Thank you Kevin Davis and Marc Mueller-Stoeffer for your help!

I would also like to thank numerous others whose help may not be directly linked to my thesis, but to graduate school life as a whole. I am also appreciative of the fact that I have met so many great people who have made my stay at Fairbanks so delightful. However, both these lists can never be exhaustive! The graduate student body, Robert and Mary Parsons, Saundra Jefko, Barbara Day, Carol Holz, Sue Wolfe, Uma Bhatt, Tilahun are just to name a few.

This section remains incomplete if I do not thank my parents although expressing my gratitude towards them in any way is never enough. My mother, Diba Mitra Samaddar, has always been a great inspiration. You had high hopes for me which I am sure never materialized, and I am to be blamed for that. But, you deserve credit for the fact that I pursued my dreams and am happy. My father, Jyotirmay Samaddar, is because of whom I am writing my thesis today! It was you who first explicitly stated that I should think on my own, and said it was ok if my ideas didn't match yours (and you said it at a time when I was ten years old and had already started thinking most of what you said was wrong and what I thought was correct - you were brave!). A love for puzzles, an appreciation of the fact that mathematics is fun and your outlook towards life in general - I absorbed a lot from you before I knew it! To both my parents - I consider myself fortunate to be your daughter!

A big thank you also to my long and extended family for all the support over the years - mentioning all names is impossible here.

I cannot imagine ending my acknowledgement without mentioning Tapas Bhattacharya. Without your support, patience and friendship, I would never have made it! You almost memorized all my talks and posters and never let me feel sorry for practicing over and over with you. Discussing any topic - be it physics or politics or sports with you - is always so much of an entertainment! Just as you help me with everything, this thesis has been no exception. From making plots in the middle of the night till proofreading - your contribution has been tremendous. Thanks for being there!

Also, a big thank you to all those I missed mentioning here, but you know you helped me - I know that, too.

Chapter 1
Introduction

1.1 Turbulence and turbulent transport

The richness of complexities associated with turbulence gives rise to a lot of intriguing physics, the pursuit of which can be as demanding as exciting. The phenomenon of turbulence is ubiquitous in fluids - be it in the ocean or the atmosphere or the hot core of stars.

A gradient in the temperature or the density in a fluid may be slowly relaxed by classical diffusion. It is predominantly a collisional process with the mean free path determined by the average distance between two successive collisions. All models of classical diffusion are based on the empirical formulation by Fick [1]:

$$\frac{\partial n}{\partial t} = D\frac{\partial^2 n}{\partial x^2} \tag{1.1}$$

where n is the density as a function of space (x) and time (t). D is the diffusion coefficient. Classical diffusion may be described as a random walk process. In cases of high collisional frequency, the mean free path is relatively very small (10^{-6}m in air at sea level), so transport occurs at a very slow rate. Turbulence, on the other hand, has a faster rate of relaxing gradients, characterized by a much larger step size or mean free path, compared to classical diffusion.

It is very difficult to precisely define turbulence, although some characteristics generally prevail [2], [3]. A turbulent system has many degrees of freedom. Its high dimensionally chaotic nature results in a very high sensitivity to initial conditions. Thus, although it is a deterministic system, long time prediction is strongly limited [2]. The Lyapunov exponent, which is a measure of the rate of separation of lagrangian trajectories, is positive for a turbulent system. Turbulence is a nonlinear phenomenon, characterized by a nonlinear parameter, such as the Reynold's number, Rayleigh number, Turbulent Prandtl number, Grashof number, Nusselt number or the inverse Richardson number. A flow becomes turbulent when these parameters exceed certain critical values [2]. One distinctive feature of turbulence is the presence of vortices or eddies of varying sizes. The dynamics of a turbulent fluid is governed by the nonlinearly coupled interactions of these eddies across a huge range of spatial scales. Larger eddies may disintegrate into smaller ones, while eddies of small scale sizes may coalesce to form larger ones [2]. A snapshot of the vorticity field of a turbulence simulation (Hasegawa-Mima model, introduced briefly in Section 1.2 and discussed in Chapter 4) is illustrated in Fig. 1.1. The spectrum of colors imply directions of the vortices, which highlights the randomness of the magnitude and orientation of the spinning of these structures.

It is the transport of particles across these different eddies that constitutes *turbulent transport*. In analogy with classical diffusion, the average eddy size quantifies the step size or mean free path for such a system, which may typically be of the order of centimeters, meters, or even kilometers [4]. Transport across eddies in a turbulent system is illustrated in Fig. 1.2. It is the dominance of nonlinear couplings over linear dynamics that leads to the transition of a laminar flow to a turbulent one. A single model can thus describe both a laminar as well as a turbulent system, depending on the strength of the linear or nonlinear terms.

A typical paradigm for describing turbulence is the Navier-Stokes equation, given by:

$$\frac{\partial \mathbf{v}}{\partial t} + \mathbf{v} \cdot \nabla \mathbf{v} = -\frac{\nabla p}{\rho} + \mu \nabla^2 \mathbf{v} + \mathbf{F} \tag{1.2}$$

where $\mathbf{v}$ is the flow velocity, ρ is the fluid density, p is the pressure, μ is the kinematic viscosity and $\mathbf{F}$ is an external force. The second term on the left hand side of the equation is a nonlinear convection term that is responsible for turbulent features in the system. Other models to describe turbulence exist, which are some variations of Eq. (1.2), that often include additional terms, or modifications of the existing ones, to relate to the physics of particular systems. Two such variations are discussed later in this chapter and in subsequent chapters as well.

One area where turbulent transport becomes the governing factor behind the dynamics of the system, and which is also a major focus of this work, is fusion plasma. Confining extremely hot plasma at the core of the fusion device with much cooler outer walls is still a major challenge. Since nature abhors any gradient turbulence plays a major role in relaxing the gradients. The main challenge to scientists has been to control turbulent transport in fusion devices. Understanding plasma turbulence and turbulent transport is one of the key issues in fusion research.

1.2 Why Fusion?

Fusion has been an active area of research over the last few decades, and has gained fundamental importance in today's society as it promises to provide a largely clean, relatively environmentally friendly solution to our search for an alternate energy source [5]. Making fusion work will be a tremendous triumph for the scientific community as it may not only provide a virtually unlimited source of energy, but will also prove our understanding of many fundamental science issues.

A typical example of a fusion reaction is combining deuterium and tritium nuclei to get a helium nucleus and a neutron, accompanied by a tremendous release of energy, as demonstrated in Eq. (1.3). The combined mass of the helium and the neutron is less than that of the reactants, and the missing mass appears as energy following Einstein's famous relation $E = mc^2$, where E is the energy, m is the mass and c is the velocity of light [6].

$$ {}_1^2D + {}_1^3T \longrightarrow {}_2^4He + {}_0^1n + 17.59MeV \tag{1.3} $$

One of the main barriers in making fusion a viable energy source is fulfilling the Lawson Criterion [7], [8], which requires the triple product $\eta\tau T_e$ to be above a critical value. Here, η is the electron density, τ is the confinement time, and T_e is the plasma temperature. For a typical Deuterium - Tritium reaction, the Lawson criterion requires $\eta\tau T_e \geq 10^{21} keVsm^{-3}$. However, satisfying this condition requires confining the plasma at a few hundred million K at the core for a finite amount of time, while the walls of the device are at a much lower temperature. Needless to say, this generates a huge temperature gradient across the system, thus trigerring turbulent transport processeses, which are believed to play a major role in relaxing the gradients. Different techniques to confine the extremely hot plasma exist, like magnetic confinement, gravitational confinement (as in stars) and inertial confinement. Of these, the first one is discussed in this work.

Magnetic confinement is employed in devices such as tokamaks and stellarators. The charged particles in the plasma are trapped around magnetic field lines. Fig. 1.3 is an example of the magnetic field lines inside a torus. However, understanding turbulent transport across these field lines is critical to the building of a successful fusion reactor.

Turbulent transport in plasmas is more complex than that in neutral fluids, simply due to the presence of charged fluids coupled with electromagnetic fields [9]. Classical diffusion in a magnetized plasma, if described as a random walk process, has a diffusivity, D, given by D $\sim \frac{\rho_e^2}{\tau_e}$, where ρ_e is the electron larmor radius and τ_e is the electron collision time. Neoclassical transport, or classical transport in toroidal devices, has well established theories [10], [11], [12] that are based on classical diffusion, taking into account toroidal geometries. The neoclassical approach describes three distinct regions in the plasma, based on collisionality, namely, (a) the collisional or Pfirsch-Schlüter regime, (b) the banana regime which is relatively collisionless and (c) the plateau regime, which is intermediate between the other two regimes [13].

However, experimental evidence suggests that the electron and ion confinement times in tokamaks are often up to two orders of magnitude shorter than that predicted by neoclassical calculations [9], [13]. A variety of approaches have been undertaken over the years to gain a deeper insight into anomalous transport and the concepts of fractional diffusion [14], [15] or self organized criticality [16] are examples of such attempts. A variety of instabilities have been identified as causes for plasma turbulence, the entire review of which is beyond the scope of the work. However, drift wave instabilities deserve some attention as they provide a platform for the work discussed in Chapters 3 and 4.

A drift wave instability can be caused by a spatial gradient in the distribution of particles in the plasma. [17] states "this instability mechanism can spontaneously convert particle thermal energy into wave energy, and the electromagnetic fields from the waves in turn can cause stochastic motion of the constituent plasma particles." This wave driven instability is what is commonly known as drift wave turbulence. They are characterized by long wavelengths or low frequencies, and are largely responsible for transport across the magnetic field lines [13], [17].

A paradigm to capture drift wave turbulence is the Hasegawa-Mima model [18], given by

$$\frac{\partial}{\partial t}(1-\rho_s^2\nabla_\perp^2)\tilde{\phi} + V_D\frac{\partial\tilde{\phi}}{\partial y} - \rho_s^3 C_s \nabla\tilde{\phi}\times\hat{\mathbf{z}}\cdot\nabla\nabla_\perp^2\tilde{\phi} = \mathbf{0} \tag{1.4}$$

where $\tilde{\phi}$ is the fluctuating potential. $V_D = (\rho_s C_s)L_n^{-1}$ represents the effective diamagnetic drift velocity, where ρ_s is the ion gyroradius, C_s is the sound speed and L_n is the density gradient scale length. The second term on the left hand side of the equation, $V_D\frac{\partial\tilde{\phi}}{\partial y}$, represents the wave term and is responsible for the propagation of the waves in the y direction. The third term on the left hand side of Eq. (1.4) is the nonlinear term arising from the polarization nonlinearity, which may be qualitatively described as the advection of the vorticity. This term is very similar to the nonlinear term in the Navier-Stokes equation for 2-D incompressible fluid (not plasma). This model also holds similarities with the quasi geostrophic equation used in geophysical models [2]. The Hasegawa-Mima model assumes adiabatic electrons, and predicts an inverse cascade of energy, that is, energy from small scales to larger ones [19].

A variation of the Hasegawa-Mima model has been derived in [20], [21], [22] with the addition of another nonlinearity, namely the $\mathbf{E}\times\mathbf{B}$ nonlinearity. This derivation assumes non adiabatic electrons in a dissipative trapped electron mode. The model is given as follows:

$$\frac{\partial}{\partial t}(1-\rho_s^2\nabla_\perp^2)\tilde{\phi}+D\frac{\partial^2\tilde{\phi}}{\partial y^2}+\frac{V_D}{2}\frac{\partial\tilde{\phi}}{\partial y}-\frac{4L_nD}{\varepsilon^{1/2}}\left[\nabla_\perp\left(\frac{\partial\tilde{\phi}}{\partial y}\right)\times\hat{\mathbf{z}}\right]\cdot\nabla_\perp\tilde{\phi}-\rho_s^3C_s\nabla\tilde{\phi}\times\hat{\mathbf{z}}\cdot\nabla\nabla_\perp^2\tilde{\phi}=\mathbf{0}. \tag{1.5}$$

Once again, here, $\tilde{\phi}$ is the fluctuating potential. $V_D=(\rho_sC_s)L_n^{-1}$ represents the effective diamagnetic drift velocity. $D=V_D^2/4\nu_{\text{eff}}$, where ν_{eff} is the effective collision frequency of ion-electron collisions. The second term, $D\frac{\partial^2\tilde{\phi}}{\partial y^2}$, represents the instability drive. The fifth term on the left hand side represents the $\mathrm{E}\times\mathbf{B}$ nonlinearity. This nonlinearity is dominant at large wavelengths and predicts direct cascade of energy, i.e, energy transfer from large scales to smaller scales. Drift wave models with the polarization nonlinearity or the $\mathbf{E}\times\mathbf{B}$ nonlinearity or both, are used in Chapters 3 and 4.

While the building of the International Thermonuclear Experimental Reactor (ITER) (Fig. 1.4) [23], [24] is in progress, a multitude of experiments are being conducted around the world in numerous tokamaks and stellarators (JT-60SC [25], Joint European Torus (JET) [26], [27], D-III D [28], Helically Symmetric Experiment (HSX) [29], Large Helical Device (LHD) [30], [31], [32] and more) involving magnetically confined hot plasmas, characteristic of fusion. (Fig. 1.4) is a cross sectional view of ITER, to become the largest tokamak in the world once it is built. The cross section of the toroidal space at the center is where the plasma is to be confined for fusion. All the experiments with different tokamaks and stellarators have the common goal of controlling turbulent transport and increasing confinement times. Although significant progress has been made in understanding plasma transport in the last few decades, a complete picture is still unavailable [3].

1.3 Numerical approach

Numerical simulations serve as an important tool to understand the underlying physics of all areas in modern scientific research [33], and fusion is no exception. Both fluid and particle codes exist for simulating turbulent transport, although only the former is explored in this work. As far as fluid codes are concerned a variety of them are used by the fusion community depending on the regime explored and the outcome desired. Each kind has its own strengths and weaknesses.

One dimensional codes, which typically represent a reduced physics, require a lot less computational wall-clock time compared to the other models. They are envelope models and can be run in real time with real parameters, allowing exploration of parts of the dynamics. Hence, having realistic dynamics is essential for these simulations. Both 1D and 2D models allow study of how transport reacts to parameter regimes. 2D and 3D models may be based on primitive equations, but are computationally more complicated than their 1D counterpart.

A numerical study of turbulence is very intensive in terms of computational wall clock time. This is particularly due to the fact that turbulence involves nonlinear interactions between a wide range of spatial scales, some of which cannot be resolved by the simulations. The most common technique is to use an "effective diffusivity" for scales not resolved by the model. Although this procedure may make the simulations feasible, they may leave out important physics by ignoring the dynamics of the smaller scales [20].

Improving the numerical techniques for the study of turbulence and turbulent transport serves as the main theme of this research. The work starts off by modifying an existing 1D model to obtain plausible solutions closer to observations in real experiments. The major concentration of this thesis is on introducing a whole new technique, known as the parareal algorithm, to simulations of turbulence.

The following few sub-sections serve as an introduction for the numerical techniques and the physical problems discussed in subsequent chapters.

1.3.1 1D envelope model

The 1D model explored as part of this research in Chapter 2 is a reaction-diffusion model [34], [35]. Prior to this work, the model had been extensively applied to study transition dynamics in turbulent plasma. The model studies the evolution of the density fluctuation, ε in the radial direction, r. The model is primarily based on ion temperature gradient or ITG driven instabilities at the plasma core and resistive ballooning mode turbulence at the plasma edge. The suppression of the turbulence in the model is achieved by a sheared $\mathbf{E} \times \mathbf{B}$ flow in the system. This sheared flow is produced by a gradient in the plasma pressure, which is also a source for driving the instabilities. Therefore, the pressure gradient serves as a drive for the turbulence, as well as a cause for its suppression.

The model generated plausible simulations for the Tokamak Fusion Test Reactor [36]. The simulations showed oscillations in the density fluctuations in the system. Fig. 1.5 is a contour plot of the fluctuation of the density profiles, evolving with time. The waves are represented by the structures observed in the region in space approximately between $r = 80$ and $r = 55$ (r being the radial distance and a being the total length). In the plot, time increases in a top-down fashion, and the radial distance increases from left to right. The structures curving downward towards the right thus indicate that the waves are moving radially outward with increasing time. However the waves as shown in Fig. 1.5 had a realistic presence, but propagated over distances that were unphysical.

In an attempt to reduce the length of propagation of these oscillations, noise was added to the system. The fact that noise has had interesting effects in other reaction-diffusion systems served as a motivation [37], [38], [39]. It has commonly been observed in [37], [38], [39] that the addition of noise to the numerical simulations helped by decorrelating the waves although noise enhanced propagation has been reported in given regimes. Oscillations induced by noisy perturbations in stocahstic systems has been observed and studied in various systems [40], [41],

[42]. It is possible, that when the noise induced oscillation is in resonance with the waves in the system, noise enhanced propagation is observed. The reverse may occur, when the noise induced wave and the wave inherent to the system are in anti-phases with each other, giving rise to decorrelation. The model used in this work, although it deals with plasma turbulence, gave results similar to other systems in terms of the behavior of noise as a decorrelation surrogate.

1.3.2 Numerical challenges in turbulence modeling

The computational approach to solving problems is becoming increasingly promising, as well as challenging, with the advent of even more powerful supercomputers. Capturing the dynamics of turbulent transport in numerical simulations is an example where the present computational techniques still need a dramatic improvement. In an attempt to capture the transport dynamics, the simulations would be required to resolve the smallest eddies, which would involve timescales that are atleast $10^6 - 10^9$ times smaller than the plasma confinement time! An ideal study of turbulence would thus require evolving the system over thousands of eddy decorrelation times in the presence of a self consistently evolving profile. The eddy decorrelation time τ is defined as the average time in which an eddy disintegrates in a turbulent fluid. It may be mathematically defined as the half width at half maximum (HWHM) of the power spectrum.

However, such a task is currently impossible even in the most powerful supercomputers due to the extremely long wall-clock times required for these calculations [43]. Simplified approaches involve allowing the turbulence to evolve over a few tens of decorrelation times, with the plasma profiles assumed to be frozen. One way to achieve the desired extent of these simulations would be to utilize parallel computation techniques available on supercomputers. However, even the state of the art fluid codes currently used by the fusion community only scale up to a few thousand processors, beyond which, increasing the number of processors does not contribute to any significant speedup. It should be mentioned here, that space parallelization has been the most common technique to solve the partial differential equations that characterize turbulent systems.

Since modern supercomputers have hundreds of thousands of processors and the current simulations (with fluid codes) with space parallelization can only use a few thousand of those there is room for improvement. That is where the idea of time parallelization introduces a whole new frontier. Combining space and time parallelization may allow the maximum utilization of the resources available on supercomputers.

1.3.3 Techniques for time parallelization and the parareal algorithm

The concept of parallelizing the time domain was proposed in 1964 [44], long before extensive use of parallel processors was in vogue. This idea was further pursued by [45] as the multiple shooting method for boundary value problems, where a time interval is divided into smaller intervals, and each subinterval is solved in parallel with continuity of the solutions achieved by the Newton method. Since then, the quest for new techniques to attain temporal parallelization has continued.

The parareal technique, the prime focus of this work, was first presented in the seminal paper by *Lions et al.* [46] in 2001. The algorithm was presented in more detail, accompanied by applications to simple problems such as molecular dynamics in [47] and nonlinear partial differential equations (pdes), namely the Black-Scholes equations in [48]. A number of publications have explored the stability and convergence of the new algorithm. For example, [49] presented a study of the stability of the parareal technique. The stability and convergence properties for time dependent equations with constant coefficients was analyzed in [50]. Convergence properties of the algorithm for bounded and unbounded problems were more recently explored in [51].

Over the last decade, the parareal algorithm has been applied to many more problems, ranging over a variety of physics involved. [52] illustrated an application to reservoir simulations, and the technique was tested in the laminar region of the Navier-Stokes equations in [53]. The study was extended to stochastic ordinary differential equations in [54], and also to other nonlinear problems such as the Viscous Burgers equation and the Lorenz equations, accompanied by convergence analysis, in [55]. A modest success with the algorithm was reported for the Princeton Ocean model, where simple convection was the dominant factor [56].

However, prior to this work, as discussed in chapter 3, this algorithm has, to the best of our knowledge, never been successfully applied to a system with fully developed turbulence. This application is an important extension, because the results are not only non-intuitive, but to some extent unexpected as well.

The parareal technique is unique because it parallelises an initial value problem along time, despite the sequential nature of the domain. With this algorithm, different slices of time of a long time series are solved simultaneously on parallel processors, using a predictor-corrector approach. The technique employs a coarse solver and a fine solver. The coarse solver is called G, and the fine solver F, in this chapter and the rest of the thesis.

Although the algorithm is reviewed in detail in Chapter 3, two diagrams, Fig. 1.6 and Fig. 1.7 describing the technique, are presented here. In Fig. 1.6, energy is a function of the time dependent variable λ that is updated at every parareal iteration, represented by k, using the parareal algorithm. The entire time series (along the x axis), is broken into pieces, and each piece is separately solved on a different processor, represented by P0, P1, P2, ... and so on. The value of energy at the initial time, t0, is given. There are two solvers, F and G, than can propagate the solution and allow the determination of the value of energy at some later time, like, t1, t2, t3, ... and so on. Of these, F is said to be a fine solver, giving an accurate solution to the given problem. F is computationally slow, and is always used in parallel. G is the coarse solver, and, as its name suggests, its solutions lack in accuracy. However, G is much faster computationally, and is employed as a serial procedure.

The black line (denoted as "Actual") in Fig. 1.6 is the converged solution that is finally attained by the parareal algorithm. The solid red line represents a serial run with G, and it is the first step for the parareal scheme. This run gives each processor an inaccurate initial value. This step is followed by an F run (green dotted curves) in parallel. The initial value used by each processor is the value generated in the previous step. However, only processor P0 uses the correct initial value, and as F is an accurate propagator, P0 arrives at the correct solution in this step. This step is followed by another serial, but discontinuous, calculation with G (solid pink line). Since the solution along P0 is already the correct one, this step starts from

P1. The initial value of the G run at subsequent processors is calculated using the parareal algorithm (the formula on Fig. 1.6). The G run is followed by an F run, and this time, too, the initial value is calculated by the same procedure as in the previous G run. Calculations involving the coarse and fine solvers thus alternate, until convergence is achieved. For a successful implementation of the algorithm, this convergence needs to be attained relatively fast, in as less parareal iterations as possible.

Fig. 1.7 is a more schematic representation of the parareal algorithm. In Fig. 1.7, each processor, solving the i^{th} time chunk is represented by a block. The diagram gives a schematic overview of the algorithm for different parareal iterations, k, denoted as a superscript for the variable λ. The subscript, i, for λ, represents the time or more accurately, the processor solving the i^{th} time chunk. So, λ_i^k reprsents the value of λ in the i^{th} time, at the k^{th} parareal iteration. $\lambda_0^0 = y_0$ is known. Details of the error calculation and convergence are given in Chapter 3.

The system where we have applied this novel technique consists of a simple drift wave turbulence model, extensively studied in [20], [21], [22], [57], [58]. Once it has been successfully applied here, it is rather straight forward to extend it to other turbulent systems. The model used here is a dissipative trapped electron mode (DTEM) turbulence model in a doubly periodic slab geometry (x-y) with the $\mathbf{E} \times \mathbf{B}$ nonlinearity or the polarization nonlinearity or a combination of both being present. The model has been introduced in Section 1.2, and more details about this model are available in Chapters 3 and 4.

The application of the parareal algorithm to a turbulent system has proven to be extremely difficult due to the exponential divergence of Lagrangian trajectories and extreme sensitivity to initial conditions (chaos). However, although non-trivial, the work described in Chapter 3 and 4 adapted it successfully for simulating typical turbulences in plasmas, proving that a significant computational gain (close to an order of magnitude) can be achieved in a turbulent system using this algorithm.

1.4 An overview of the thesis

The rest of this thesis consists of Chapters 2, 3, 4 and 5. Of these, Chapters 2, 3 and 4 are either prepared for submission or have been submitted to different journals. Chapter 2 describes the effects of adding noise to the simulation of a 1D transport model, briefly described in this chapter in Section 1.3.1. The model used for the 1D simulation is described in detail in Sec. 2.2. The technique and results are given in Sections 2.3 and 2.4.

Chapter 3 is about the first application of the parareal technique to fully developed turbulence. A brief description of the parareal algorithm has been given in Section 1.3.3. More details of the algorithm and the results constitute Chapter 3. Chapter 4 gives two additional examples of the application of the parareal technique to fully developed turbulent systems. Chapter 4 further explores the characteristics of convergence described in Chapter 3. It also gives an account of the various coarse solvers that have been explored for the application of the parareal algorithm to turbulence. Chapter 5 contains the concluding remarks about the entire work, consisting of a summary and a description of the scope for future work.

1.5 Statement of work

Chapter 2 of this thesis is a result of numerous discussions with my thesis advisor David E Newman. The work had been started by him and John Broussard before I joined in. So, some of the figures (namely Figs. 2.1, 2.4, 2.5 and 2.6) in Section 2.2 where the groundwork for the research is discussed, were generated by either David or John. The rest of the plots in Chapter 2 were results of simulations run by me, with David's inputs, of course. The simulations were completed on local linux computers as well as the ARSC (Arctic Region Supercomputing Center, Fairbanks, Alaska) super computers.

Chapter 2 is prepared for submission in the Physics of Plasmas. Along with my co-authors, I have presented the work, at its various stages, at the following conferences:

(i) Dynamics of Complex Systems Meeting, July 5 - July 7, 2006; Fairbanks, Alaska.
(ii) APS - 48th Annual Meeting of the Division of Plasma Physics, October 30 - November 3, 2006; Philadelphia, Pennsylvania.

(iii) 12th US-EU Transport Taskforce Workshop, April 17 - April 20, 2007; San Diego, California.
(iv) Dynamics of Complex Systems Meeting, July 25 - July 27, 2007; Fairbanks, Alaska.
Apart from David and me, Chapters 3 and 4 resulted from various ideas proposed by Raúl Sánchez. The model introduced in Section 3.5.2 and later discussed in Chapter 4 was Raúl's idea. The various techniques adapted for the simulations described in these chapters were discussed between all of us, and implemented by me in the code. The simulations for these chapters were run by me on the ARSC super computers.

Chapter 3 has already been submitted to the Journal of Computational Physics [59]. Chapter 4 is prepared for submission in the Physics of Plasmas. The work discussed in these two chapters have also been presented by me and my co-authors at the following conferences:
(i) Dynamics of Complex Systems Meeting, August 6 - August 8, 2008; Fairbanks, Alaska.
(ii) APS - 50th Annual Meeting of the Division of Plasma Physics, November 17 - November 21, 2008; Dallas, Texas.
(iii) Joint EU-US Transport Taskforce Workshop, April 28 - May 1, 2009; San Diego, California.
(iv) APS - 51st Annual Meeting of the Division of Plasma Physics, November 2 - November 6, 2009; Atlanta, Georgia.
(v) Dynamics of Complex Systems Meeting, March 8 - March 11, 2010; Fairbanks, Alaska.
This work is to be presented at the following forthcoming conferences:
(i) US Transport Taskforce Workshop, April 13 - April 16, 2010; Annapolis, Maryland.
(ii) International Sherwood Fusion Theory Conference, April 19 - April 21; Seattle, Washington.
(iii) EPS - 37th Conference on Plasma Physics, June 21 - June 25, 2010; Dublin, Ireland.
In addition, I have also presented this research at the following seminars:

(i) UAF/GI Physics Journal Club, October 9, 2009; Fairbanks, Alaska.

(ii) Oak Ridge National Laboratory, December 10, 2010; Oak Ridge, Tennessee.

The first draft for all Chapters 1, 2, 3, 4 and 5 were written by me. However, with innumerable inputs and suggestions from not just David and Raúl, but from the rest of my committee as well, particularly Renate Wackerbauer, they now look very different from their original versions!

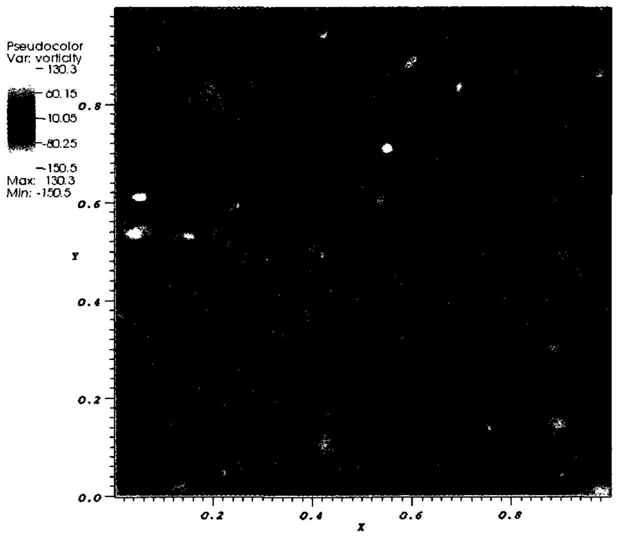


Figure 1.1. Numerical simulation of the vorticity field of a turbulent fluid displays eddies of different scale sizes.

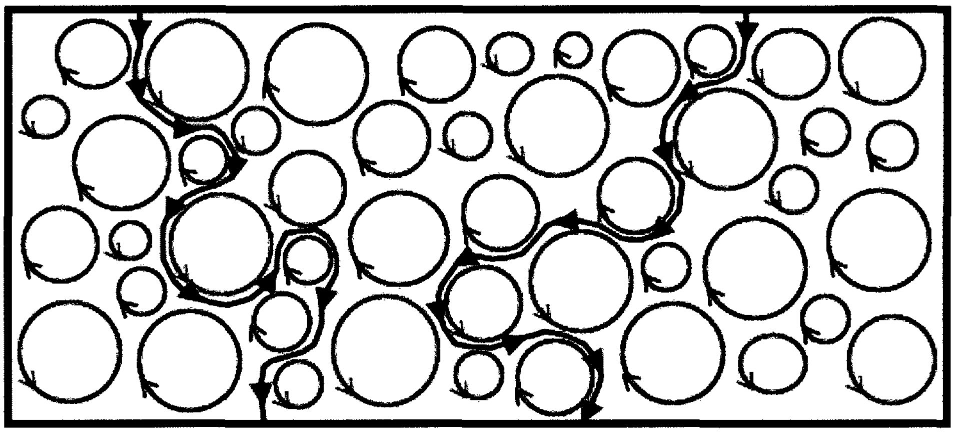


Figure 1.2. Transport across eddies in a turbulent system leads to transport coefficients much larger than classical diffusion.

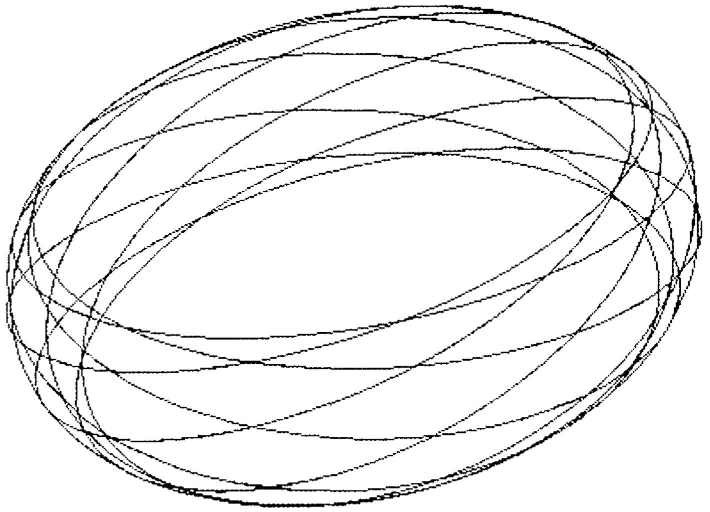


Figure 1.3. A diagram of the magnetic field lines inside a device such as the tokamak.

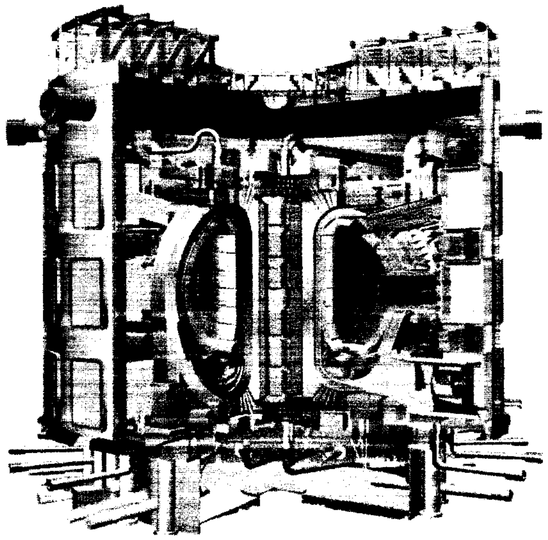


Figure 1.4. A diagram of the interior of ITER, an international project, currently under construction in France. *Ref* : *http* : //*fusion.gat.com*.

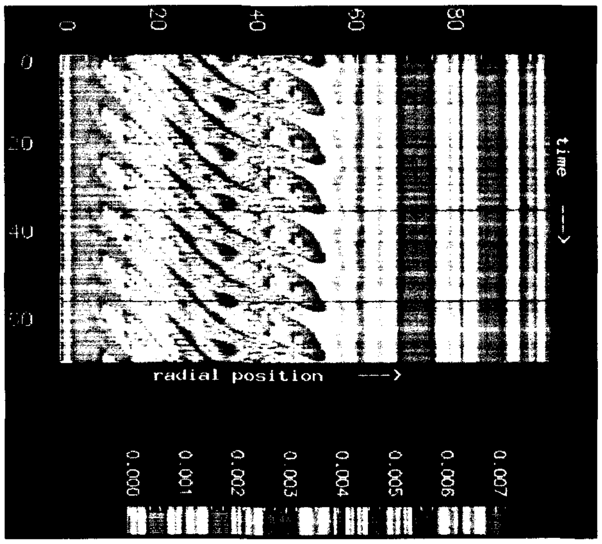


Figure 1.5. Numerical simulation of the 1D reaction-diffusion model shows the presence of waves that travel unphysical distances.

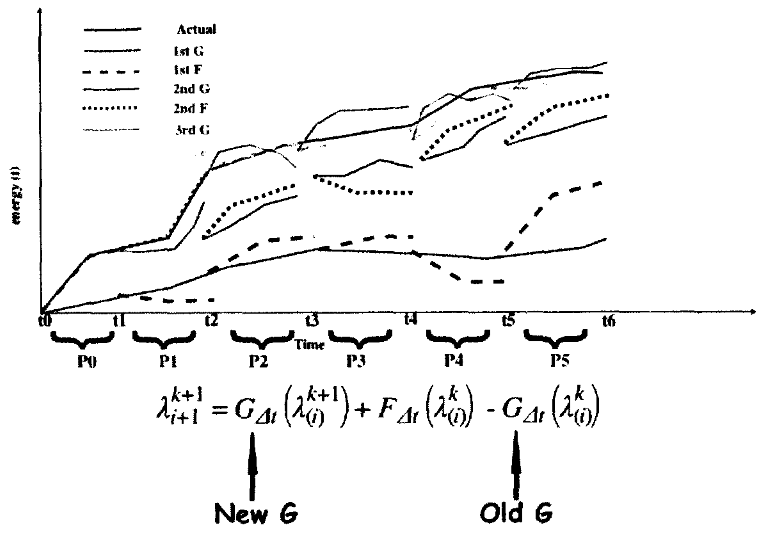


Figure 1.6. G and F are the coarse and fine solvers, respectively. Energy, along the y-axis, is a time dependent function for the system.

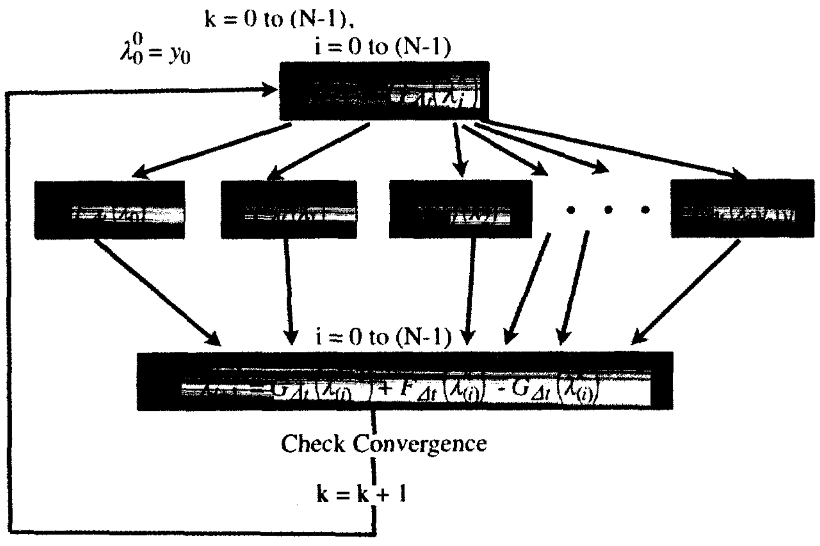


Figure 1.7. Each block represents a single processor and k is the parareal iteration. λ_i^k is the variable at the i^{th} time step, for the k^{th} iteration.

Bibliography

[1] A. Fick, Phil. Mag. 10 (1855), pp. 3039.

[2] P. K. Kundu and I. M. Cohen, Fluid Mechanics, Elsevier Academic Press, (2004).

[3] U. Frisch, Turbulence, The Legacy of A. N. Kolmogorov. Cambridge, U.K.: Cambridge Univ. Press, (1995).

[4] D. E. Newman, P. W. Terry, A. S. Ware, Shear flows and turbulence in nature, Computing in Science and Engineering, vol. 9, no. 6, (2007) pp. 45-52.

[5] J. Ongena and G. V. Oost, Trans. Fusion Sci. Technol., 41, (2002) 314.

[6] A. Einstein, Ist die Trägheit eines Körpers von seinem Energieinhalt abhängig?, Annalen der Physik 18: (1905) 639641.

[7] J. D. Lawson, Proceedings of the Physical Society B, Vol. 70, (1957) p. 6.

[8] J. Hugill, Nucl. Fusion 25, (1983) 331.

[9] B. A. Carreras, Progress in anomalous transport research in toroidal magnetic confinement devices, IEEE Trans. Plasma Sci. 25, (1997) 1281.

[10] A. A. Galeev and R. Z. Sagdeev, Transport phenomena in a collisionless plasma in a toroidal magnetic system, Sov. Phys.JETP, vol. 26, (1968) pp. 233240.

[11] A. A. Galeev, R. Z. Sagdeev, H. P. Furth, and M. N. Rosenbluth, Plasma diffusion in a toroidal stellarator, Phys. Rev. Lett., vol. 22, (1969)pp. 511514.

[12] F. L. Hinton and R. D. Hazeltine, Theory of plasma transport in toroidal confinement systems, Rev. Mod. Phys. 48, (1976) 239.

[13] J. Wesson, Tokamaks, Oxford University Press, (1987).

[14] D. del Castillo Negrete, B. A. Carreras and V. E. Lynch, Fractional diffusion in plasma turbulence, Phys. Plasmas 11, (2004) 3854.

[15] R. Sánchez, B. A. Carreras, D. E. Newman, V. E. Lynch and B. Ph. van Milligen, Renormalization of tracer turbulence leading to fractional differential equations, Phys. Rev E 74, (2006) 016305.

[16] B. A. Carreras, D. E. Newman, V. E. Lynch and P. H. Diamond Phys. Plasmas 3 (1996) 2903-11.

[17] W. Horton, Drift waves and transport, Rev. Mod. Phys. 71, (1999) 735778.

[18] A. Hasegawa and K. Mima, Phys. Rev. Lett. 39, (1977) 205.

[19] A. Hasegawa, C. G. Maclennan and Y. Kodama, Phys. Fluids 22, (1979) 2122.

[20] D. E. Newman, P. W. Terry, P. H. Diamond, A two-nonlinearity model of dissipative drift wave turbulence, Phys. Fluids B 4 (3) (1992) 599-610.

[21] Y. M. Liang, P. H. Diamond, X. H. Wang, D. E. Newman, P. W. Terry, A two-nonlinearity model of dissipative drift wave turbulence, Phys. Fluids B 5 (4) (1993) 1128-1139.

[22] D. E. Newman, P. W. Terry, P. H. Diamond, Y. M. Liang. The dynamics of spectral transfer in a model of drift wave turbulence with two nonlinearities, Phys. Fluids B 5 (4) (1993) 1140-1153.

[23] 1992 ITER EDA Agreement and Protocol 1, ITER EDA Documentation Series No. 1 (IAEA, Vienna).

[24] Y. Shimomura et al. Nucl. Fusion 39 (1999) 1295.

[25] Objectives and design of the JT-60 superconducting tokamak, S. Ishida et al Nucl. Fusion 43 (2003) 606.

[26] A. Gibson, Fusion relevant performance in JET, Plasma Phys. Control. Fusion 32 (1990) 1083.

[27] Fusion energy production from a deuterium-tritium plasma in the JET tokamak, JET Team Nucl. Fusion 32 (1992) 187.

[28] A design retrospective of the DIII-D tokamak, J. L. Luxon Nucl. Fusion 42 (2002) 614.

[29] Overview of recent results from HSX, D. T. Anderson, A. Abdou, A. F. Almagri et al., Fusion Sci. Technol. 50, 171 (2006).

[30] Design study for the large helical device, A. Iiyoshi et al., Fusion Technology; (USA) (1990).

[31] Initial physics achievements of large helical device experiments, Phys. Plasmas 6, (1999) 1843.

[32] Overview of the Large Helical Device project, A. Iiyoshi et al. Nucl. Fusion 39 (1999) 1245.

[33] S. Steven, "The End of Insight", in Brockman, John, What is your dangerous idea?, Harper Collins (2007).

[34] D. E. Newman, B. A. Carreras, D. Lopez Bruna, P. H. Diamond, and V. B. Lebedev, Dynamics and control of internal transport barriers in reversed shear discharges, Phys. Plasmas 5, (1998) 938.

[35] D. Lopez Bruna, D. E. Newman, B. A. Carreras and P. H. Diamond, Fluctuation level bursts in a model of internal transport barrier formation, Phys. Plasmas 6, (1999) 854.

[36] G. Bateman, Phys. Fluids B, 4, (1992) 634.

[37] S. Kadar, J. Wang, and K. Showalter, Nature (London) 391, (1998) 770-772.

[38] J. F. Lindner, S. Chandramouli, A. R. Bulsara, M. Locher, and W. L. Ditto, Phys. Rev. Lett. 81, (1998) 5048-5051.

[39] C. Zhou, J. Kurths, Z. Neufeld, and I. Z. Kiss, Phys. Rev. Lett. 91,(2003) 150601.

[40] M. Sieber, H. Malchow and L. Schimansky-Geier, Constructive effects of environmental noise in an excitable preypredator plankton system with infected prey, Ecol. Complex. 4, (2007) 223233.

[41] H. Treutlein and K. Schulten, Noise-induced neural impulses. Eur. Biophys. J. 13 (6), (1986) 355365.

[42] M. A. Zaks, A. B. Neiman, S. Feistel and L. Schimansky Geier, Noise controlled oscillations and their bifurcations in coupled phase oscillators. Phys. Rev. E 68, (2003) 066206.

[43] J. Dahlburg, J. Corones, D. Batchelor, R. Bramley, M. Greenwald, S. Jardin, S. Krasheninnikov, A. Laub, J. -N. Leboeuf, J. Lindl, W. Lokke, M. Rosenbluth, D. Ross, D. Schnack, Fusion simulation project: Integrated simulation and optimization of magnetic fusion systems, J. Fus. Energy 20 (4) (2001) 135.196.

[44] J. Nivergelt, Parallel methods for integrating ordinary differential equations, Commun. ACM 7 (12) (1964) 731733.

[45] H. B. Keller, Numerical Methods for Two-Point Boundary Value Problems, Blaisdell, Waltham, MA, (1968).

[46] J. Lions, Y. Maday, G. Turinici, A "parareal" in time discretization of pde's, Comptes Rendus de l'Academie des Sciences - Series I - Mathematics 332 (7) (2001) 661-668.

[47] L. Baffico, S. Bernard, Y. Maday, G. Turinici, G. Zerah, Parallel in time molecular dynamics simulations, Phys. Rev. E 66 (5) (2002) 057706.

[48] G. Bal, Y. Maday, A parareal time discretization for nonlinear PDEs with application to the pricing of an American put, Vol. 23 of Lect. Notes Comput. Sci. Eng., Springer, Berlin, 2002, pp. 189-202.

[49] G. Staff, G. Rnquist, Stability of the parareal algorithm, in: Proceedings of Fifteen International Conference on Domain Decomposition Methods, Springer Verlag, (2003), pp. 449-456.

[50] G. Bal, On the Convergence and the Stability of the Parareal Algorithm to Solve Partial Differential Equations, Vol. 40 of Lect. Notes Comput. Sci. Eng., Springer, Berlin, (2005), pp. 425-432.

[51] M. Gander and S. Vandewalle, Analysis of the Parareal Time-Parallel Time-Integration Method, Siam J. Sci. Stat. Comput. 29 (2) (2007) 556-578.

[52] I. Garrido, M. S. Espedal, G. E. Fladmark, A convergent algorithm for time parallelization applied to reservoir simulation, in: Proceedings of Fifteen International Conference on Domain Decomposition Methods, Vol. 40, Springer, Berlin, (2004), pp. 469-476.

[53] P. F. Fischer, F. Hecht, Y. Maday, A parareal in time semi-implicit approximation of the navier-stokes equations, in: Proceedings of Fifteen International Conference on Domain Decomposition Methods, Vol. 40, Springer Verlag, (2004), pp. 433-440.

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

[55] M. J. Gander and E. Hairer, Nonlinear convergence analysis for the parareal algorithm, in: Domain Decomposition Methods in Science and Engineering XVII, Vol. 60, Springer, (2008), pp. 45-56.

[56] Y. Liu, J. Hu, Modified propagators of parareal in time algorithm and application to Princeton Ocean model, Internat. J. Numer. Methods Fluids 57 (12) (2008) 1793-1804.

[57] B. A. Carreras, K. Sidikman, P. H. Diamond, P. W. Terry, L. Garcia,Theory of shear flow effects on long-wavelength drift wave turbulence, Phys. Fluids B 4 (10) (1992) 3115-3131.

[58] J. A. Mier, R. Sánchez, L. Garcia, D. E. Newman, B. A. Carreras, On the nature of transport in near-critical dissipative-trapped-electronmode turbulence: Effect of a subdominant diffusive channel, Phys. Plasmas 15 (11) (2008) 112301.

[59] D. Samaddar, D. E. Newman, R. Sanchez, Parallelization in time of numerical simulations of fully developed plasma turbulence using the parareal algorithm, J. Comp. Phys. (2009) (submitted).

Chapter 2
The effect of noise as a decorrelation surrogate in a reaction-diffusion equation like transport model [1]

Abstract

Simple dynamical models of transport have been able to capture much of the dynamics of the transport barriers found in many devices. However, these models, which have much in common with the classic reaction-diffusion equations, have wave like structures that can propagate in certain regimes near transition points. This extended propagation, while being realistic in a reaction diffusion model, is probably limited in a turbulent plasma due to the turbulent decorrelation. In order to investigate methods for correcting the extended propagation, noise is added to the system to simulate the intrinsic decorrelations. The wave propagation characteristics are studied as a function of the noise amplitude and compared to similar studies in reaction diffusion systems in which propagation can actually increase as well as decrease in the presence of noise.

2.1 Introduction

The successful confinement of energy and particles in fusion plasmas has remained an area of active research for decades. Internal transport barriers (ITB) suggest the possibility of a partial solution to confinement issues, but are also not devoid of issues that need to be resolved. The presence of internal transport barriers has long been verified in experiments [1], [2], [3]. However, numerical studies of tokamak plasmas can also help in understanding the underlying physics, although modeling the actual dynamics of such plasmas can be highly challenging. A full 3-D turbulence model, that encompasses the entire physics on transport timescales, and aims at describing all the phenomena associated with tokamak plasmas, can be computationally exhaustive. Therefore, the comparatively simpler approach of studying the dynamics by using a one dimensional transport model is often used. Such a model is computationally much less intensive, but can still give realistic

[1]D. Samaddar, D.E. Newman and J. Broussard. 2010. The effect of noise as a decorrelation surrogate in a reaction-diffusion equation like transport model. Prepared for submission in Physics of Plasmas.

dynamics for the system.

A transport model based on nonlinear coupling between the turbulent fluctuations and the radial electric field, giving realistic transition dynamics for tokamak plasmas, has been developed and used in [4], [5]. This is an envelope model, with averaging over the flux surface leading to a one dimensional (radial) system. In this model, the turbulent fluctuations evolve with time. The model well describes the transport properties observed in the Tokamak Fusion Test Reactor (TFTR) [6]. Along with other transition dynamics, the model exhibits oscillations in the fluctuations, which are found to propagate through the system. As has already been described in [5], these waves have a physical and not numerical origin, and may be treated as bursts in the fluctuation level. However, although the waves are physically realistic, they appear to propagate unphysically far. It is proposed that the cause of this enhanced propagation is the lack of realistic turbulent decorrelation in an envelope equation. Therefore, the addition of noise can act as a surrogate to decorrelate the waves.

Noise added to other systems, including those based on reaction diffusion equations, have been found to have interesting consequences with respect to wave propagation [7], [8], [9]. In these systems, an optimal value of noise has been found, when an enhanced propagation of the waves is observed, whereas, for other values, noise leads to decorrelation. The latter type of behavior has particularly acted as a motivation for adding noise to our model. In this chapter, we explore the question whether noise can act as a decorrelation surrogate in an envelope model, thus leading to more realistic results with respect to experiments.

In the next section, we briefly describe the model. The effect of noise on the propagation distance of the waves is discussed in Section 2.3. Section 2.4 describes our methods of evaluating the correlation of the waves, followed by a discussion of the results obtained. Section 3.5 gives a summary of the effectiveness of noise in decorrelating the waves in our system.

2.2 The model

The basic heuristics of the model, as proposed and studied in [4], [5], is given in Fig. 2.1. In this model,the turbulent fluctuations depend on the square of the gradient of the radial electric field. In addition to that, the turbulent fluctuations evolve with time. Moreover, the instability growth rate, γ, which depends on the pressure gradient, has a radially dependent profile. The fluctuations lead to anomalous transport, helping to set a pressure gradient across the plasma. The external energy source also affects the pressure gradient. The radial electric field, in this model, is a function of the pressure gradient. Momentum transported into the plasma by an external source, also affects the radial electric field. The part of the loop in Fig. 2.1, concerning the Reynolds Stress, is primarily related to the poloidal flow, which is not being considered in this envelope model, at present. A system of time dependent equations of the density (n), ion temperature (T_i), electron temperature (T_e), and ε, which is the local fluctuation intensity, is used.

$$\frac{\partial n}{\partial t} = S_{NBI} + S_{gp} + \frac{1}{r}\frac{\partial}{\partial r}[rD_n\frac{\partial n}{\partial r}] \tag{2.1}$$

$$\frac{3}{2}\frac{\partial nT_i}{\partial t} = \frac{1}{r}\frac{\partial}{\partial r}[r(\chi_i n\frac{\partial T_i}{\partial r} + \frac{5}{2}D_n T_i\frac{\partial n}{\partial r})] - D_n\frac{1}{n}\frac{\partial n}{\partial r}\frac{\partial nT_i}{\partial r} + Q^i_{NBI} + Q_{ei}(T_e - T_i) \tag{2.2}$$

$$\frac{3}{2}\frac{\partial nT_e}{\partial t} = \frac{1}{r}\frac{\partial}{\partial r}[r(\chi_e n\frac{\partial T_e}{\partial r} + \frac{5}{2}D_n T_e\frac{\partial n}{\partial r})] + D_n\frac{1}{n}\frac{\partial n}{\partial r}\frac{\partial nT_i}{\partial r} + Q^e_{NBI} + Q_{Ohm} + Q_{ie}(T_e - T_i) \tag{2.3}$$

$$\frac{\partial \varepsilon}{\partial t} = \{\gamma - \alpha_1\varepsilon - \alpha_2[\frac{r}{q}\frac{\partial}{\partial r}(\frac{q}{r}\frac{E_r}{B_\phi})]^2\}\varepsilon + \frac{1}{r}\frac{\partial}{\partial r}[rD_\varepsilon\frac{\partial \varepsilon}{\partial r}] \tag{2.4}$$

E_r is the radial electric field given by

$$E_r = -V_\theta + V_\phi\frac{B_\theta}{B_0} + \alpha[\frac{\partial T_i}{\partial r} + \frac{T_i}{n}\frac{\partial n}{\partial r}], \tag{2.5}$$

In this model, the transport is based on the ion temperature gradient (ITG) driven turbulence at the core. The variable $Q_{ie} = -Q_{ei} = 3\frac{m_e}{m_i}n\nu_e$ is the energy exchange term between electrons and ions, and the other source terms are Q_{NBI}, which is the heating due to the neutral beam injection, and Q_{ohm} is the Ohmic heating. The source terms corresponding to the density are S_{gp}, which is the edge source term resulting from gas being pumped in from the edge, and S_{NBI}, which corresponds to the neutral beam injection. χ_i and χ_e are the thermal conductivities for the electrons and the ions, respectively. The diffusivities are taken as constants in time and space, in this model, with D_n being the sum of the neoclassical and anomalous diffusivities, and D_ε being the diffusivity for anomalous transport.

Eq. (2.4) shows that the fluctuations ε depend on the square of the gradient of the radial electric field, and ε also evolves with time, t. γ in Eq. (2.4) is the growth rate of the fluctuations. α_1 is given by a function of the plasma parameters and the mode width, as given in [4]. α_2 is a parameter which is a function of the radial correlation length, discussed in more detail in [4].

As is evident from the last term in Eq. (2.5), the radial electric field, E_r depends on the gradient of the pressure, P, which is a function of the temperature, T, and the density, n. As there are no toroidal (ϕ) or poloidal ((θ) components of the velocity in this envelope model, V_θ and V_ϕ in Eq. (2.5) are set equal to zero. The coefficient, α, is given by $\alpha = \frac{1}{|e|}$, e being the charge of an electron.

In this work we have added white noise (ξ) at random locations to γ in Eq. (2.4). So, $\gamma \longrightarrow \gamma + \xi$ with the addition of noise. As is evident from Fig. 2.1, the fluctuations, pressure gradient, radial electric field are all interdependent. This justifies our adding noise to just one term, to study the effects.

The radial distance, a, was divided uniformly into 100 points, also referred to as sites, in order to achieve spatial discretization. r is the radial separation of each site from the center or r = 0. At each time step, a given number of randomly selected sites were perturbed by random noise of a predetermined range. Since the sites that were perturbed were also picked randomly, it resulted in the fact that the same site was not perturbed for the entire length of the total simulated time. This method gave us two different ways of varying the noise. One way was to vary ρ or the number of sites perturbed, keeping the amplitude or α constant. The amplitude or α is the maximum fractional pertubation of . The second way was to vary the amplitude of noise, for a fixed number of sites. Hence, the effective noise in the system is a function of both the number of perturbed sites, and the amplitude of noise. This method gave us the opportunity to explore the parameter space.

2.3 Propagation distance

As stated above the noise added to our system has been quantified in two different ways. One way was to vary the number of sites perturbed, and the other was to vary the amplitude of the noise. Both variations appear to lead to interesting consequences. Figs. 2.2 and 2.3 show the propagation of the oscillations with time, in presence of different amplitudes of noise added to a fixed number (40) of sites. Figs 2.2 and 2.3 are contour plots of the time evolution of the fluctuation profiles, ε, showing the waves and their propagation. The structures observed in the region in space approximately between $r/a = 0.08$ and $r/a = 0.55$ (r being the radial distance and a being the total length) actually represent the waves. Since the time increases in a top-down fashion in the plots, and the radial distance increases from left to right, the structures curving downward towards the right indicate that the waves are moving radially outward with increasing time.

As is evident from Figs. 2.2 and 2.3, a region exists around a certain radial location (about $r/a = 0.35$), where an amplification of the waves occurs. This results from the fact that the $\vec{\nabla}P$ and $\nabla^2 P$ both have their maxima around that point, which causes a significant increase in γ, the growth rate. This fact is also evident in the radial profile of the fluctuations, which is shown in Fig. 2.4. Fig. 2.4 is a plot of the fluctuation level with the radial separation. The fluctuation is very small at the inner edge (small radial separation), and has a maximum close to the outer edge (larger radial separation). However, this transition is not linear, but has a maximum at about $r/a = 0.35$ due to the variation in γ, as mentioned earlier. [5] discusses this in greater detail.

When no noise is added to the system, waves are found to propagate to $r/a = 0.73$. Oscillations in the fluctuation level can be clearly observed at $r/a = 0.60$, when no noise is added to the system. As the noise level is increased, the propagation distance of the waves appears to be reduced. The oscillations in the fluctuation decrease as r/a increases. In Fig. 2.5, for a case where 10% of sites are randomly perturbed by a noise amplitude of 20%, the oscillations die out with increasing radial separation. At $r/a = 0.44$, the oscillations are big, but at $r/a = 0.56$ the same oscillations are almost reduced to noise level.

As the perturbation amplitude is continued to increase, the furthest position of propagation or propagation length, R continues to exponentially (approximately) decrease, until a point is reached where increased noise has very minimal effects. This is evident in Fig. 2.6, although the points for very low perturbation amplitudes were difficult to determine because of the scatter. With systems experiencing most noise, the propagation distance is roughly reduced to $r/a = 0.50$, which is approximately $r/a = 0.73$ in the absence of any noise. The addition of noise does not eliminate all oscillations in the fluctuation, but it does appear to restrict the region of its occurrence, perhaps consistent with real experiments.

2.4 Correlation

To obtain a quantitative estimate of the effect of noise in decorrelating the waves, the cross correlation function of the fluctuations at different sites is measured, with respect to a particular site ($r/a = 0.25$). The cross correlation function, C, for two signals g(t) and f(t), say, is defined as follows:

$$C(\tau) = \frac{\Sigma_t f(t) g(t+\tau)}{\Sigma_t (f^2(t) g^2(t))^{\frac{1}{2}}}, \tag{2.6}$$

Fig. 2.7 shows a plot of the cross correlation functions versus τ, for different radial sites. The fact that the peak of the cross correlation function changes position with respect to τ, for different cases of radial sites, actually highlights the wave-like nature of the propagation. The amplitude of the peak also decreases with increasing radial separation, or, as the waves propagate, which means, the waves are gradually decorrelated with time.

As the waves propagate, the peak in C occurs for a different lag, or τ, for each site. Hence, C is calculated for different τs, from which the maximum (C_{max}) is extracted, for each site. These maximum values of the correlation function are then plotted against the radial separations. A 'bump' or an increase in correlation appears in the plot at around a given value of r, resulting from the amplification of the waves in that region, as is evident in Fig. 2.8. This 'bump' is expected as when the waves are amplified around $r/a = 0.35$, there is also an enhancement of the correlation. However, this is true only when C is measured with respect to a site for which $r/a < 0.35$. If this reference site has $r/a > 0.35$, the bump is expected to lose prominence, as there is no significant amplification of waves at sites beyond the reference site. In this case, an approximately exponential decay in the cross-correlation function is expected.

At this point, it needs to be noted, that, the height and width of this bump, for reference site with $r/a < 0.35$, decreases with increasing noise, in general (Figs. 2.9

and 2.10). This is due to the fact, that, if noise contributes to decorrelating the waves, the amplification of these waves reduces, resulting in reducing the area of the 'bump'. However, for low value of noise, the 'bump' becomes quite wide, and thus dominates the correlation. Thus, even for a reference site with $r/a > 0.35$, an exact exponential decay of the cross-correlation function, as the radial separation with the reference site is increased, is not obtained.

The cross correlation function, C, is first calculated with respect to the signal at $r/a = 0.25$. So, as we move radially outward, the region of wave amplification, and hence, that of enhanced correlation, is included. Once C is calculated, two approaches are pursued to derive a measure for the correlation. If the bump that appears due to the enhanced correlation, is treated as a gaussian, then, the standard deviation, (σ), for it, may be treated as a measure of the correlation. On the other hand, the same plot of the maximum correlation versus radial separation, without the 'bump', is treated as an exponential. Now, if the area under this exponential fit is subtracted from the total area of the plot, the remaining area gives the area of the gaussian, which also serves as a measure of the correlation.

Next, C is measured with respect to $r/a = 0.40$, so the 'bump' disappears from the radial profile of the cross correlation function. Then, if an exponential fit to the plot of maximum correlation vs. radial separation is made, the exponentiation length also gives an estimate of the correlation. It needs to be noted at this point, that, as already discussed, for very low noise values, when waves propagate far out along the radial distance, a perfect exponential decay of the maximum correlation with radial separation is hard to obtain. However, this method of calculating the correlation length also succeeds in giving an estimate about the effect of noise in decorrelating the waves.

2.4.1 Results

The addition of noise clearly affects our system. Our results suggest that in certain cases, noise can lead to enhanced correlation of the waves, and in other instances, the waves are decorrelated. First, we consider the situation where the reference site for the calculation of the cross correlation function is $r/a = 0.25$. In that case, when a given number of sites are perturbed with gradually increasing amplitude of noise, the correlation exhibits an interesting trend. It initially increases with noise amplitude, but then starts falling off, indicating a noise induced decorrelation of the waves. This is evident in Fig. 2.11, where the area of the gaussian is taken as a measure of correlation. Using σ as a measure also gives a very similar result.

If the amplitude, α is fixed at about 0.30%, and the number of sites, or (pthresh or ρ), is varied, the correlation also decreases with increasing noise, as shown in Fig. 2.12, where the area of the gaussian is used to measure the correlation. The same plot, using σ, gives a similar result.

In order to explore in greater detail, the effect of noise on our system, a 3-D plot is used.

A three dimensional plot is made with the correlation, number of perturbed sites (pthresh) and noise amplitude along the three axes. In Figs. 2.13 and 2.14, the area of the gaussian is used to measure the correlation. In general, for a large number of sites perturbed by a high amplitude of noise, the cross correlation function greatly reduces, indicating a strong decorrelation of the waves with noise. In this case, the bump in the plot for the maximum cross correlation vs. radial separation almost disappears. Again, for a very low number of sites, or pthresh, perturbed with a very small amplitude, noise is expected to have minimal effects. In that case, the 'bump' in the plot mentioned earlier, becomes wide, indicating the amplification of waves over a large number of sites. This leads to stronger correlation, thus resultig in a comparatively large value of the cross-correlation function. However, results become interesting, when the amplitude is slightly increased from its small initial value.

This is evident from Fig. 2.13 and 2.14, in which multiple regimes exist, where noise helps in the propagation of the waves. In case of Figs. 2.13 and 2.14, for a region of relatively low (but not extremely small) noise amplitude, the increase in amplitude first increases the cross-correlation function, then reduces it again.

The same characteristics of the correlation are observed, where the standard deviation, σ, of the gaussian, is taken as its measure.

Next, the case with $r/a = 0.40$ as the reference site for the calculation of the cross correlation function is considered. As this reference site is at a radial distance greater than $r/a = 0.35$, the maximum value of the cross correlation function at each site, approximately exhibits an exponential decrease with increasing radial separation. If the exponentiation length is taken as the correlation, a three dimensional plot of the number of perturbed sites (pthresh), noise amplitude and the correlation should give a similar result as obtained in the previous case with $r/a = 0.25$ as the reference site. This is observed in Fig. 2.15. In this case, it also becomes clearly evident that at low noise amplitude, there is a regime where the correlation first increases, then falls off again, with increasing amplitude of noise.

We tried to overlap the two parameters of noise - the amplitude and the number of perturbed sites- that we have separately explored. With ρ being the number of sites perturbed, α the noise amplitude and ξ_{eff} being the effective noise, the best relation between them that we succeeded in obtaining is

$$\xi_{eff} = (\alpha)^{\frac{3}{2}} \times (\rho), \qquad (2.7)$$

The variation of the correlation with noise, expressed by this relation of amplitude and pthresh, is shown in Fig. 2.16. It is evident that there exists some value of noise for which the cross correlation function increases, but for other values, there is decorrelation in the waves resulting in significant decrease in the value of the cross correlation function.

Hence, it appears that noise does not necessarily decorrelate the waves, but also leads to enhanced propagation, at certain regimes. This aspect of noise is also seen in other systems as has been discussed earlier. Systems which obey reaction diffusion equations tend to exhibit noise enhanced propagation for certain noise values, but there, too, decorrelation of the signal is observed at other noise regimes. However, in general, optimum noise does decorrelate the waves, thus restricting propagation.

2.5 Conclusion

The envelope equation used in the transport model gives realistic transition dynamics, but the waves propagate unphysically far. Addition of a large amplitude of noise to a large number of sites appears to decorrelate the waves. There are multiple regimes in noise space where noise actually enhances propagation, but in general, noise reduces propagation. We have used three different measures to estimate the correlation - namely, the area and the standard deviation (σ) of the gaussian for reference sites with $r/a < 0.35$, and the exponentiation length for reference sites with $r/a > 0.35$. The results based on these three methods support each other as expected. In all cases, the variation of the correlation with the noise amplitude and the number of perturbed sites is similar.

We have also tried to overlap these two parameters of noise and tried to explore the variation of correlation with it. This also indicates a region of noise enhanced propagation, while there is a decrease in the correlation for other values of noise. With the parameters overlapping, for very low noise values, the correlation length does not appear to vary significantly. With increasing noise, the correlation length then increases, followed by a sharp fall off.

Thus, sufficient noise may be used as a surrogate for turbulent decorrelation in a real turbulence model. While a region with low amplitude of noise exhibits an increase in the correlation, consistent with noise enhanced propagation reported in other reaction diffusion systems, the correlation seems to fall off for further increase in noise amplitude, thus suggesting a mechanism to reduce the propagation in the full transport model, and hence, improve their validity.

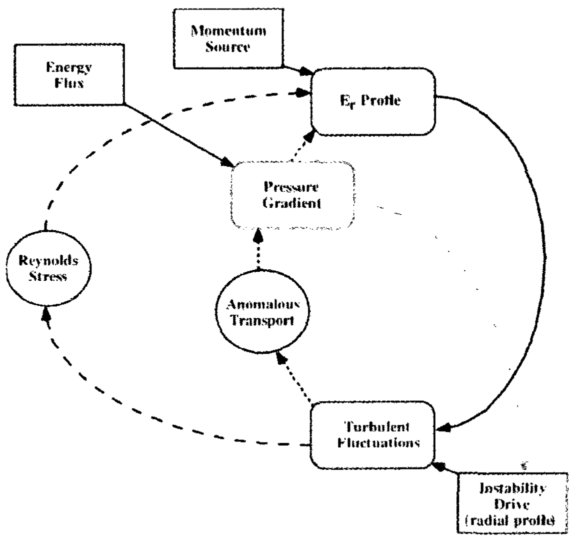


Figure 2.1. The heuristics of the 1D model.

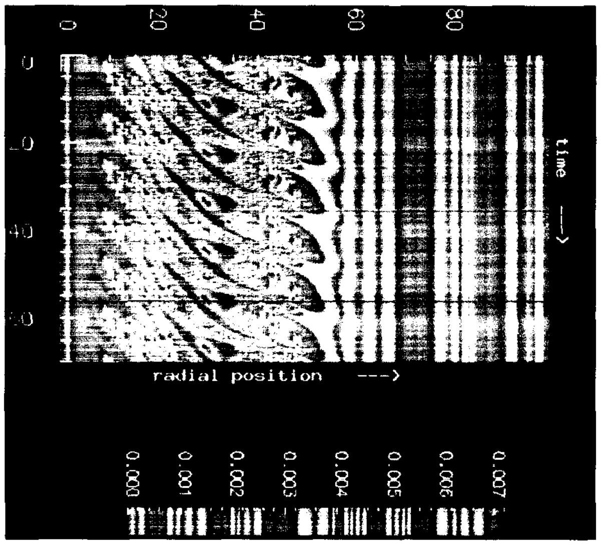


Figure 2.2. 40% of sites are perturbed by noise of amplitude of 0.0000001. Increased noise reduces the propagation length, in general.

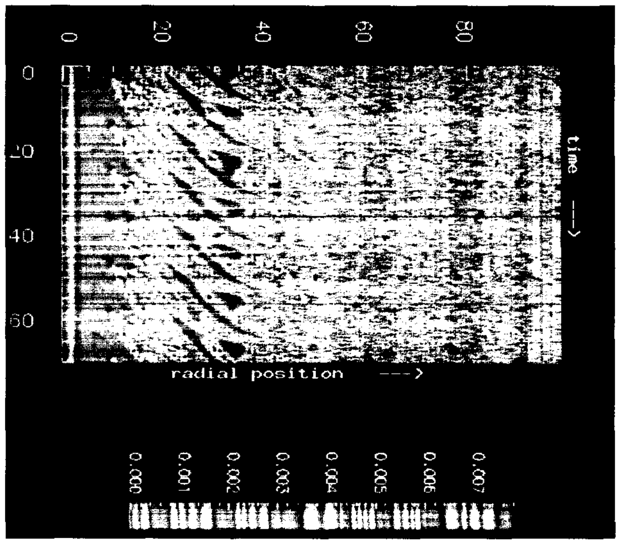


Figure 2.3. 40% of sites are perturbed by noise of amplitude of 0.30. Increased noise reduces the propagation length, in general.

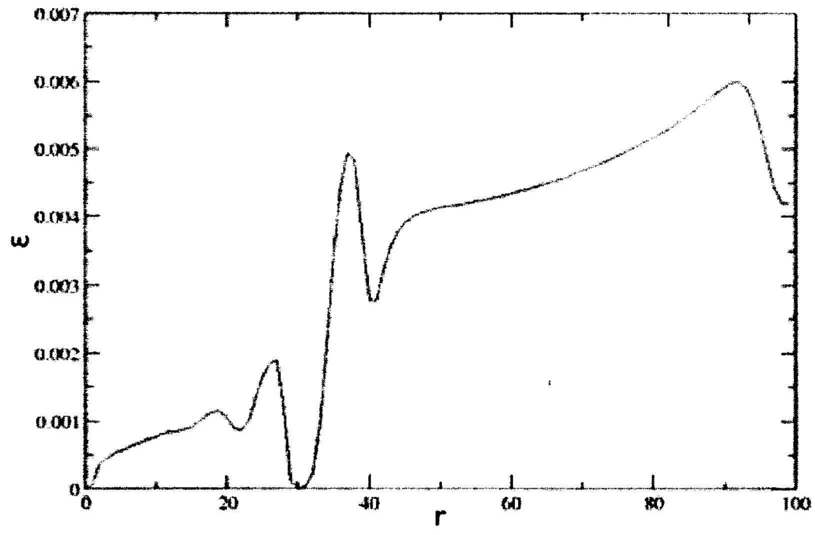


Figure 2.4. A radial profile of the fluctuations shows a region of amplification of the waves, in the absence of any noise.

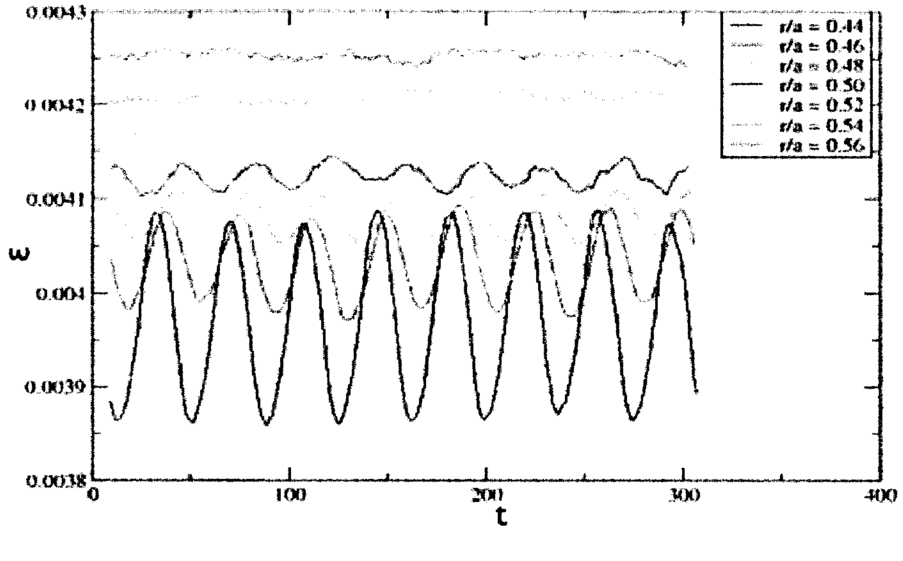


Figure 2.5. 10% of sites are randomly perturbed by 20%. The oscillations in the fluctuation gradually appear to die out.

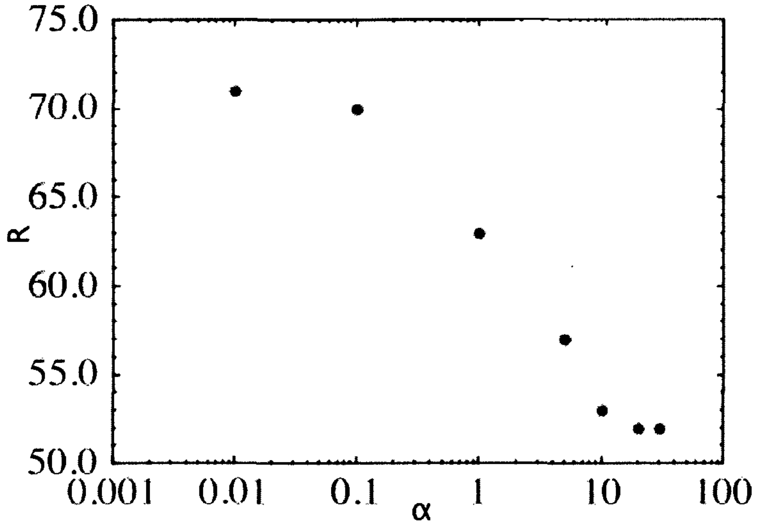


Figure 2.6. The propagation distance, or the furthest point for the observation of the oscillations, reduces with increasing noise amplitude, until a saturation point is reached.

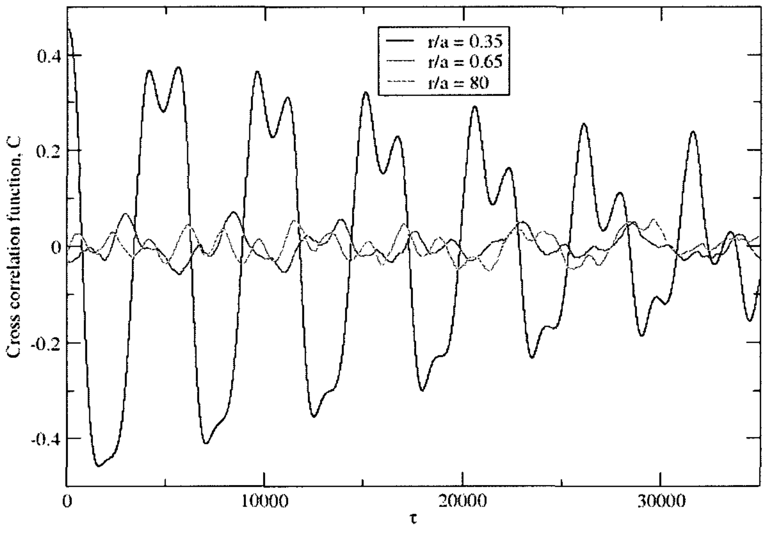


Figure 2.7. Cross correlation function, C, with respect to $r/a = 0.23$, plotted against increasing τ, for different radial distances. The maximum of C or C_{max} reduces in magnitude and shifts outward with respect to τ, for sites with increasing r/a.

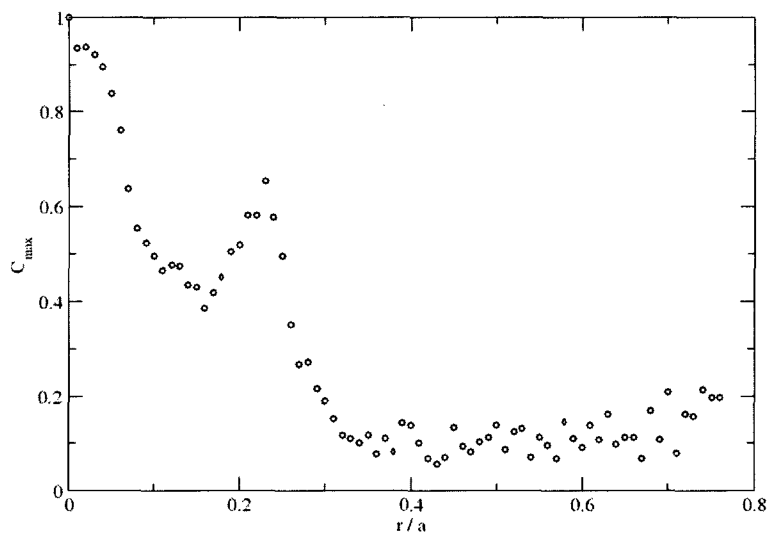


Figure 2.8. Cross-correlation function
The maximum value of the cross-correlation function is plotted against the radial separation. The 'bump' is due to the amplification of the waves in that region.

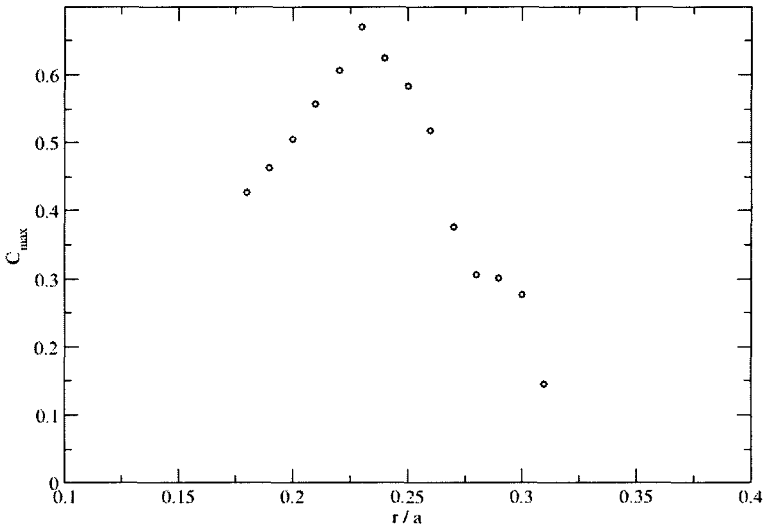


Figure 2.9. The area of the bump decreases with increasing noise - as noise contributes to decorrelating the waves.

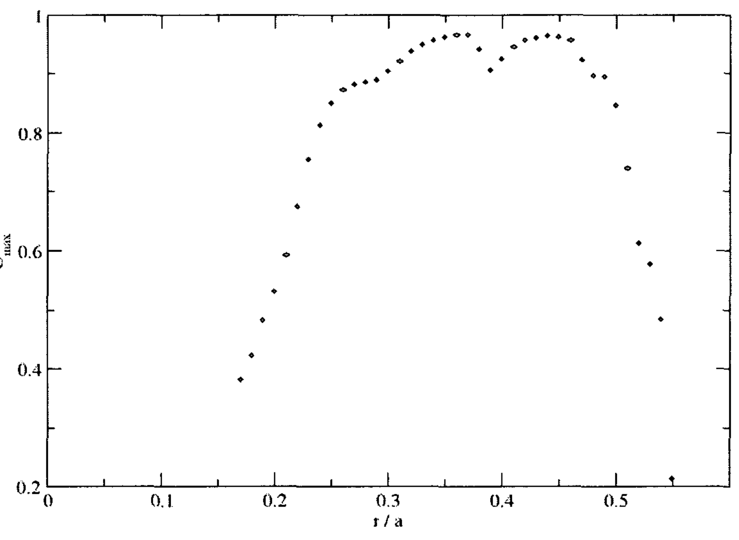


Figure 2.10. Less decorrelation by small noise : The area of the bump is large for small noise.

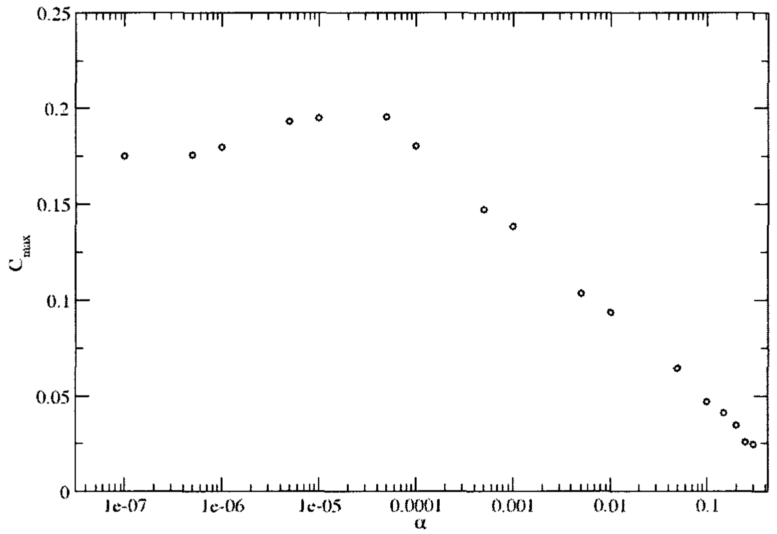


Figure 2.11. The correlation tends to increase, then falls off, with increasing amplitude of noise, when 40% of sites are perturbed, when the area of the gaussian is taken as a measure of the correlation.

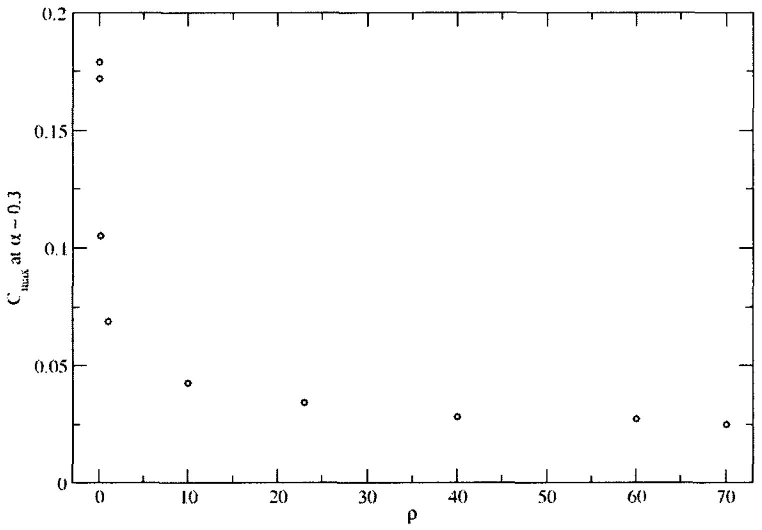


Figure 2.12. As the number of perturbed sites is increased, with the perturbation amplitude remaining fixed at 0.30%, the correlation first decreases sharply, then exhibits a very slow decrease, with the area of the gaussian being a measure for the correlation.

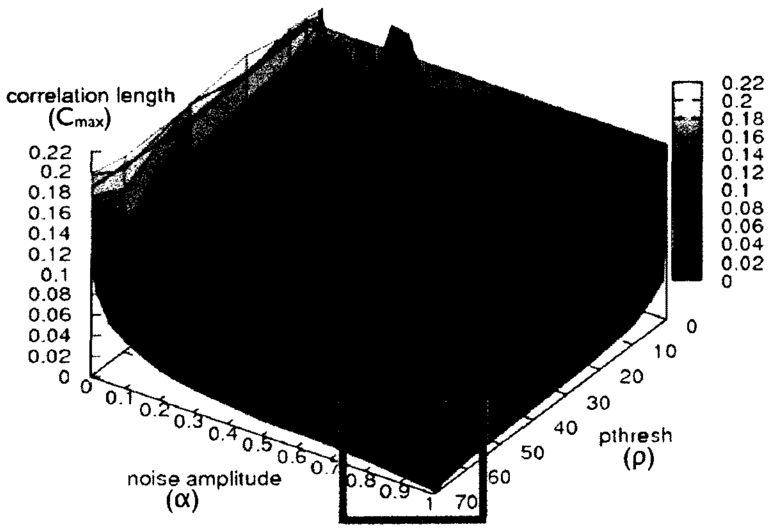


Figure 2.13. Using the area of the gaussian as a measure of the correlation, a high number of sites perturbed by a large noise amplitude shows reduced correlation.

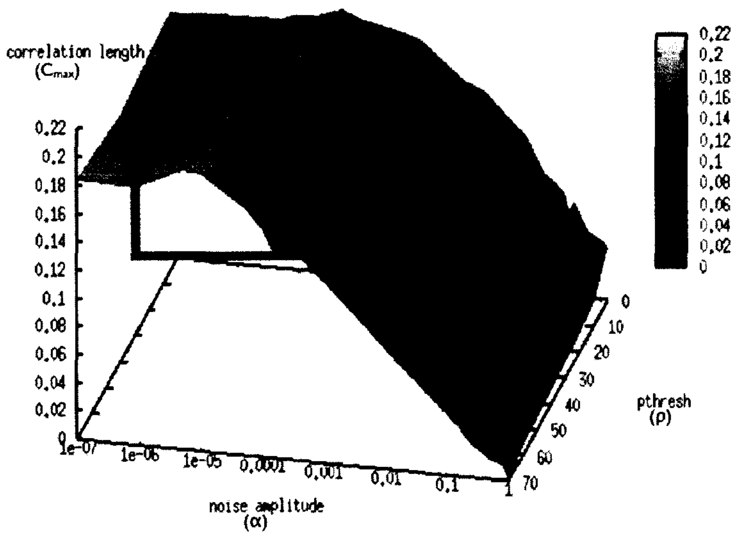
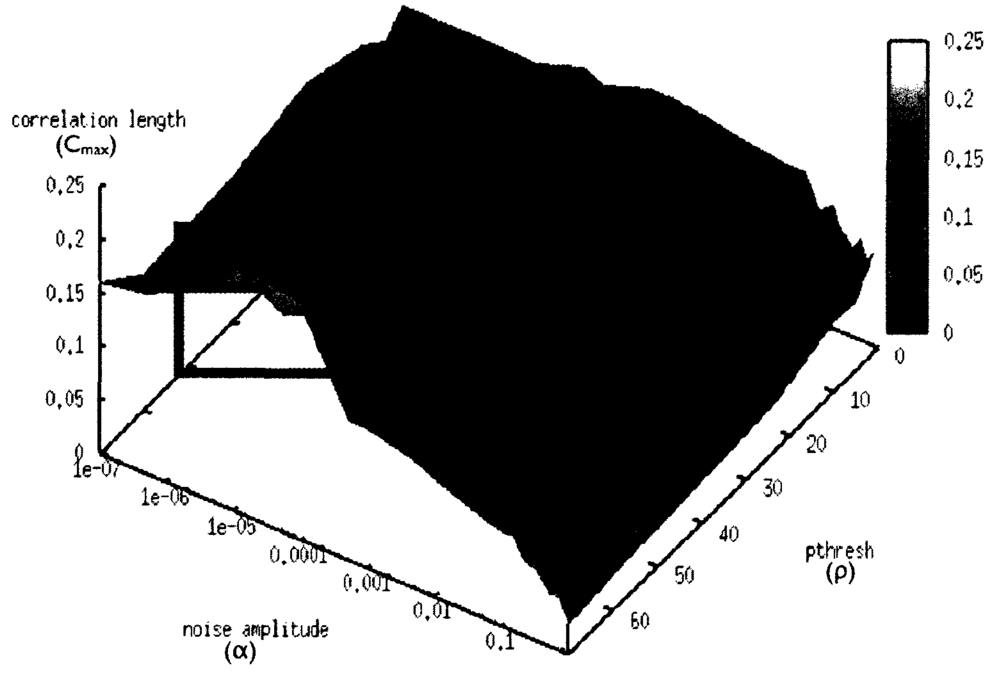


Figure 2.14. Using the area of the gaussian as a measure of the correlation, tehsemi log plot shows a region of low noise amplitude exists, where increasing the amplitude increases correlation.



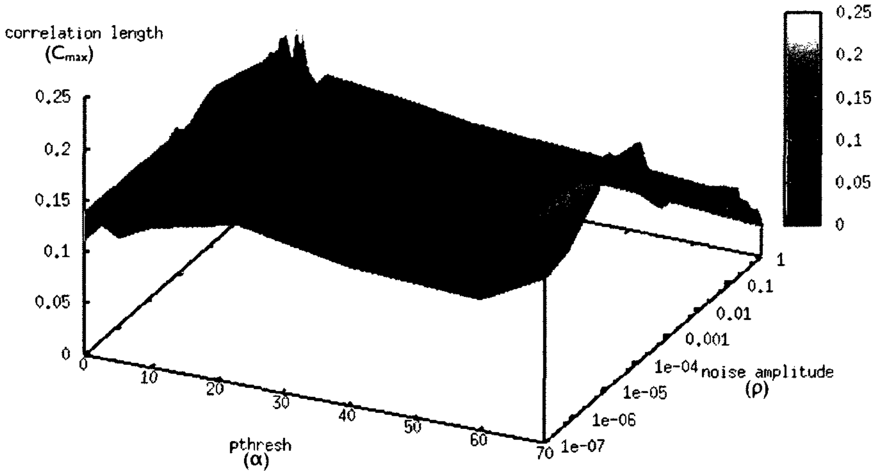


Figure 2.15. When $r/a = 40$ is used as the reference site for calculating the cross correlation function, the correlation varies similarly to the case where $r/a = 0.25$ is used as a reference site. Here, too, noise enhanced propagation is observed at a region of low amplitude noise.

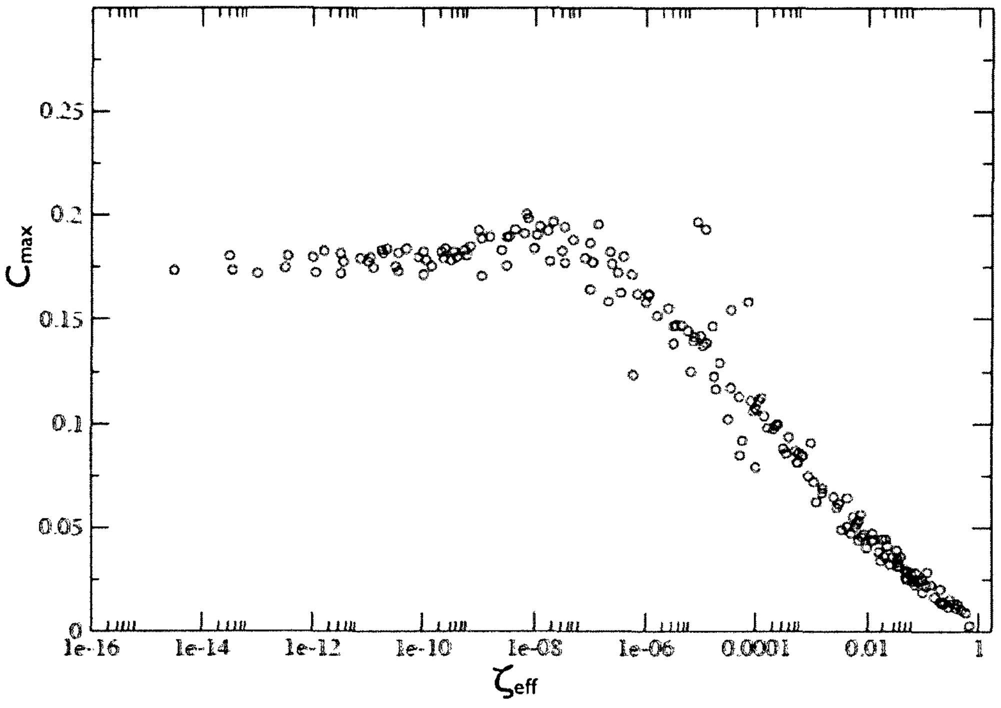


Figure 2.16. If the effective noise is defined as $\xi_{eff} = (\alpha)^{\frac{3}{2}} \times (\rho)$, noise enhanced propagation is observed where the correlation function increases with noise.

Bibliography

[1] Y. Koide, M. Kikuchi, M. Mori, S. Tsuji, S. Ishida, N. Asakura, Y. Kamada, T. Nishitani, Y. Kawano, T. Hatae, T. Fujita, T. Fukuda, A. Sakasai, T. Kondoh, R. Yoshino, and Y. Neyatani, Phys. Rev. Lett. 72, 3662 (1994).

[2] B. LeBlanc et al., Phys. Plasmas,2, 741 (1995).

[3] M. Greenwald et al., Phys. Rev. Lett. 53, 352 (1984) .

[4] D. E. Newman, B. A. Carreras, D. Lopez-Bruna, P. H. Diamond, and V. B. Lebedev, Phys. Plasmas,5(4), 938-952 (1998).

[5] D. López-Bruna,D. E. Newman, B. A. Carreras, and P. H. Diamond, Phys. Plasmas,6(3), 854-862 (1998).

[6] G. Bateman, Phys. Fluids B, 4, 634 (1992).

[7] S. Kádár, J. Wang, and K. Showalter, Nature (London) 391, 770-772 (1998).

[8] J. F. Lindner, S. Chandramouli, A. R. Bulsara, M. Löcher, and W. L. Ditto, Phys. Rev. Lett. 81, 5048-5051 (1998).

[9] C. Zhou, J. Kurths, Z. Neufeld, and I. Z. Kiss, Phys. Rev. Lett. 91,150601 (2003).

Chapter 3
Parallelization in time of numerical simulations of fully developed plasma turbulence using the parareal algorithm [1]

Abstract

It is shown that numerical simulations of fully developed plasma turbulence can be successfully parallelized in time using the parareal algorithm. The result is far from trivial, and even unexpected, since the exponential divergence of Lagrangian trajectories as well as the extreme sensitivity to initial conditions characteristic of turbulence set these type of simulations apart from the much simpler systems to which the parareal algorithm has been applied to this day. It is also shown that the parallel gain obtainable with this method is very promising (close to an order of magnitude for the cases and implementations described), even when it scales with the number of processors quite differently to what is typical for spatial parallelization.

3.1 Introduction

In magnetically confined hot plasmas with interest for fusion research, such as those confined in a tokamak or a stellarator, the disparity in timescales between the microturbulence responsible for the transport processes and the plasma confinement times is of the order of $10^6 - 10^9$. Thus, in order to properly capture the transport dynamics of these strongly turbulent plasmas, they would need to be simulated for thousands or even tens of thousands of turbulent decorrelation times. Such a task is currently well beyond the reach of even the most powerful supercomputers. As a result, simplified approaches are used in which the microturbulence is only evolved for a few tens of decorrelation times under the assumption that the dynamics become decorrelated after this timescale, together with the assumption that plasma profiles can be assumed 'frozen' during these timescales. These approximations enable a description via effective transport coefficients derived from these restricted simulations, whose validity should however be checked

[1] D. Samaddar, D.E. Newman and R. Sánchez. 2009. Parallelization in time of numerical simulations of fully developed plasma turbulence using the parareal algorithm. Submitted in Journal of Computational Physics.

a posteriori, whenever possible. But even with these simplifying approximations, these simulations are still extremely challenging from a computational point of view [1]. On the one hand, because parallelization along the space domain reaches saturation (typically, at a few thousand CPUs in most codes, as a result of Amdahl's law and interprocessor communication overflow), for which a further increase of the number of processors beyond a certain point does not contribute to enhanced speedup. On the other hand, higher spatial resolutions imply the use of smaller time steps for numerical stability reasons. This makes that reaching the needed simulation times takes much longer, due to the serial nature of the temporal coordinate. The situation becomes even more dramatic in cases in which the aforementioned simplifications may not be justified, as might be the case when turbulence is near-marginal [2], or in the presence of strong sheared flows [3].

It thus seems self-evident that if these turbulent simulations could also be parallelized in time (in addition to their spatial parallelization), it would enable a more efficient utilization of the many processors currently available in supercomputers while, at the same time, allowing the running of much longer simulations in considerably shorter wallclock times. This would also open the path to including a more complete set of the physics in the simulations in the near future, maybe even the full transport dynamics. Various approaches have been proposed over the years to decompose the time direction when solving a partial differential equation [4, 5, 6, 7, 8, 9, 10, 11, 12, 13], although with varying degrees of success. Of these, the parareal (parallel in time) algorithm, which we explore in this paper, was first proposed by *Lions et al.* [14] and has received an increasing amount of attention in recent years. It has been successfully applied to a number of relatively simple problems, like molecular dynamics simulations [15], linear and nonlinear parabolic ordinary differential equations [16, 17], stochastic ordinary differential equations [18], reservoir simulations [19] and even, the laminar regime of the Navier-Stokes equation [20]. The scheme has also been applied very recently to the Princeton ocean model, dominated by convection, although with a rather modest sucess [21].

In this paper, we report the first (and very promising) results regarding the parallelization of the temporal direction of numerical simulations of turbulent plasmas using the parareal algorithm. To the best of our knowledge, the parareal technique has never been applied to a fully-developed turbulent problem, although it has been used successfully with low-dimensional chaotic systems, such as the Lorenz system [22]. Turbulent systems represent a very challenging case of study, and there are in fact many reasons to expect failure. Indeed, the parareal algorithm parallelizes along time, despite the sequential nature of the time domain, using a predictor-corrector approach. Since the corrector step carried out at each cycle of the algorithm feeds on the results of the uncoupled predictor runs, it might be expected that the strong sensitivity to initial conditions and exponential divergent growth of uncertainties characteristic of turbulence should deteriorate or even impede the convergence of the algorithm, in contrast to what happens in a more laminar regime. In this paper we show that this is not the case, and that the parareal method, when properly tuned, can be applied successfully to fully-developed turbulent simulations and yields considerable parallel speed-ups (an order of magnitude, for the cases studied here).

In order to avoid the complexities associated with the toroidal geometries characteristic of fusion plasmas, we have chosen to apply the parareal method to a simpler dissipative trapped electron mode (DTEM) turbulence model in a doubly-periodic slab geometry. Some kind of drift wave turbulence is the most probable candidate for governing transport in these plasmas and thus, this model has been studied extensively [23, 24, 25, 26]. For its numerical implementation we use the BETA code [23], which uses a pseudo-spectral approach and advances the system in time using an implicit, preconditioned integrator. BETA routinely provides with fully-developed turbulent states, with large positive Lyapunov exponents. It thus provides an excellent platform to explore and test the merits of the parareal method in this context, before embarking on its implementation in any of the state-of-the-art codes used by the fusion community. The paper is thus organized as follows. Section 3.2 briefly reviews the parareal algorithm. The physics of the DTEM model are then described in Section 3.3. Section 3.4 introduces an analysis of the parallel performance to be expected from the algorithm, which will clarify its strong and weak points as well as guide us through its tuning. Next, Section 3.5 comprises the numerical results obtained in this study. Finally, some conclusions are drawn in Section 4.6.

3.2 The parareal algorithm

In this section, we provide a review of the basic algorithm, including some modifications of our own devising that are appropriate to the turbulent context and yield a singificant boost in its performance.

3.2.1 Review of the algorithm

The parareal algorithm is based on a predictor-corrector iterative approach. It is best understood by describing its application to a single ordinary differential equation of the type:

$$\frac{d\lambda}{dt} = A(\lambda, t), \quad \lambda(0) = \lambda_0 \tag{3.1}$$

where A is an arbitrary, possibly nonlinear, function. Let's assume that we are interested in finding the value of λ at some later time $T > 0$. Let's also assume that we can numerically advance this equation from an arbitary time $t \in [0, T]$ to time $t + dt$ by means of several discretization schemes. We will formally write this advance using;

$$\lambda(t+dt) = \mathbf{F}^t_{dt} \cdot \lambda(t), \tag{3.2}$$

where **F** represents our advancing operator, acting on the appropriate space to which λ belongs, and dependent on the discretization scheme chosen (the superscript t is used here to denote that the operator may depend explicitly on t, although we will drop it in what follows). Clearly, to go from the initial time to T, we will need to apply **F** as many times as required given the value of dt.

The parareal algorithm assumes that there are two different advancing operators (or solvers) at our disposal, that we will denote as **F** and **G**. The distinction between these two solvers is that **G** is much faster (usually at the price of being coarser and more inaccurate) than **F**, the one we are really interested in using for our problem but which is computationally too expensive to be run serially between the initial time and T. In the parareal algorithm, **G** is run serially between $t = 0$ and T, whilst **F** is always run in parallel. Let's see how this works.

Let there be N processors, denoted by $P_0, P_1, P_2, P_3, ..., P_{N-1}$. Let the total simulation time, T, be divided into N smaller *chunks* of size $\Delta T = T/N$. In what follows, the i index is used to represent the i^{th} instant of time defined as $t_i = i \cdot \Delta t$ for $i = 0,1,2,$ ..., N. The index $k = 0,1,2,$ ... represents the iteration number in the parareal cycle. Then, λ_i^k represents the solution at time t_i at the k-th iteration of the parareal cycle. The initial value, that is already given, is then denoted by λ_0^0.

The steps involved in the parareal scheme are as follows:

- Iteration $k = 0$:

  P_0 uses **G** serially to calculate initial values λ_i^0 for the start time of every time chunk, t_i.

- Iteration $k > 0$:

  Step 1: Each processor (i.e., P_j) then separately applies **F** to propagate the solution, starting with the initial values provided by **G** (i.e., λ_j^{k-1}), between the initial (t_j) and final time (t_{j+1}) of its respective time chunk. This process is of course carried out in parallel. The result of this propagation is forwarded to the next processor in line (P_{j+1}).

  Step 2: **G** is now applied as a sequential (but not continuous) process, using the parareal prescrition to update the initial value at each time chunk:

$$\lambda_{i+1}^{k+1} = \mathbf{G}_{\Delta T}(\lambda_i^{k+1}) + \mathbf{F}_{\Delta T}(\lambda_i^k) - \mathbf{G}_{\Delta T}(\lambda_i^k), \tag{3.3}$$

  Note that this part of the algorithm cannot be done in parallel, because of the first term on the right hand side. Note also that the second and third terms have already been obtained in previous steps and/or iterations.

  Step 3: Check for convergence. The measure of convergence is discussed in the next subsection. If the solution is converged for all chunks, the cycle is exited. Otherwise, another iteration of the parareal cycle is done which, in the standard implementation, involves all chunks. This will not be the case in our simulations, as discussed in the next subsection.

Some remarks are useful at this point. First, in order for the parareal cycle to converge, certain mathematical conditions must be satisfied by **G** and **F**, which were made explicit in *Lions et al.* [14]. seminal paper and appear as specific conditions on the boundedness of the norm of the difference between the two solvers in an appropriate mathematical space [14]. Regretfully, it is very difficult (if not impossible) to translate these conditions into a practical prescription for any particular problem. Trial and error, combined with experience, seems to be the most reliable guide to choose **G**. However, note that in practice, the parareal algorithm will always converge in at most N interations, independently of how 'badly' **G** is chosen. This happens because, at the end of iteration $k = 1$, both P_0 and P_1 already have the correct value of the solution, since **F** has been used in the first processor to propagate the exact initial condition at $t = 0$. For the same reason, at $k = j$, all processors $P_0, \cdots P_j$ already have the correct value. And so on. However, note also that if N cycles are finally needed to converge to the exact solution, we would have used the same (or more, including communications and time for **G**) wallclock time that if we had run the simulation serially with **F**. So no parallel speedup is gained, and we would have done N times more computing work! Thus, parareal works only if convergence is achieved for a number of cycles K much smaller than N, implying that at each iteration more than one chunk needs to converge on average. Whether this is the case or not will depend on our ability to choose **G** for a given **F**.

3.2.2 Metric for convergence

Convergence is achieved whenever some convergence measure reduces below a certain prescribed tolerance. Following with our previous example, in which $\lambda^k(t)$ represents the solution at time t in the k^{th} iteration of the parareal algorithm, we define the local convergence measure as,

$$\sigma_i^k = \int_{t_{i-1}}^{t_i} \frac{|\lambda^k(s) - \lambda^{k-1}(s)|}{|\lambda^{k-1}(s)|} ds. \tag{3.4}$$

That is, the average relative error between the solution at the k-th and the $(k-1)$-th parareal cycles integrated over each chunk i. It should be noted that we have purposely avoided using the exact serial solution that **F** would provide, since it is not useful in practice due to the fact that, in most cases of interest it will not be known, since the main aim of the parareal scheme is to replace calculations involving serial processors! The solution is then converged up to time chunk I if,

$$\sigma_i^k < \text{tol}, \ \ \forall i \leq I. \tag{3.5}$$

Typically, convergence occurs in an orderly fashion, starting with the first chunk and propagating to later times due to the fact that errors are propagated (and enhanced) in that direction by the parareal cycle. This fact has prompted us to modify the standard parareal algorithm in the following way: instead of 'correcting' the solution at each parareal cycle starting with the first chunk, *we perform the 'correction' only for those chunks corresponding to times larger than the last converged chunk time.* In this way, one still obtains a reasonably converged solution, and avoids at the same time resonances that may deteriorate the solution at already converged chunk times as observed in some problems [12]. With this modification, the cases that will be discussed in Section 3.5 have been made to converge using up to four times less parareal cycles than with the original prescription.

3.2.3 Particulars of the application of the parareal method to fully-developed turbulent simulations

We now discuss some crucial issues pertaining to turbulent systems which are relevant to the application of the parareal algorithm to the BETA simulations. First, it should be noted that for any numerical simulation of a very high dimensional chaotic system with positive Lyapunov exponents, it is impossible to define *a unique solution.* The reason is that these systems exhibit exponential Lagrangian (i.e., along turbulent trajectories) divergence and as a result the final solution depends very sensitively on the initial values. So sensitively that, in fact, a serial run done with the same parameters and numerical scheme, but compiled using two different compilers or run on two different machines unavoidably leads to different results when the simulation time is long enough. But note that this fact does not make the solutions worthless because even when different, they are *statistically identical.* What we mean by that is that the fluctuations exhibit the same statistical and correlation properties at saturation, even if the fluctuations are not identical when compared point-by-point at every time. That is, if one compares the probability distribution function (pdf) of the fluctuations at any point, or the temporal or spatial correlation functions, all these solutions will give the same results. The reason is ultimately the fact that the Eulerian (i.e., fixed point) fluctuations are bounded by the finite saturated fluctuation levels, which are themselves limited by the finite amount of free energy available to the system. This fact translates into the simulations having well-defined statistical properties. So, although the Lagrangian trajectories diverge due to their high sensitivity to initial conditions, the solutions are statistically indistinguishable. And this is ultimately what matters, specially when what one cares about is the mean transport or dynamics of these simulations. In the case of the parareal scheme just sketched, this observation is essential, because the scheme matches together solutions obtained at different chunks. If the mistmatch at those connecting times could grow exponentially without bound, it might prevent any convegence of the method (in fact, this is why it has been suggested that turbulence and parareal would be a deadly mix). But on the contrary, the same boundedness that exists for Eulerian fluctuations holds also for these mismatchs. As we will show in what follows, a solution can be found by the parareal method. And it is very easy to see that this parareal solution, although not identical to the serial one obtained starting from the same initial values, is statistically identical in the sense previously described.

A second important aspect regarding the parareal parallelization of turbulent simulation has to do with the convergence measure. As discussed in the previous subsection, the traditional parareal method computes the relative error between the solution of the problem at two successive parareal cycles. In the case of a turbulent simulation, the number of degrees of freedom is enormous (for instance, $\sim 6 \times 10^5$ for the BETA simulations used in this paper), which would make the construction of the convergence measure rather cumbersome. However, it turns out that this is not needed. In fact, we will show that it is enough to require convergence in one global quantity. In our case, the time history of the energy in the system, defined as:

$$E_\phi(t) = \sum_{\mathbf{k}} \phi_{\mathbf{k}}(t) \cdot \phi^*_{\mathbf{k}}(t), \tag{3.6}$$

where $\phi_{\mathbf{k}}(t)$ is the (complex) amplitude of the Fourier harmonic with wave number $\mathbf{k}$ (see Section 3.3 for details). Then we will show that, if the relative error between successive cycles of the total energy is less than a prescribed tolerance, **all** the individual modes containing enough energy to affect that tolerance value are also converged. This is a remarkable result, which simplifies the calculation enormously. It is ultimately due to the fact that in a fully-developed turbulent system, the relative energy fluxes between any pair of modes are set by the nonlinearities, which endows the system with a very strong and resilient coupling [27]. Thus, convergence in one quantity is only possible if all others converge simultaneously, so that the relations imposed by this coupling are preserved. We indeed observed this in our simulations, as will be shown later.

3.3 The model

The parareal algorithm is applied in what follows to numerical simulations of a reduced model, which is a paradigm of plasma drift waves derived in the limit of long wavelengths and a two-dimensional $(x-y)$ slab geometry. This limit of the DTEM model has been extensively studied in the plasma physics literature [23, 25]. It assumes that a uniform magnetic field that confines the plasma exists perpendicular to the slab (i.e., along $\hat{z}$), and that all profile gradients are directed along the $\hat{x}$ direction. These gradients are responsible for the existence of drift waves that propagate along the $\hat{y}$ direction. The underlying instability is assumed to be the dissipative trapped electron mode (DTEM). Plasma ions are treated as a cold fluid, while electrons are assumed to satisfy the Boltzmann relation except for those trapped in the magnetic ripple, which contribute to the dynamics in a way that allows the drift waves to grow unstable. The model can be reduced to a single equation for the potential fluctuations by assuming quasi-neutrality and using the long-wavelegth limit to retain only the so-called $\mathbf{E}\times\mathbf{B}$ nonlinearity:

$$\frac{\partial}{\partial t}(1-\rho_s^2\nabla_\perp^2)\tilde{\phi}+D\frac{\partial^2\tilde{\phi}}{\partial y^2}+\frac{V_D}{2}\frac{\partial\tilde{\phi}}{\partial y}-\frac{4L_nD}{\varepsilon^{1/2}}\left[\nabla_\perp\left(\frac{\partial\tilde{\phi}}{\partial y}\right)\times\hat{\mathbf{z}}\right]\cdot\nabla_\perp\tilde{\phi}=0. \qquad (3.7)$$

Here, $\tilde{\phi}$ is the fluctuating potential. $V_D=\varepsilon^{1/2}(cT_i/eB)L_n^{-1}$ represents the effective diamagnetic drift velocity, T_i being the ion temperature, $\varepsilon^{-1/2}$ the fraction of trapped electrons, B the magnetic field strength and L_n the density gradient scale length. $D=V_D^2/4\nu_{\mathrm{eff,e}}$, where $\nu_{\mathrm{eff,e}}$ is the effective collision frequency of ion-electron collisions. The last term on the left hand side of Eq. (3.7) represents the E×B nonlinearity, the second one is the instability drive and the third one is the one which causes the drift waves to propagate along the y direction.

The numerical solution of this model is done using the spectral code BETA [23].

The Fourier transform of Eq.(3.7) has the form :

$$\frac{d\tilde{\phi}_{\mathbf{k}}}{dt} - \frac{Dk_y^2}{1+\rho_s^2 k_\perp^2}\tilde{\phi}_{\mathbf{k}} + \frac{iV_D k_y}{2(1+\rho_s^2 k_\perp^2)}\tilde{\phi}_{\mathbf{k}} - - \frac{4iL_n D}{\varepsilon^{1/2}(1+\rho_s^2 k_\perp^2)}\sum_{\mathbf{k}'} k_y'(\mathbf{k}\times\mathbf{k}'\cdot\hat{\mathbf{z}})\tilde{\phi}_{\mathbf{k}}\tilde{\phi}_{\mathbf{k}-\mathbf{k}'} = S_{\mathbf{k}} - T_{\mathbf{k}},$$

where $\tilde{\phi}_{\mathbf{k}}$ are the (complex) Fourier harmonics of the potential. $S_{\mathbf{k}}$ are sources and $T_{\mathbf{k}}$ are sinks in k-space that may also be added at this point for convenience. This set of ordinary differential equations (ODEs) are solved using a pseudo-spectral method to evaluate the nonlinearities. Time is advanced by the stiff solver within the VODPK package [28], which uses a (diagonally, right-) preconditioned, Krylov implicit solver based on backward differentiation formulas of variable order and timestep, that are tuned as the integration proceeds in order to keep the errors below a prescribed tolerance.

An example of a typical simulation is shown in Figs. 3.1 and 3.2. The first one shows a snapshot of the vorticity distribution in the system at saturation (the local turbulent velocity is obtained from the potential via $\mathbf{v} = \nabla_\perp\phi \times \hat{\mathbf{z}}$, so that the vorticity is given by $w = \nabla_\perp^2\phi$.]. It illustrates a salient feature of a turbulent system - namely, the existence of eddies at multiple scales, which nonlinearly interact with each other. Also, the anisotropy introduced by the explicit dependence on k_y of the model can be noticed. Fig. 3.2 is a plot of the power spectrum that shows that the system is in a state of fully developed turbulence with most of the energy concentrated at low k values. If one computes the Lyapunov exponent of the system, it is large and positive as expected. This is illustrated in Fig. 3.3, that shows the log-linear plot of the separation of two initially close trajectories along the x and y directions with time, the slope of which gives the Lyapunov exponent.

3.4 Expected performance of the parareal algorithm

In order to gain some insight on what could be expected from the parareal algorithm (and, perhaps, also allow us to design an optimal coarse solver), we carry out in this section a small study on the performance of the model. Two regimes are examined: the so-called *strong* and *weak* scaling regimes. In the strong one, the performance as a function of the number of processors (denoted by N) is examined for a problem of fixed size in time. In the weak regime, the performance as a function of N is considered as the problem is made longer, but keeping the part of the simulation done by a single processor constant. In perfect parallelism, one should find that the speed-up in the strong scaling improves as N whilst, in the weak regime, one finds that the work-per-processor remains constant as N is increased. Of course, in spatial parallelization these ideal scalings are only maintained up to a certain number of processors, above which the serial part of the calculation (i.e. Amdahl's law) or the inter-processor communication dominate the calculation time. Due to the iterative nature of the parareal cycle, its strong and weak scalings are somewhat different, which sets this type of parallelization apart from spatial parallelization strategies. We briefly discuss them in what follows.

3.4.1 Strong scaling

We first consider the case in which the problem to be solved is fixed in size or, in our case, the final time T that must be reached at the end of the simulation is fixed. In the parareal algorithm, two propagators are used - **G** and **F**. Let the wallclock time to solve the problem (even if much more inaccurately) serially using **G** be $T_G^{ser}(T)$, and to solve it serially with **F**, be $T_F^{ser}(T)$. We then define the parameter β as the ratio of these two times:

$$\beta = \frac{T_F^{ser}(T)}{T_G^{ser}(T)} \tag{3.8}$$

That is, β measures how much faster **G** is. Clearly, $\beta > 1$, as **G** is the cheap solver. In each cycle of the parareal algorithm, **F** is applied in parallel with N processors for a period of the simulation $\Delta T = T/N$, and **G** is run in serial for the whole length of the simulation T. Thus, the total time to solve the problem can be estimated (ignoring overheads) as:

$$T_{PA} = k_s(N)\left(T_G^{ser}(T) + \frac{T_F^{ser}(T)}{N}\right) \tag{3.9}$$

where $k_s(N)$ is the number of iterations of the parareal cycle required to achieve convergence, which is an unknown function of N. Thus, the parallel speed-up factor (or gain) for the parareal algorithm is given by:

$$H_{PA}(N) = \frac{T_F^{ser}(T)}{T_{PA}} = \left[k_s(N)\left(\frac{1}{\beta} + \frac{1}{N}\right)\right]^{-1} \tag{3.10}$$

Note that the typical strong scaling for spatial parallelization, $H(N) = N$, is only recovered when $\beta \longrightarrow \infty$ and $k_s(N) = 1$. But in the parareal case, $k_s(N)$ will be a function of N and β will be finite. Success or failure of the parareal will depend on the value for $k_s(N)$, which will depend itself on the choice of the coarse solver $\mathbf{G}$. But even without knowing $k_s(N)$ at this point, several things can be learnt from this model. First, β seems to roughly set the maximum number of processors for which the parareal method yields any net parallel gain. For N much larger than this value, the serial part of the code dominates the calculation and, as predicted by Amdahl's law, performance deteriorates quickly (although our numerical simulations will show that this statement needs to be somewhat revised!). Secondly, a net parallel gain is obtained only for as long as $k_s(N) < N$. In the next section we will construct a phenomenological model for this function based on the BETA simulations.

3.4.2 Weak scaling

We next consider the case in which the problem to be solved has a length $T = N \cdot \Delta T$, being ΔT fixed. That is, the problem length increases linearly with the number of processors. As we said previously, in perfect parallelism one would expect that the wallclock time required to do the simulation is independent of N, since its processor would be doing exactly the same amount of work. That is,

$$W \equiv \frac{T_N(N * \Delta T)}{T_1(\Delta T)} = 1, \tag{3.11}$$

where $T_n(t)$ denotes the wallclock time needed to solve a problem of length t using n processors. In the parareal case, the time needed to solve a problem of size $N \cdot \Delta T$ using N processors is:

$$T^w_{PA} = k_w(N)\left(N \cdot T^{\text{ser}}_G(\Delta T) + T^{\text{ser}}_F(\Delta T)\right), \tag{3.12}$$

from which the work for processor becomes:

$$W_{PA}(N) \equiv \frac{T^w_{PA}}{T^{\text{ser}}_F(\Delta T)} = k_w(N)\left(1 + \frac{N}{\beta}\right) \tag{3.13}$$

Note that the function $k_w(N)$ is different from $k_s(N)$, since now T is not kept fixed. As before, the classical weak scaling for the spatial case, $W_{PA} = 1$, is only recovered if $\beta \longrightarrow \infty$ and if $k_w(N) = 1$. This will certainly not be the case. Again, it seems clear that β roughly sets the maximum number of processors for which a favorable scaling for the work-per-processor should be expected, although how good the scaling would be ultimately depends on the form of $k_w(N)$. We will also use BETA simulations to try to get its phenoomenological form in the next section.

3.5 Results

In what follows, we describe the results of applying the parareal method to parallelize in time a typical BETA simulation that implements the DTEM model as described in Section 3.3. The run used has a resolution of 385×385 (complex) modes in Fourier space [i.e., (k_x, k_y), with k_x and k_y running between $-k_{\max}$ and $+k_{\max}$, with $k_{\max} = 192$], and corresponds to a restart from an already nonlinearly saturated run, in which the initial values of the mode amplitudes are read from a pre-existing file. Turbulence is fully developed at this stage. All linearly unstable modes in the simulation have $|\mathbf{k}| < 70$, value that roughly sets the starting point of the dissipation range. Stability at higher k's is ensured by using a hyperviscosity. The (modified) parareal cycle has been implemented in BETA by making extensive use of subroutines from the Message Passing Interface (MPI) libraries. The different solvers are added as separate subroutines (one for $\mathbf{F}$, and as many as desired for all the $\mathbf{G}$'s that will be tested), and are called as required within the parareal cycle. The section is structured as follows: first, we will simply describe a typical simulation to show that the method works. Secondly, we will use all the simulations we have done to construct a phenomenological model for $k(N, T)$, that gives the number of parareal cycles needed for convergence once N and T are prescribed. We will use this function in the final two subsections, that describe the results of two scaling studies: one for the strong scaling regime, and a second for the weak scaling one.

3.5.1 Parallelization in time via the parareal scheme of BETA simulations

As already mentioned in the previous section, the success of the parareal scheme depends on making a sound choice of $\mathbf{G}$. In fact, arguments could be made, based on the sensitivity to initial conditions and the exponential divergence of Lagrangian trajectories present in turbulent situations, to the effect that such a choice should not exist in a turbulent system, and that the parareal algorithm will inevitably need $K = N$ iterations to converge when using N processors. The results presented and discussed in this section simply intend to show that this is not the case, and that convergence can be indeed achieved for $K \ll N$, which in itself is a quite remarkable result.

After exploring multiple options for $\mathbf{G}$, we have found that one that seems to work pretty well is solving the problem using a smaller resolution in k-space (that is, using a smaller $k_{\max}$ than $\mathbf{F}$) and, in addition, changing the time-advance scheme to something simpler with respect to the fine solver $\mathbf{F}$. The smaller resolution allows us to use a much larger time step, dt_G, for the time advance (in comparison to that used by the fine solver, dt_F). This, in combination with the simpler implementation (for instance, we have tried 2nd and 4th Runge-Kutta (RK) schemes, in addition to the original VODPK scheme), allows us to reach values of β as large as $\beta \sim 1,000$. However, the best results (i.e., maximum parallel gain) are not obtained for the largest β. The reason is that the strategies used to speed up $\mathbf{G}$ (and thus increase β) may also increase the number of cycles required for convergence for a given number of processors N (i.e., the functions $k_s(N)$ and $k_w(N)$ from the previous section). For instance, we have observed that if the region of k-space solved by $\mathbf{G}$ is too restricted (if $k_{\max}$ is too small), the parareal performance deteriorates quickly. The minimum acceptable size seems to be imposed by the physics of the underlying turbulence: the reduced k-space region must contain a sufficient number of dissipative scales!

We conclude this subsection by showing an example of a successful parallelization in time of a BETA simulation using the parareal method. In Fig. 3.4, the time history of the total energy of a saturated BETA simulation is shown as a function of the parareal cycle, k. The coarse solver used for $\mathbf{G}$ is a 4th-order RK scheme with $dt_G = 40dt_F$, and including only a reduced k-space of size 201×201 harmonics (i.e., $k_{\max} = 100$). The serial time of the simulation (that has a total length of $T = 7,040dt_F$) using the fine solver is 2.62 hours on the *pingo* supercomputer at the Arctic Research Supercomputing Center (ARSC) at Fairbanks. The coarse solver $\mathbf{G}$ is about 200 times faster than $\mathbf{F}$ (in fact, $\beta = 210.3$). Using $N = 88$ processors (which corresponds to time chunks of size $\Delta T = 80dt_F$), the parareal calculation converges in only $K = 5$ cycles (see inset in Fig. 3.4), requiring only 0.29 hours. That is, the parareal scheme yields a speed-up $H_{PA} = 8.80$. In Fig. 3.5, we also show the time history of the energy contained in one of the low-k modes (in particular, the $(k_x, k_y) = (0, -1)$ mode), to show that convergence of the total energy also implies convergence of the individual harmonics as previously stated. Similar behavior is also observed at modes with higher k's, at least until the contribution to the total energy of the mode gets of the order or smaller than the required tolerance. Thus, there is no doubt that the parareal method can be applied successfully to fully-developed turbulent simulations.

3.5.2 Phenomenological model for $k(N,T)$ for the BETA model

We now carry out the first step in the determination of two unknowns of the performance model from BETA simulations: $k_s(N)$ (strong scaling) and $k_w(N)$ (weak scaling). That step is the building of a model for $k(N,T)$, the number of parareal cycles needed to converge a simulation of length T using N processors (the chunk size is then $\Delta T = T/N$).

We proceed as follows. First, we use the simulations to 'measure' the convergence rate function, $\Delta n(k;N,T)$, defined as the number of new, successive time chunks converged during the parareal cycle k. We have collected this information for many runs with various values of length T and number of processors N, and using various implementations for the coarse solver **G**. Interestingly, all of them seem to follow the same pattern, that is sketched in Fig. 3.6: at first only one chunk (the minimum possible, since that is the rate at which the **F** solution advances!) is converged per cycle for the first k_1 parareal cycles. Then, after $k > k_2$ cycles ($k_2 > k_1$), the convergence rate is again roughly constant, but equal to some value $B > 1$. In the intermediate region, i.e. for cycles $k_1 < k < k_2$, we assume that the convergence rate is described by a linear model. That is:

$$\Delta n(k;N,T) \simeq \left\{ \begin{array}{cc} 1, & k \leq k_1 \\ \\ 1 + \left(\dfrac{B-1}{k_2-k_1}\right)(k-k_1), & k_1 < k < k_2 \\ \\ B, & k \geq k_2 \end{array} \right\}. \tag{3.14}$$

where the slope of the linear part is $m = (B-1)/(k_2-k_1)$. To find $k(N,T)$ from Eq.

(3.14), one first computes $n(k;N,T)$, the accumulated number of converged time chunks via integration of $\Delta n(k;N,T)$:

$$n(k;N,T) = \left\{ \begin{array}{cc} k, & k \leq k_1 \\ k + \frac{1}{2}\left(\frac{B-1}{k_2-k_1}\right)(k-k_1)^2, & k_1 < k < k_2 \\ Bk - \frac{B-1}{2}(k_2+k_1), & k \geq k_2 \end{array} \right\}. \tag{3.15}$$

Finally, one inverts $n(k;N,T) = N$ to get:

$$k(N,T) = \left\{ \begin{array}{cc} N, & N \leq N_1 \\ k_1 + \frac{k_2-k_1}{B-1}\left\{\sqrt{1+2(B-1)\frac{N-k_1}{k_2-k_1}} - 1\right\}, & N_1 < N < N_2 \\ \frac{N}{B} + \frac{B-1}{2B}(k_2+k_1), & N \geq N_2 \end{array} \right\}. \tag{3.16}$$

where $N_1 = k_1$ and $N_2 = [(B+1)k_2 - (B-1)k_1]/2$.

However, Eq. (3.16) is still not very useful at this point because the values of k_1, k_2 and B change with T and N, *even when the coarse solver* **G** *remains unchanged.* An important observation at this point is that these values seem to be roughly constant when the simulations keep ΔT fixed and use the same coarse solver. For example, Fig. 3.7 shows the convergence rate functions $\Delta n(k;T,N)$ for various runs done using as **G** the same VODPK stiff solver as for the fine solver **F**, but including only 145×145 modes (i.e., $k_{\max} = 72$) and $dt_g = 4dt_F$ (i.e., $\beta = 16.3$) [Of course, there are important fluctuations around the mean values, but that is also a consequence of the fact that one must take discrete derivatives to compute them.]. This observation suggests that maybe one can find more universal quantities if k_1, k_2 and B are expressed in physical time units. And indeed, we have found that for each fixed solver **G**, $t_1 \equiv N(k_1) \cdot \Delta T$, $t_2 \equiv N(k_2) \cdot \Delta T$ and $b = B \cdot \Delta T$ (that is, expressed in units of dt_F, our physical timescale) are roughly the same for all the simulations. An example is shown in Fig. 3.8. The inset shows the convergence rate functions $\Delta n(k)$ obtained again with the VODPK stiff solver as **G** for simulations with various chunk sizes and number of processors. They have a general shape similar to that sketched in Fig. 3.6, but the parameter values are all different. However, when expressed using physical units, all of them collapse to the same universal curve. What these observations suggest is that there appears to be two timescales (t_1 and t_2) that must be resolved before the scheme can transition from a slow (i.e., 1 chunk per cycle) to a fast convergence rate b which is independent of ΔT, or T, or N, once the coarse solver is chosen.

Thus, the values of $\{t_1, t_2, b, \beta\}$ seem to be a good practical way to characterize each $\mathbf{G}$ in the parareal context. We have collected in Table 3.1 the values of these quantities for the three coarse solvers discussed in this paper; 1) the VODPK solver with $k_{\max} = 72$ and $dt_G = 4dt_F$ just discussed; 2) a 4th-order RK with $k_{\max} = 100$ and $dt_G = 40dt_F$; and 3) a 2nd-order RK with $k_{\max} = 100$ and $dt_G = 8dt_F$. The last two coarse solvers will be used in the strong and weak scaling sections. Once t_1, t_2 and b are known (for the coarse solver $\mathbf{G}$ chosen), the model for $k(N,T)$ given by Eq. (3.16) can be completed via the inverse relations:

$$k_1(N,T) = \frac{t_1 N}{T}, \quad B(N,T) = \frac{bN}{T}, \tag{3.17}$$

$$k_2(N,T) = \left[2\frac{t_2 N}{T} + \left(\frac{bN}{T} - 1\right)\frac{t_1 N}{T}\right] / \left(\frac{bN}{T} + 1\right). \tag{3.18}$$

3.5.3 Strong scaling study

We are now in a position to discuss the strong scaling properties of the parareal method when applied to the BETA runs. We start by deriving an expression for $k_s(N)$ from Eq. (3.16). To do that, it is sufficient to require that T, *the length of the simulation, be fixed.* The result depends on how T compares with the other two timescales present, t_1 and t_2:

$$k_s(N) = \left\{ \begin{array}{cc} N, & T < t_1 \\ \\ \frac{t_1}{T}N + \Sigma(t_1, t_2, b, T, N), & t_1 < T < t_2 \\ \\ \frac{T}{b} + \frac{\frac{bN}{T} - 1}{\frac{bN}{T} + 1}\left(\frac{t_2}{b} + \frac{t_1}{T}N\right), & T \geq t_2 \end{array} \right\}. \tag{3.19}$$

where the complicated function for the transition range of $T's$ is given by:

$$\Sigma(t_1,t_2,b,T,N) \equiv \frac{2(t_2-t_1)\frac{N}{T}}{\left(\frac{bN}{T}\right)^2-1}\left\{\sqrt{1+\left(\left(\frac{bN}{T}\right)^2-1\right)\frac{(T-t_1)}{t_2-t_1}}-1\right\}. \tag{3.20}$$

Although we will use the full Eq. (3.19) to compute the theoretical gains to compare with the simulation results, it is useful to examine this equation in several limits. First, one easily notes that if the simulation is so short that $T < t_1$ then parareal fails completely since it will converge at a rate of one chunk per cycle, thus needing $K = N$ to converge. Indeed, the speed-up factor (Eq. (3.10)) becomes,

$$H_{PA}^{T<t_1}(N) = \left(\frac{\beta}{N+\beta}\right) < 1. \tag{3.21}$$

Luckily, most actual applications will be in the opposite limit, $T >> t_2$ (since the interest is in doing very long simulations of transport dynamics). Note that the imprint of having to resolve t_1 one chunk at a time is still present in Eq. (3.10), (appears as t_1N/T), but it is not always deleterious for performance. In fact, its effect is small as long as $N << N_s \equiv T^2/bt_1$, which separates one regime in which the parareal cycle converges roughly in a constant number of cycles from another where the number of cycles required increases with N. Clearly, the transition occurs when N gets large enough so that the resolution of t_1 dominates the convergence process. But luckily, note that this transition occurs at a rather large number of processors in practical cases, since $N_s \propto T^2$. Assuming $bN/T >> 1$, $\beta << N_s$ and $T >> t_2$, conditions that will be almost always satisfied in very long simulations, three asymptotic regimes can be identified:

$$H_{PA}^{T>>t_2}(N) \sim \left\{\begin{array}{ll} \left(\frac{b}{T}\right)N, & N << \beta \\ \frac{b\beta}{T}, & \beta << N << N_s \\ \left(\frac{\beta T}{t_1}\right)\frac{1}{N}, & N >> N_s \end{array}\right\}. \tag{3.22}$$

That is, the speed-up of the strong scaling increases first linearly with N as in the spatial parallelization ideal scaling, but note that the slope is in general smaller than unity, b/T. The other aspect that is quite different from the spatial scaling laws is that, for a wide range of processors (between β and N_s), the speed-up saturates at a roughly constant maximum value, $\sim b\beta/T$. Thus, $N \sim \beta$ sets the value of N at which the speed-up no longer improves as we predicted, but it may take a much larger number of processors to observe deterioration of the wallclock time. The reason is that, in this regime, the wallclock time of the calculation is completely dominated by the number of times that the coarse solution with **G** must be applied serially! The efficiency of the parallel part of the algorithm is of no importance at this stage. Finally, when the resolution of t_1 at one chunk at a time becomes the dominant process, the speed-up deteriorates as $\sim 1/N$.

The analytical considerations compare well with actual numerical calculations of a strong scaling sequence, although the asymptotic formulas tend to overestimate actual results. As an illustration we present two such numerical strong scaling studies in Figs. 3.9 and 3.10. The first study corresponds to a BETA simulation with length $T = 6,400 dt_F$, using as coarse solver the 2^{nd}-order RK previously mentioned (see Table 3.1). The aforementioned linear phase is observed up to $N \sim 100$. The maximum speed-up obtained in the simulations is slightly below 11 using $N = 400$ processors, and it should be noted that the speed-up varies very little above $N \simeq 100$. For this case, the theoretical values obtained are $\beta \sim 84.29$ and $N_s \sim 800$, again in agreement with the simulations. However, note that the saturated speed-up predicted by the asymptotic formula Eq. (3.22) is $H_{PA}^{\max} \sim 17.6$, which is larger than the one observed. The reason for the discrepancy must be sought in the fact that the conditions to derive Eq. (3.22) are not satisfied in this example, since the simulation is too short ant T is only slightly larger than t_2 (see Table 3.1). In fact, when using the full model for $k_s(N)$, the agreement is much better, as shown in Fig. 3.9.

The second example corresponds to a BETA simulation with length $T = 25,600 dt_F$, that has been done using as coarse solver the 4^{th} order RK previously mentioned (see Table 3.1). The maximum speed-up observed in this case is slightly above 5, using $N \simeq 300$ processors, but it should be noted that the speed-up value does not seem to be yet close to saturation. And indeed, one finds from the asymptotic formulas that $\beta \sim 210.3$ and $N_s \sim 3,000$ in this case, with a maximum speed-up of roughly $H_{PA}^{\max} \sim 10$. We thus expect that we could do somewhat better with more processors in this second case.

3.5.4 Weak scaling

We now proceed to analyze the weak-scaling properties of the parareal scheme implemented in BETA. The figure-of-merit here is, as previously described, the amount of work per processor W_{PA} defined in Eq. (3.11). To obtain $k_w(N)$ it is sufficient to assume that ΔT is fixed, and that the length of the simulation is $T = N \cdot \Delta T$. In this case, the process is trivial because all the relevant quantities,

$$k_1 = \frac{t_1}{\Delta T}, \quad B = \frac{b}{\Delta T}, \tag{3.23}$$

$$k_2 = \left[2\frac{t_2}{\Delta T} + \left(\frac{b}{\Delta T} - 1 \right) \frac{t_1}{\Delta T} \right] / \left(\frac{b}{\Delta T} + 1 \right) \tag{3.24}$$

are all constants that do not change when the number of processors. Thus, $k_w(N)$ is given exactly by Eq. (3.16).

As in the case of the strong scaling, asymptotic formulas can be derived to better interpret the results. Two regimes can be distinguished depending on the value of T. First, we consider the case when N is not large enough to resolve t_1, $N < t_1/\Delta T$. Then, $k_W(N) = N$, and the work-per-processor is:

$$W_{PA}^{T<t_1}(N) = N\left(1 + \frac{N}{\beta}\right) > N, \quad N < t_1/\Delta T. \tag{3.25}$$

which is larger than N, signaling a complete failure of parallelism, as expected.

But again the limit of interest for applications is when N is large enough so that $T >> t_2$. Then, assuming $T >> t_2$ and $b > \Delta T$, the resulting work-per-processor becomes:

$$W^{T>>t_2}(N) \sim \left\{ \left(\frac{\Delta T}{b} \right) N + \frac{t_1}{\Delta T} \right\} \left(1 + \frac{N}{\beta} \right) \tag{3.26}$$

As in the strong limit case, the imprint of having to resolve t_1 still appears here (the $t_1/\Delta T$ term). But now, since ΔT is prescribed, this term will be unimportant once $N >> N_w \equiv (bt_1)/(\Delta T)^2$. Thus, assuming also that $\beta >> N_w$, one can easily distinguish three asymptotic regimes:

$$W_{PA}^{T\gg t_2}(N) \sim \left\{ \begin{array}{ll} \dfrac{t_1}{\Delta T}, & \dfrac{t_1}{\Delta T} \ll N \ll N_w \\ \left(\dfrac{\Delta T}{b}\right) N, & N_w \ll N \ll \beta \\ \left(\dfrac{\Delta T}{b\beta}\right) N^2, & N \gg \beta \end{array} \right\}. \qquad (3.27)$$

That is, the work-per-processor is first constant as in the ideal spatial parallelism case, although the value is usually greater than unity. And it does not imply good parallelism: only that the fixed amount of work needed to resolve t_1 dominates the calculation. But once $N > N_w$, the work-per-processor begins to increase linearly, in contrast to the spatial case. The slope is simply given by $\Delta T/b$, as expected, since $(\Delta T/b)N$ gives roughly the number of parareal cycles needed to converge in this regime, and the number of times each processor has to repeat the serial work. Finally, for $N \gg \beta$, things deteriorate and the work-per-processor increases quadratically with N.

To illustrate this behavior, we have also carried out a numerical weak scaling study using as coarse solver the 4^{th} order RK previously mentioned and with a prescribed chunk size $\Delta T = 80dt_F$. The work-per-processor as a function of the number of processors is shown in Fig. 3.12. The first two regimes of Eq. (3.27) can be clearly seen in the figure. First, the work-per-processor stays roughly constant at about ~ 4 up to $N \sim 40$. Then, it increases close to linearly at least until $N \sim 300-400$ with a slope $\sim 0.10 < 1$, indicating that weak parallelism is different from the ideal one but not that bad. It is interesting to compare these numbers with the predictions of the asymptotic formulas. For these runs, the asymptotic expressions just derived predict that $N_w \sim 17$ and $\beta \sim 210.3$, consistent with the extent of the linear scaling in W seen in the numerical observations. The slope predicted by the model would be ~ 0.06. On the other hand, the constant value at the smallest N would be ~ 1 or 2. Again, as in the strong scaling case, these formulas tend to overestimate performance slightly. For completeness, we have also included a figure with the speed-up of the calculations, although note that the length of the simulation is increasing with N. The largest speed-up observed in this series is

~ 9 using $N = 88$ processors, corresponding to the case cited as example at the beginning of this section.

3.6 Conclusion

The two main conclusions of this work are that turbulent simulations can be parallelized in time using the parareal algorithm, and that rather sizeable gains are achievable with this method. Indeed, we have shown speed-ups close to an order of magnitude, and even these could be further increased by further optimization of the coarse solver. As we mentioned several times, this is a remarkable result in view of the many existing concerns regarding the effect of the sensitivity to initial conditions and exponential Lagrangian divergence on the convergence of the method. In fact, it has turned out that parareal may probably work better here than in non-turbulent systems, since convergence only needs to be enforced on a single scalar quantity (here, the total energy), and the whole spectrum (containing hundreds of thousands of degrees of freedom) also converges even when unforced (at least, up to modes with an energy larger than the required convergence tolerance) due to the existence of strong nonlinear couplings between any pair of modes. This is indeed a pretty remarkable result.

In this work we have also shown that it is possible to select a coarse solver that works for this type of problems based on using a reduced grid size in k-space, accompanied by a coarser timestep and a simpler time-advance scheme. We have also characterized the performance of the method using a model function for $k(N,T)$, the number of parareal cycles required to converge that, for the BETA simulations examined here, depends on four quantities: β, t_1, t_2 and b. Clearly, performance depends more sensitively on β and b. The larger they are, the larger the parallel speed-up will be. A matter of investigation for the future will be to investigate the origin and meaning of all these parameters. That is, to clarify whether their values are imposed by the physics of the underlying turbulence or by the choice of the coarse solver and its relation with the fine one. This knowledge would allow us to further optimize the parareal implementation inside BETA to obtain further gains. More importantly for other applications, we also plan to repeat the current exercise in other turbulent models, in order to see if the model function found phenomenologically for $k(N,T)$ has a universal character, or whether the shape used here is only particular to the DTEM case and the $\mathbf{E} \times \mathbf{B}$-nonlinearity.

Other possibilities can and should also be explored in the future for further improvement in the gain and efficiency. For instance, a better **G** (i.e., larger β and b) than the ones used here may be an option. It may be noted here, that a faster **G** (i.e., larger β) will not necessarily mean a better one, unless the number of iterations required for convergence remains small enough (i.e., larger b as well) so that the total time for simulation is reduced. A combination of both space and time parallelization, thus leading to hybridization of the parallel scheme, should certainly be explored, since it may help to better utilize already existing resources. Also, it is worth noting that the current implementation of the parareal algorithm requires processors where convergence has been observed to remain idle until the solution along the entire time domain is converged. Reuse of idle processors may thus enhance the efficiency of the parareal scheme.

3.7 Acknowledgements

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. Research was also carried out in part at Oak Ridge National Laboratory, managed by UT-Battelle LLC, for U.S. DOE under Contract No. DE-AC05-00OR22725, and funded via the Seed Money Initiative Program. For simulation runs, the authors are thankful for grants for the use of supercomputing resources at the University of Alaska's Arctic Region Supercomputing Center (ARSC) in Fairbanks. We also acknowledge valuable discussions with L. Chacon, L.A. Berry and W. Elwasif at ORNL.

Table 3.1. Values of parameters defining $k(N)$ model in physical units (see Eq. (3.16)) for the three different coarse solvers discussed in Section 3.5.

G	$k_{\max}$	dt_F/dt_G	β	t_1	t_2	b
VODPK	72	4	16.3	290 ± 60	$4{,}120 \pm 500$	$1{,}320 \pm 280$
RK2	100	8	84.3	32 ± 32	$3{,}100 \pm 1{,}600$	$1{,}340 \pm 480$
RK4	100	40	210.3	80 ± 80	$2{,}560 \pm 1{,}000$	$1{,}280 \pm 480$

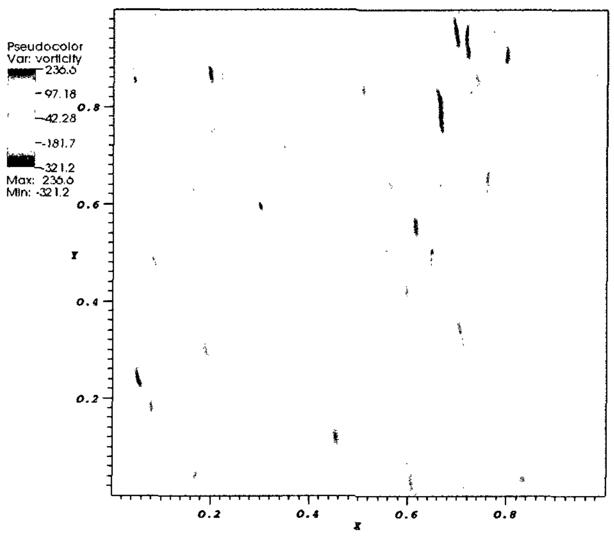


Figure 3.1. Snapshot of vorticity for a typical BETA long-wavelength DTEM simulation.

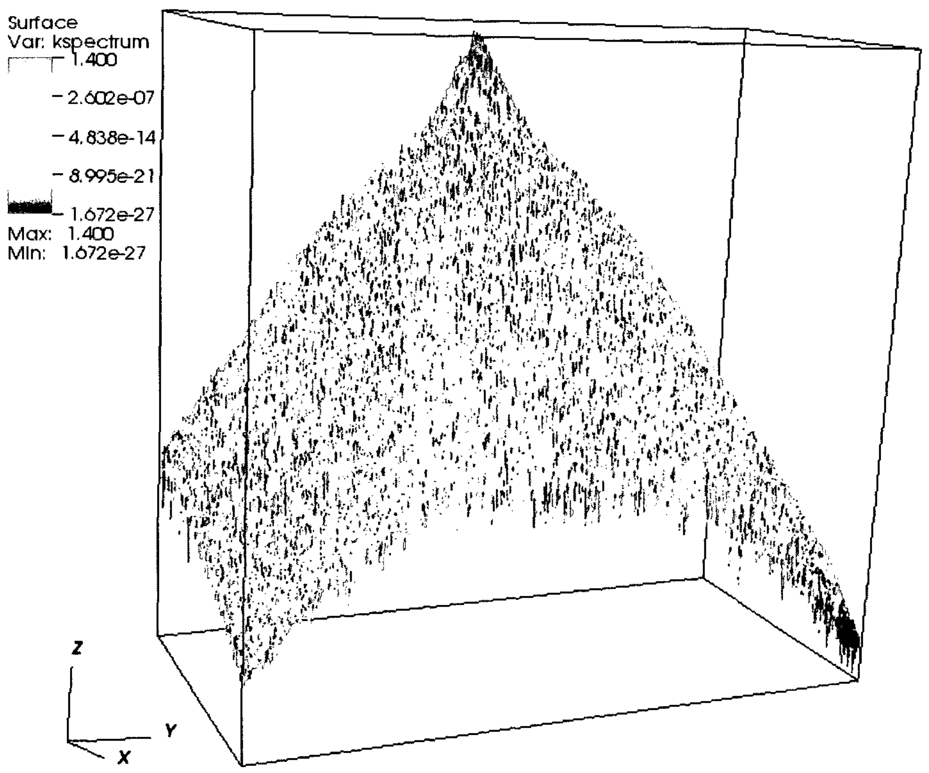


Figure 3.2. Power spectrum (in lin-log scale) for the same BETA simulation shown in Fig. 3.1.

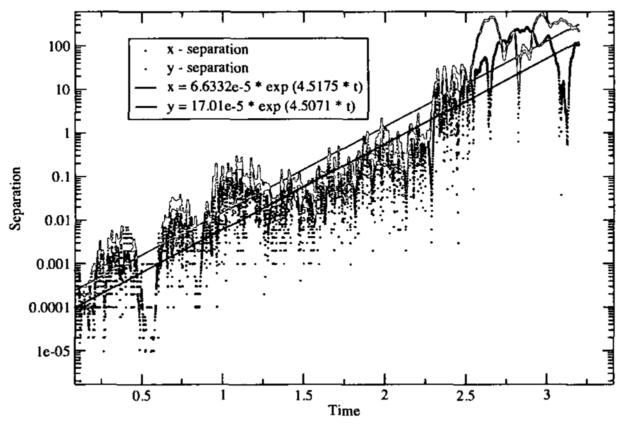


Figure 3.3. Separation of two initially close trajectories as a function of time for the same BETA simulation shown in Fig. 3.1. Both x (black) and y (red) directions are included.

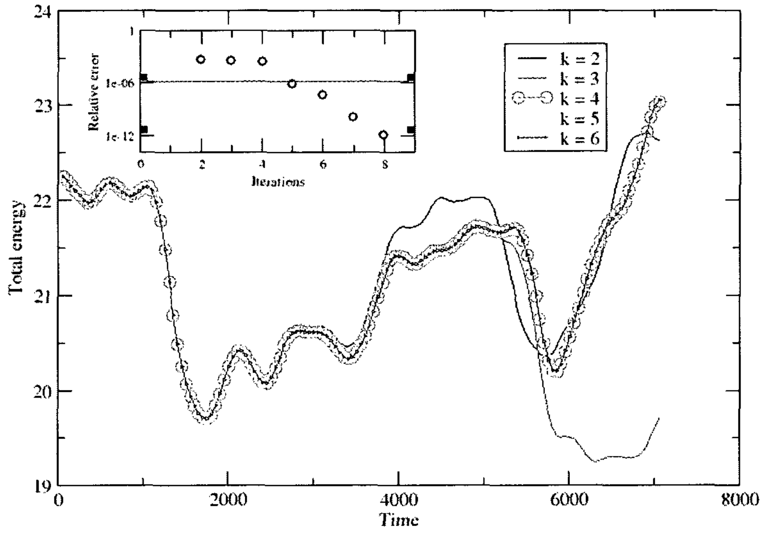


Figure 3.4. Total energy as a funtion of time for a BETA run using $N = 88$ processors and chunk size $\Delta T = 80$ as a function of the parareal cycle index k. Coarse solver is a 4^{th} order Runge-Kutta described in Section 3.5. Convergence is observed at $k = 5$. **Inset:** Relative error averaged over all chunks as a function of k.

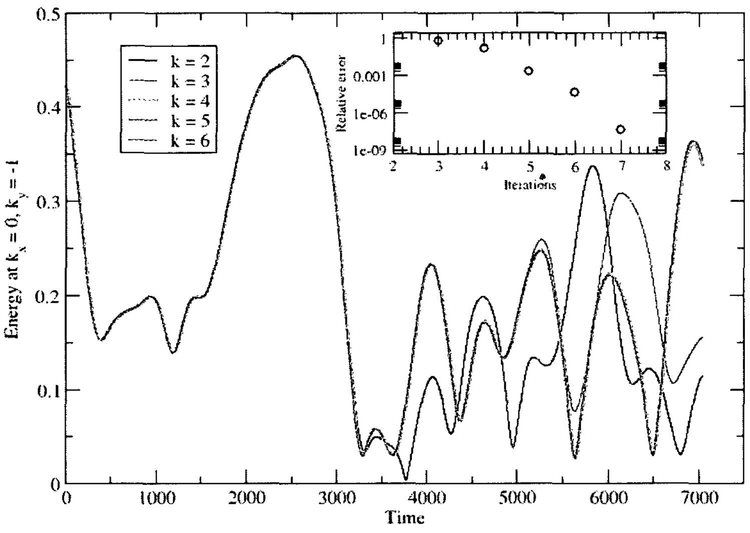


Figure 3.5. Convergence history of the energy contained in the low-k mode $(k_x, k_y) = (0, -1)$ as a function of the parareal cycle index k for the same run as Fig. 3.4. **Inset:** Relative error averaged over all chunks as a function of k.

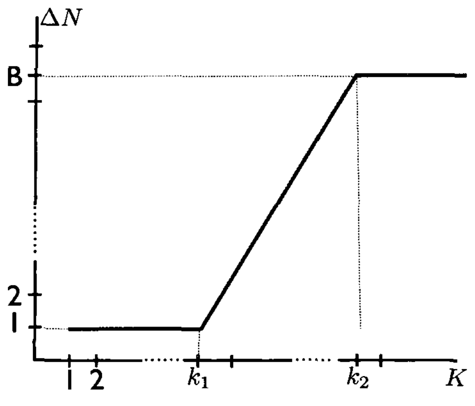


Figure 3.6. Sketch showing the typical features of the convergence rate function $\Delta n(k; T, N)$ observed in the BETA simulations discussed in Section 3.5.

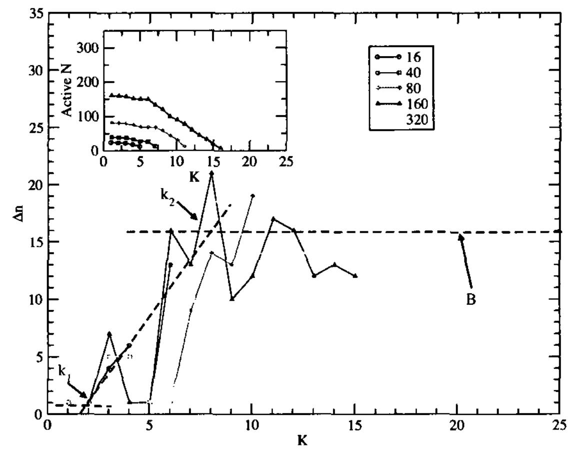


Figure 3.7. Example of convergence rate function $\Delta n(k;T,N)$ obtained with the VODPK coarse solver described in Section 3.5 and fixed chunk size $\Delta T = 80dt_F$. Number of processors used is shown in the legend. **Inset**: number of unconverged chunks as a function of parareal cycle index k for the same cases.

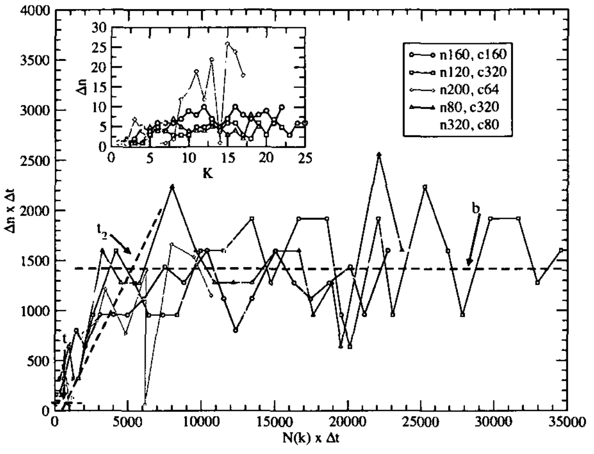


Figure 3.8. Convergence rate in physical time

Collapse of convergence rate curves when expressed in physical time for parareal BETA runs using the same coarse solver (VODPK, described in Section 3.5) and varying values of processors (n in legend) and chunk size ΔT (c in legend). **Inset:** Same convergence rate curves when expressed in parareal cycles as in Fig. 3.7.

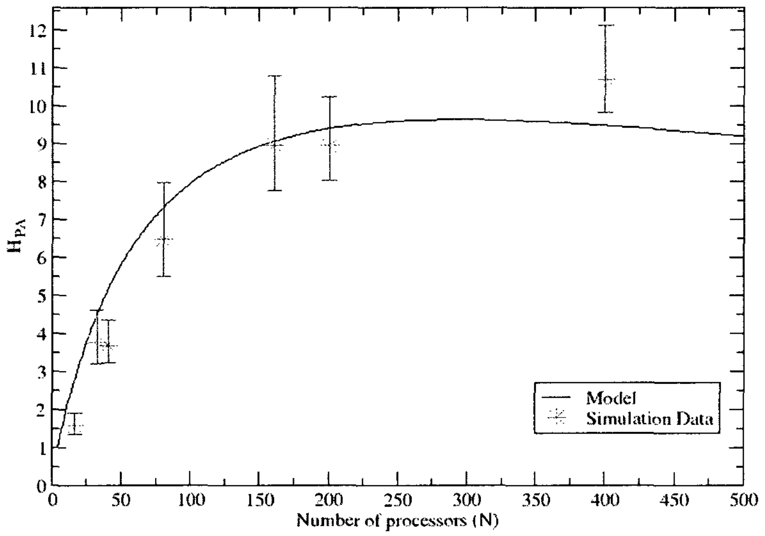


Figure 3.9. Results of strong scaling study using as coarse solver **G** the 2^{nd}-order Runge-Kutta solver described in Section 3.5. Model curve for gain corresponds to values from Table 3.1.

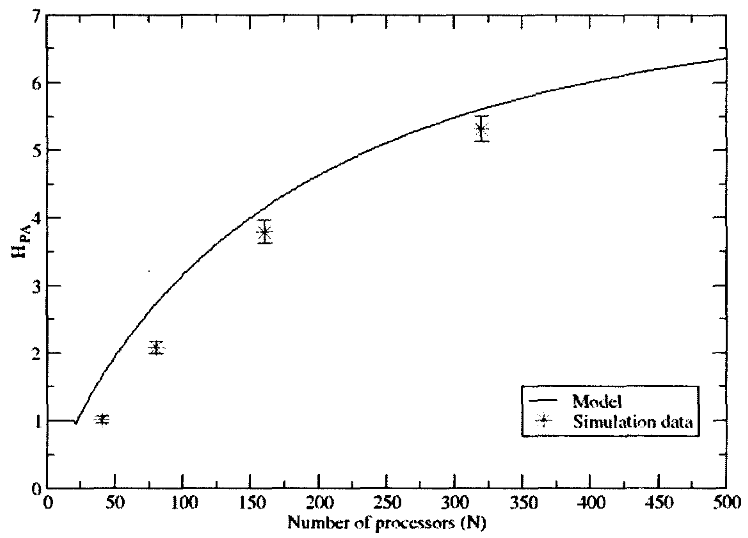


Figure 3.10. Results of strong scaling study using as coarse solver G the 4th-order Runge-Kutta solver described in Section 3.5. Model curve for gain corresponds to values from Table 3.1.

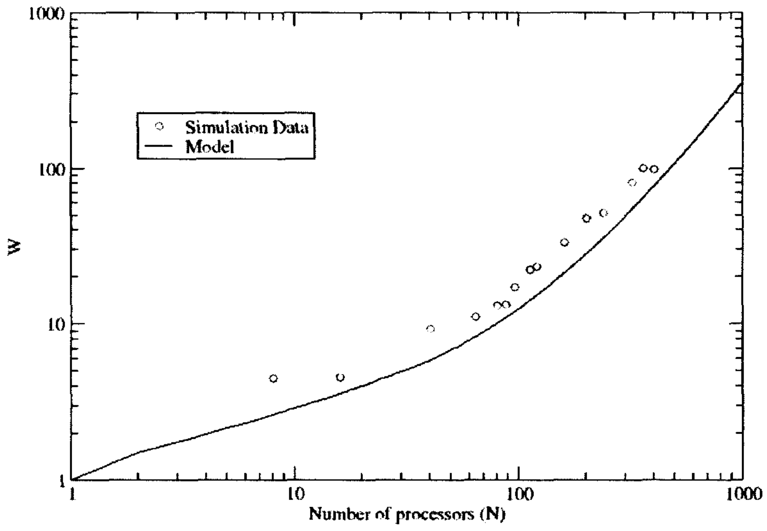


Figure 3.11. Weak scaling study for series of runs done using the 4th-order Runge-Kutta solver described in Section 3.5. Model curve corresponds to values from Table 3.1.

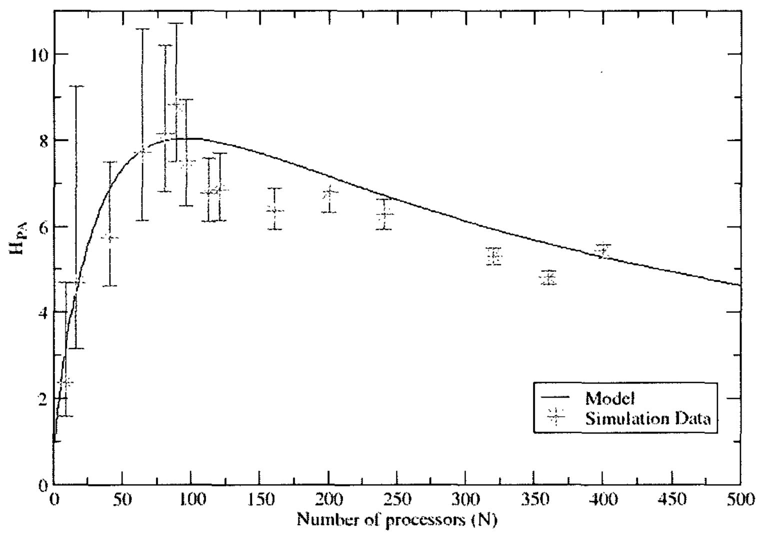


Figure 3.12. Parallel gain for weak scaling series of runs using the 4^{th}-order Runge-Kutta solver described in Section 3.5. Model curve plotted corresponds to values from Table 3.1.

Bibliography

[1] J. Dahlburg, J. Corones, D. Batchelor, R. Bramley, M. Greenwald, S. Jardin, S. Krasheninikov, A. Laub, J. N. Leboeuf, J. Lindl, W. Lokke, M. Rosenbluth, D. Ross, D. Schnack, Fusion simulation project: Integrated simulation and optimization of magnetic fusion systems, J. Fus. Energy 20 (4) (2001) 135.196.

[2] B. A. Carreras, D. E. Newman, V. E. Lynch, P. H. Diamond, A model realization of self-organized criticality for plasma confinement, Phys. Plasmas 3 (8) (1996) 2903-2912.

[3] R. Sanchez, D. E. Newman, J. N. Leboeuf, V. K. Decyk, B. A. Carreras, Nature of transport across sheared flows in electrostatic, ion-temperature-gradient driven gyrokinetic plasma turbulence, Phys. Rev. Lett. 101 (2008) 205002.

[4] J. Nivergelt, Parallel methods for integrating ordinary differential equations, Commun. ACM 7 (12) (1964) 731-733.

[5] A. Bellen, M. Zennaro, Parallel algorithms for initial value problems for difference and differential equations, Phys. Rev. E 25 (3) (1989) 341-350.

[6] P. Chartier and B. Philippe, A parallel shooting technique for solving dissipative ODEs, Computing, 51 (3-4) (1993) 209-236.

[7] M. Kiehl, Parallel Comput., 20 (3) (1994) 275-295.

[8] W. L. Miranker and W. Liniger, Math. Comp., 21 (99) (1967) 303-320.

[9] J. H. Saltz and V. K. Naik, Parallel Comput., 6 (1) (1988) 19-44.

[10] D. E. Womble, Siam J. Sci. Stat. Comput., 11 (5) (1990) 824-837.

[11] C. Farhat and M. Chandersis, Internat. J. Numer. Methods Engng., 58 (9) (2003) 1397-1434.

[12] C. Farhat et al., Internat. J. Numer. Methods Engng., 67 (5) (2006) 697-724.

[13] J. Cortial and C. Farhat, Internat. J. Numer. Methods Engng., 77 (4) (2008) 451-470.

[14] J. Lions,Y. Maday,G. Turinici, C. R. Acad. Sci. paris-Serié 332 (7) (2001) 661-668.

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

[16] G. Bal and Y. Maday, A parareal time discretization for nonlinear PDEs with application to the pricing of an American put, Vol. 23 of Lect. Notes Comput. Sci. Eng., Springer, Berlin, 2002, pp. 189-202.

[17] G. Staff and G. Ronquist, Stability of the parareal algorithm, in: Proceedings of Fifteen International Conference on Domain Decomposition Methods, Springer Verlag, 2003, pp. 449-456.

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

[19] I. Garrido, M. S. Espedal, G. E. Fladmark, A convergent algorithm for time parallelization applied to reservoir simulation, in: Proceedings of Fifteen International Conference on Domain Decomposition Methods, Vol. 40, Springer, Berlin, 2004, pp. 469-476.

[20] P. Fischer, F. Hecht, Y. Maday, A parareal in time semi-implicit approximation of the navier-stokes equations, in: Proceedings of Fifteen International Conference on Domain Decomposition Methods, Vol. 40, Springer Verlag, 2004, pp. 433-440.

[21] Y. Liu and J. Hu, Modified propagators of parareal in time algorithm and application to Princeton Ocean model, Internat. J. Numer. Methods Fluids 57 (12) (2008) 1793-1804.

[22] M. J. Gander and E. Hairer, Nonlinear convergence analysis for the parareal algorithm, in: Domain Decomposition Methods in Science and Engineering XVII, Vol. 60, Springer, 2008, pp. 45-56.

[23] D. E. Newman, P. W. Terry, and P. H. Diamond, Phys. Fluids B, 4 (3) (1992) 599-610.

[24] Y. -M. Liang, P. H. Diamond, X. H. Wang, D. E. Newman, P. W. Terry, A two nonlinearity model of dissipative drift wave turbulence, Phys. Fluids B 5 (4) (1993) 1128-1139.

[25] B. A. Carreras, K. Sidikman, P. H. Diamond, P. W. Terry, L. Garcia, Theory of shear flow effects on long-wavelength drift wave turbulence, Phys. Fluids B 4 (10) (1992) 3115-3131.

[26] J. A. Mier, R. Sanchez, L. Garcia, D. E. Newman, B. A. Carreras, On the nature of transport in near-critical dissipative-trapped-electron-mode turbulence: Effect of a subdominant diffusive channel, Phys. Plasmas 15 (11) (2008) 112301.

[27] F. Takens, Detecting Strange Attractors in Turbulence, Vol. 898 of Lecture Notes in Mathematics, Springer, Berlin, 1981, p. 366.

[28] A. Hindsmarsh, Gear: Ordinary equation system solver, in: Report UCID-30001, Rev. 3, LLNL, 1974.

Chapter 4
Extension of the parareal technique to new nonlinearities and analysis of convergence for the case of a turbulent system [1]

Abstract

In this work we show that time parallelization utilizing the parareal algorithm can be successfully applied to a wide variety of turbulent systems from general plasma turbulence to neutral fluid turbulence. The algorithm is successfully applied to a drift wave turbulence model with polarization drift nonlinearity which is isomorphic to the quasi-geostrophic, beta-plane models for geophysical fluids. The algorithm is also successfully implemented in a turbulent system containing both the polarization and $\mathbf{E} \times \mathbf{B}$ nonlinearities. The computational gain obtained from these simulations show great promise as shown by an analysis of the convergence properties. It is found that the parameters governing convergence depend not only on the numerics, but on the dynamics of the system as well.

4.1 Introduction

Since its introduction in 2001 by *Lions et al.* [1], the parareal algorithm has found application in many numerical problems. However, until [2], the algorithm had not been successfully applied, to the best of our knowledge, to any simulation of fully developed turbulence. Prior to [2], the algorithm had only been applied to problems of much less complexity, like molecular dynamics [3], reservoir simulations [4], the laminar region of the Navier-Stokes equations [5], nonlinear partial differential equations (pdes, namely the Black-Scholes equations) [6], stochastic ordinary differential equations [7] and other nonlinear problems such as the Viscous Burgers equation and the Lorenz equations [8]. The work on the Princeton Ocean Model by [9], where convection was the dominant factor, reported modest success. The stability and convergence of the parareal scheme for systems similar to the ones mentioned above have also received significant attention, for example [10] presented a detailed study of the stability of the technique. The stability and

[1] D. Samaddar, D.E. Newman and R. Sánchez. 2010. Extension of the parareal technique to new nonlinearities and analysis of convergence for the case of a turbulent system. Prepared for submission in Physics of Plasmas.

convergence properties for time dependent equations with constant coefficients was analyzed in [11]. Convergence properties of the algorithm for bounded and unbounded problems were more recently explored in [12].

The success of the parareal technique for a turbulence simulation is perhaps surprising in many ways. Turbulence is characterized by extremely high sensitivity to initial values and exponential divergence of lagrangian trajectories [13], [14]. The parareal algorithm employs a predictor-corrector method to attain temporal parallelization, despite the sequential nature of the time domain. Since the solutions for different parts of the time series are obtained in parallel on different processors, convergence may be unexpected for a highly complex system such as turbulence where there are very strong nonlinear couplings between disparate scales. However, convergence, accompanied by a computational gain of almost an order of magnitude was demonstrated in [2].

As has already been pointed out in [2], convergence in a turbulent system may be possible due to the strong nonlinear couplings, which sets the relative energy fluxes between any pair of modes within the system. This guarantees that any one quantity (the total energy of the system, for example) converges, when all other quatities converge as well [15]. This characteristic may actually enhance the possibility of convergence with the application of the parareal algorithm for a turbulent system in comparison to a linear system.

This chapter extends the parareal technique to new nonlinearities compared to what was observed in [2]. A dissipative trapped electron mode (DTEM) model for drift wave turbulence has been used as the test bed for the parareal algorithm in [2] as well as in this chapter. The model has doubly periodic slab geometry with the $\mathbf{E} \times \mathbf{B}$ nonlinearity. The physics of the model had already been extensively studied in [16], [17] and [18]. The model of plasma drift wave turbulence used for this new application is different from the one used in [2] due to the fact that here, the $\mathbf{E} \times \mathbf{B}$ is replaced by the polarization drift nonlinearity. This new model has already been derived and studied in [17]. The polarization drift nonlinearity is the nonlinear term in the Hasegawa-Mima equation and is isomorphic to the Navier-Stokes nonlinearity found in neutral fluids. The parareal technique has also been shown to work for a model containing both the $\mathbf{E} \times \mathbf{B}$ and polarization nonlinearities.

This chapter demonstrates that a computational gain of almost an order of magnitude can be obtained by the application of the parareal scheme to two new models of plasma drift wave turbulence. This further establishes the utility of time parallelization in turbulence simulations.

A phenomenological model for the convergence rate for the application of the parareal algorithm in the turbulence model with the $\mathbf{E} \times \mathbf{B}$ was presented in [2]. This model is further explored in this chapter by characterizing the various parameters that control the convergence rate in this system.

To attain time parallelization, the parareal scheme employs two solvers, the coarse solver (used in serial) and the fine solver (used in parallel). The mathematical formulation about the boundedness of norm of the difference between the two solvers as mentioned in [1] is very difficult to implement in a physical problem. As a result, the choice of the coarse solver, to date, has primarily been based on trial and error for all the applications of the parareal algorithm. This has also been the case in the application of the parareal algorithm to turbulence. This paper attempts to describe the choice of an appropriate coarse solver for the model explored in [2]. The fact that the gain and success of the parareal algorithm also depends largely on the physics of the turbulent system is discussed in this paper.

The rest of this paper is organized as follows. Section 4.2 gives an overview of the parareal technique, accompanied by the modification presented in [2]. Section 4.3 briefly describes the models studied in this paper, followed by results obtained by applying the parareal algorithm to them. Section 4.4 explores the parameters that set the convergence rate with the help of the model already introduced in [2]. Section 4.5 describes the different coarse solvers that have been explored for the application of the parareal scheme to turbulence. Section 4.6 finally summarizes all the results.

4.2 The parareal algorithm : a review

The parareal algorithm is based on a predictor-corrector scheme. For the time dependent ordinary differential equation

$$\frac{d\lambda}{dt} = A(\lambda, t), \quad \lambda(0) = \lambda_0 \tag{4.1}$$

let λ be a function of time, t, with its initial value at $t = 0$ being given as λ_0. Let A be a function operating on λ and t. It is also assumed that for Eq. (4.1), there exist two propagators that advance $\lambda(t)$ to $\lambda(t+dt)$. Of the two propagators, F is the fine solver, giving a more correct solution for the problem, but is computationally very slow. G, on the other hand, is the coarse solver, gives a less accurate solution, but is computationally fast. F is always applied in parallel, and G is a serial procedure.

For the parareal scheme, the total simulation time, T, is divided into N different parts or chunks, defined as $\Delta T = T/N$. The value of λ across each such chunk is separately solved on individual processors, in parallel. Let there be N processors denoted by $P_0, P_1, P_2, P_3, ..., P_{N-1}$ to evolve λ through the N different chunks. If k represents the parareal iteration and i is the index for the i^{th} instant time, i.e, $t_i = i \cdot \Delta t$ for $i = 0, 1, 2, ...,$N, λ_i^k represents the solution at time t_i at the k-th iteration of the parareal cycle. The initial value of λ_i^k is known and is denoted by λ_0^0. The parareal algorithm may then be described by the following steps.

- Iteration $k = 0$:

  P_0 uses G in serial, to calculate initial values λ_i^0 for each processor dealing with separate time chunks. As G is a coarse solver, these initial values are only an estimation of the actual values.

- Iteration $k > 0$:

  Step 1: Each processor (i.e., P_j) then separately applies **F** to propagate the solution, starting with the initial values provided by **G** (i.e., λ_j^{k-1}), between the initial (t_j) and final time (t_{j+1}) of its respective time chunk. This process is in parallel. The result of this propagation is forwarded to the next processor in line (P_{j+1}).

  Step 2: **G** is now applied as a sequential (but not continuous) process, using the parareal prescription to update the initial value at each time chunk:

$$\lambda_{i+1}^{k+1} = \mathbf{G}_{\Delta T}(\lambda_i^{k+1}) + \mathbf{F}_{\Delta T}(\lambda_i^k) - \mathbf{G}_{\Delta T}(\lambda_i^k), \tag{4.2}$$

  Note that this part of the algorithm cannot be done in parallel, because of the first term on the right hand side. Note also that the second and third terms have already been obtained in previous steps and/or iterations.

Step 3: Check for convergence. The cycle is exited if the solution is converged for all chunks. Otherwise, another iteration of the parareal cycle follows, starting from the chunk previous to which all chunks have converged. This is a slight modification to the standard implementation of the parareal scheme, and is discussed in detail in [2].

Convergence is said to have occured when the average relative error between two successive iterations, k and $k-1$ is less than the tolerance. The average relative error, σ_i^k, for the chunk evolving λ_{i-1}^k at t_{i-1} to λ_i^k at t_i is given by

$$\sigma_i^k = \int_{t_{i-1}}^{t_i} \frac{|\lambda^k(s) - \lambda^{k-1}(s)|}{|\lambda^{k-1}(s)|} ds. \tag{4.3}$$

4.3 Application of the technique to models with different nonlinearities

The parareal algorithm was successfully applied to a turbulent system in [2]. The DTEM model used in [2] consisted of the $\mathrm{E} \times \mathbf{B}$ nonlinear term. This chapter explores the performance of the algorithm for turbulent systems with other nonlinearities.

4.3.1 Models

The turbulence simulations discussed in this paper involve DTEM models. These plasma drift wave models have been extensively studied in [16], [17] and [18]. The models assume a shearless slab geometry in the $x-y$ direction, with a uniform magnetic field along the $\hat{z}$ direction. $\hat{x}$ actually represents the radial coordinate, $\hat{y}$ the poloidal coordinate, and $\hat{z}$ the toroidal coordinate of an actual three dimensional system. In this reduced model, profile gradients exist along the $\hat{x}$ direction resulting in the propagation of drift waves along the $\hat{y}$ direction. The ions are treated as a cool fluid and the instability drive is caused by the strongly dissipative trapped electron mode. The model where the parareal scheme is first applied in this chapter has the polarization drift nonlinearity and is given by

$$\frac{\partial}{\partial t}(1 - \rho_s^2 \nabla_\perp^2)\tilde{\phi} + D\frac{\partial^2 \tilde{\phi}}{\partial y^2} + \frac{V_D}{2}\frac{\partial \tilde{\phi}}{\partial y} - \rho_s^3 C_s \nabla\tilde{\phi} \times \hat{\mathbf{z}} \cdot \nabla\nabla_\perp^2 \tilde{\phi} = 0. \tag{4.4}$$

where $\tilde{\phi}$ is the fluctuating potential, $V_D = \varepsilon^{1/2}(cT_i/eB)L_n^{-1}$ is the effective diamagnetic drift velocity and T_i is the ion temperature. B represents the magnetic field strength. $D = V_D^2/4\nu_{\text{effe}}$, with ν_{effe} being the effective collision frequency of ion-electron collisions. The second term on the left hand side of Eq. (4.4) represents the instability drive and the third one is the wave term which causes the drift waves to propagate along the y direction. The last term on the left hand side of Eq. (4.4) is the polarization drift nonlinearity. This nonlinear term actually represents the advection of the vorticity in the turbulent field.

The original turbulence model, described in [16] and [17], where the parareal algorithm was first applied [2], was very similar to Eq. (4.4), except for the fact that the fourth term on the left hand side was replaced by the $\mathbf{E} \times \mathbf{B}$ nonlinearity, given by $\frac{4L_nD}{\varepsilon^{1/2}}\left[\nabla_\perp\left(\frac{\partial\tilde{\phi}}{\partial y}\right)\times\hat{\mathbf{z}}\right]\cdot\nabla_\perp\tilde{\phi}$. $\varepsilon^{-1/2}$ is the fraction of trapped electrons and L_n is the density gradient scale length. The advection of the potential (or the density) is represented by the $\mathbf{E} \times \mathbf{B}$ nonlinearity.

Eq. (4.4) is very similar to the Hasegawa-Mima equation, while the polarization drift nonlinearity is very similar to the nonlinear term in the two Dimensional Navier Stokes equation for incompressible fluids. This model is also similar to the quasi-geostrophic equation commonly used to study planetary flows. The polarization drift nonlinearity is dominant at short wavelengths, while the opposite is true for the $\mathbf{E} \times \mathbf{B}$ nonlinearity. An example of the vorticity field of this model is illustrated in Fig. 4.1. The eddies in this case (unlike the $\mathbf{E} \times \mathbf{B}$ [2], [16], [17]) are isotropic since the nonlinear term in k-space, for this model, given by $\frac{\rho_s^3 C_s}{2}\sum_{\mathbf{k}'} k^2(\mathbf{k}\times\mathbf{k}'\cdot\hat{\mathbf{z}})\tilde{\phi}_{\mathbf{k}}\tilde{\phi}_{\mathbf{k}-\mathbf{k}'}$ has no exclusive dependence on k_x or k_y. The different colors in the plot represent different spinning orientations of the vortices.

This model has an inverse cascade of energy, i.e, the energy travels from small scales to large scales - a characteristic of the Hasegawa-Mima equation, while enstrophy cascades to small scales. This is in contrast to the $\mathbf{E} \times \mathbf{B}$ nonlinearity which has only one quadriatic invariant, energy, which cascades to small scales.

The fact that the $\mathbf{E} \times \mathbf{B}$ and the polarization nonlinearity have contrasting effects on the direction of the energy cascade makes study of a model containing both the nonlinearities interesting. The parareal algorithm is thus applied to such a turbulent system. The model is given by

$$\frac{\partial}{\partial t}(1-\rho_s^2\nabla_\perp^2)\tilde{\phi}+D\frac{\partial^2\tilde{\phi}}{\partial y^2}+\frac{V_D}{2}\frac{\partial\tilde{\phi}}{\partial y}+\frac{4L_nD}{\varepsilon^{1/2}}\left[\nabla_\perp\left(\frac{\partial\tilde{\phi}}{\partial y}\right)\times\hat{\mathbf{z}}\right]\cdot\nabla_\perp\tilde{\phi}-\rho_s^3C_s\nabla\tilde{\phi}\times\hat{\mathbf{z}}\cdot\nabla\nabla_\perp^2\tilde{\phi}=\mathbf{0}. \tag{4.5}$$

Fig. 4.2 illustrates the vorticity field for the model with both nonlinearities. The different colors represent the magnitude and orientations of the spins. The lack of isotrophy in the eddies is introduced by the presence of the $\mathbf{E}\times\mathbf{B}$ nonlinear term, which has an additional k_y dependence.

The simulations were done using the pseudo spectral code BETA [16]. The VODPK package [19] is employed to solve the stiff problems, using a preconditioned, Krylov implicit solver based on backward differentiation formulas of variable order and timestep. The timesteps were tuned accordingly within the solver so that the error was always below a prescribed tolerance. In the Fourier space or k-space, the $\mathbf{E}\times\mathbf{B}$ nonlinearity takes the form $\frac{4iL_nD}{\varepsilon^{1/2}(1+\rho_s^2k_\perp^2)}\sum_{\mathbf{k}'}k'_y(\mathbf{k}\times\mathbf{k}'\cdot\hat{\mathbf{z}})\tilde{\phi}_{\mathbf{k}}\tilde{\phi}_{\mathbf{k}-\mathbf{k}'}$ and the polarization drift nonlinearity is represented as $\frac{\rho_s^3C_s}{2}\sum_{\mathbf{k}'}k^2(\mathbf{k}\times\mathbf{k}'\cdot\hat{\mathbf{z}})\tilde{\phi}_{\mathbf{k}}\tilde{\phi}_{\mathbf{k}-\mathbf{k}'}$. The original grid size for the fine solver comprised of 385×385 modes, with $kmax=192$ representing the maximum mode number on each side of the (0,0) mode.

4.3.2 Results for model with polarization drift nonlinearity

Success of the parareal algorithm with this model was achieved by a reduction of the grid size in k-space for the coarse solver, as described in Section 4.5.3. The technique of reducing the grid size was successful only when the high k modes in Fourier space that were removed from the calculations in G, contained very little energy compared to the total energy of the system. This effect was achieved in our simulations in two distinct ways. One was by introducing a high damping coefficient for the mid k region, and another was by adding a large damping in the high k region. The original simulations with the $\mathbf{E}\times\mathbf{B}$ nonlinearity in [2] already had the high k modes damped, due to which they contributed very little energy to the entire system.

Apart from reducing the grid size in G, the VODPK solver was replaced by 2nd and 4th order Runge-Kutta solvers, which significantly reduced the computation time. Convergence was observed with this choice for the coarse solver for both cases, and as an example, Fig. 4.3 shows the convergence of the total energy with increasing iterations for a case where the mid k region is highly damped. Fig. 4.3 shows a simulation with 960 processors and a total simulation length of 384000 steps. In this figure, each color reresents the plot of the total energy of the system for a given parareal iteration. It may be observed that the black and the red lines, corresponding to the 2^{nd} and 3^{rd} parareal iterations, respectively, deviate largely from the solution from the final iterations (cyan for the 7^{th} and pink for the 8^{th} iterations). However, for a short simulation length (10^5), the solutions converge even in the 2^{nd} and 3^{rd} iterations. With increasing iterations, the solutions converge towards a final value along the entire time series, until the difference between the values in the total energy from two successive iterations is less than 1.5×10^{-6}.

For the case where the mid k region is highly damped, a typical serial run with the same parameters would take approximately 38.176 hours for 384000 timesteps. The parareal run, with this technique, generated a gain of 4.5 with 80 processors. The gain increases with increasing processors (to about gain = 9.12), but is accompanied by worse efficiency. Results were similar for the case with a hyperviscosity in the high k modes, where a typical serial run approximately required 5.445 hours for 24000 steps. A parareal run for these parameters, with 80 processors generated a gain of 6.25, while the use of 320 processors yielded a gain of 10.66. The bad efficiency rises largely from the fact that processors, for which convergence has already occured, and which correspond to earlier parts of the long time series, remain idle through the rest of the parareal cycles. We believe that the efficiency would greatly improve if all the processors could be used uniformly through all the parareal iterations, which may be carried out as a future endeavor.

4.3.3 Results for model with two nonlinearities

The parareal algorithm could be successfully applied to this model. The coarse solver used for this implementation was very similar to the one already discussed in Sections 4.3.2 and 4.5.3. Once again, applying a relatively large damping in the mid k modes allowed the high k modes to have lesser energy compared to that of the entire system.

Fig. 4.4 illustrates the convergence of the parareal solution with increasing iterations for a simulation with 320 processors for 64000 tiemsteps. Each iteration (k) is represented by a different color in the plot. The total energy of the system, plotted against time, t, attains a steady value after 11 iterations (k = 11).

A typical serial run for this simulation took 17.52 hours for 64000 steps. Gains of 3.17 for 80 processors and 9.87 for 320 processors could be achieved using the parareal technique. The efficiency can be improved by the same technique discussed in sections 4.3.3.

4.4 Convergence analysis

Once convergence is confirmed to be attainable in a turbulent system by application of the parareal algorithm, a systematic study of the convergence properties becomes necessary. In order to study the convergence rate of the parareal algorithm for turbulence simulations, a model for estimations of computational gain was introduced in [2]. The model calculated the computational gain for strong scaling studies, and work per processor for weak scaling studies. The model thus allowed the use of the correct number of processors with the ideal chunk size, to achieve optimum gain and efficiency for a given simulation. This model resulted from the fact that k, the number of iterations required for convergence, depended on the total simulation length, T, as well as the number of processors, N. For strong scaling studies, k_s obeyed the following relation:

$$k_s(N) = \begin{cases} N, & T < t_1 \\ \frac{t_1}{T}N + \Sigma(t_1, t_2, b, T, N), & t_1 < T < t_2 \\ \frac{T}{b} + \frac{\frac{bN}{T} - 1}{\frac{bN}{T} + 1}\left(\frac{t_2}{b} + \frac{t_1}{T}N\right), & T \geq t_2 \end{cases} \tag{4.6}$$

where the function for the transition range of $T's$ is given by:

$$\Sigma(t_1, t_2, b, T, N) \equiv \frac{2(t_2 - t_1)\frac{N}{T}}{\left(\frac{bN}{T}\right)^2 - 1}\left\{\sqrt{1 + \left(\left(\frac{bN}{T}\right)^2 - 1\right)\frac{(T - t_1)}{t_2 - t_1}} - 1\right\}. \tag{4.7}$$

Here, t_1 is a simulation time until which the convergence rate is 1, or, one processor is converged per iteration. It was observed, that beyond t_1 and until t_2, the convergence rate linearly increased with the simulation time. Beyond t_2 the convergence rate was b 1.

Based on the expression for k_s, the following expression for the gain, H_{PA}, was also derived in [2].

$$H_{PA}^{T \gg t_2}(N) \sim \begin{cases} \left(\frac{b}{T}\right) N, & N \ll \beta \\ \frac{b\beta}{T}, & \beta \ll N \ll N_s \\ \left(\frac{\beta T}{t_1}\right) \frac{1}{N}, & N \gg N_s \end{cases} \tag{4.8}$$

By a similar derivation for the weak scaling, the work per processor had the following expression in [2]:

$$W_{PA}^{T \gg t_2}(N) \sim \begin{cases} \frac{t_1}{\Delta T}, & \frac{t_1}{\Delta T} \ll N \ll N_w \\ \left(\frac{\Delta T}{b}\right) N, & N_w \ll N \ll \beta \\ \left(\frac{\Delta T}{b\beta}\right) N^2, & N \gg \beta \end{cases} \tag{4.9}$$

Here, ΔT represents the chunk size per processor.

A reasonable agreement of the model with actual simulation results was observed in [2]. However, the origins of t_1, t_2, b and m were not explored. A study of these parameters based on variations of the coarse solver is presented in Section 4.4.1.

4.4.1 Convergence for varying Gs

A number of simulations with a variety of coarse solvers were run for the model with the $\mathbf{E} \times \mathbf{B}$ nonlinearity, in order to study the interdependence of the four parameters t_1, t_2, b and m, as well as their dependence on the system. This was done in order to make the formulas have predictive values when optimizing. Three kinds of solvers were used for G, namely Runge Kutta 2^{nd} order, Runge Kutta 4^{th}

order and VODPK (which, also, is the fine solver). For each of the solvers, two reduced grid sizes with *kmax* = 72 and *kmax* = 100 were explored.

Fig. 4.5 is a sketch of the properties displayed by the convergence rate for a typical parareal run for a turbulent system with the $\mathbf{E} \times \mathbf{B}$ nonlinearity. The convergence rate, ΔN, multiplied by the chunk size, gives the physical (simulation) time that is converged per iteration, and this is plotted along the y-axis. The number of processors that have attained convergence, multiplied by the chunk size, gives the physical/simulation time that has already converged, and it is plotted along the x-axis. The sketch shows a distinct lower flat region, till t_1, where the number of processors converging per iteration is small. There is another flat region on top, beyond t_2, where the convergence rate is higher than 1 chunk size. The path from t_1 to t_2 may be said to follow a straight line of slope m.

Figs. 4.6 and 4.7 are actual simulated results for the convergence rate *vs.* simulation time. Fig. 4.6 illustrates the results in a lin-lin plot. Fig. 4.7 is a plot of the same simulation data, but in log-log, to highlight the flatness of the different regimes. pe_{done} is the number of processors already converged. The different colors on the plots correspond to different choices for the coarse solver, G. In these plots, RK stands for Runge Kutta. kmax_pr is the maximum number of modes on either side of the $(0,0)$ mode in Fourier space, for the calculations by the coarse solver. dt_pr is the timestep used in G, and dt, as shown in the plot, is the time step for the fine solver, F. Also, in these plots, n stands for the number of processors involved for a given simulation, and c is the corresponding chunk size per processor for that simulation.

It must be noted here that the lower flat portion is not available for the simulations with *kmax* = 100 for G in Fig. 4.6 and Fig. 4.7. This is due to numerical restrictions in the smallness of a given chunk size. The chunk size needs to be significantly small (and, of course, smaller than t_1), for the lower flat region to appear on the plot. This case is further highlighted by Fig. 4.8. Fig. 4.8 may be considered as very similar to Fig. 4.6, with only the chunk sizes being big enough so that only the upper flat portion of the convergence rate is obtained. It is likely that t_1 is much smaller when *kmax* = 100 is used for the coarse solver.

What is interesting about Fig. 4.6 and Fig. 4.7, is that, for the same reduced grid size (72 or 100) in G, the plots overlie on each other, irrespective of the solver - be it, Runge Kutta 2nd order, 4th order or VODPK. However, this observation is clearer for the case *kmax* = 72, where there are lesser fluctuations in the value of b than for *kmax* = 100. Table 4.1 lists the results obtained from using different solvers for G, all having the same reduced grid size, *kmax* = 72, for the coarse solver. As this table suggests, $b \sim 1200$ for different coarse solvers using *kmax* = 72.

The grid size plays an important role in the dynamics of the system and largely determines how different the solution of the coarse solver is, from that of the fine solver. This implies that the variables t_1, t_2, b and m depend more strongly on the physics of the coarse and fine solvers. The difference in the performance with respect to the computational gain among the three solvers is then primarily caused by β, as given in Eq. (4.8) and Eq. (4.9). $\beta = \frac{T_F^{ser}(T)}{T_G^{ser}(T)}$, where T_F^{ser} is the serial time using the fine solver F, and T_G^{ser} is serial time using the coarse solver G. Typically, for the cases considered, for Runge Kutta 4th order, $\beta = 210.3$, for Runge Kutta 2nd order, $\beta = 84.3$ and for VODPK, $\beta = 16.3$.

A similar study with the polarization drift nonlinearity would be very beneficial as it would allow us to see if the convergence rate illustrated in Fig. 4.5 for the $\mathbf{E} \times \mathbf{B}$ nonlinearity also retains the same shape for the polarization drift nonlinearity. However, the fact that t_1 is possibly very small for this case gives rise to figures similar to Fig. 4.8. The numerical restriction in reducing the chunk size by an excessive amount poses a problem in generating these plots for analysis.

4.5 Coarse solvers : a discussion

A variety of options were explored for the coarse solver, G, for the application of the parareal algorithm to the turbulence model described in [2]. Finding the optimum choice of G is a laborious task for a turbulent system. We explored various techniques, which we describe in the following subsections. Some of the methods for G were successful in terms of convergence of the parareal solutions, but did not help in achieving a significant speedup. The cases presented in this section refer to the model with the $\mathbf{E} \times \mathbf{B}$ nonlinearity, but similar analysis prevails for the polarization drift nonlinearity as well.

4.5.1 Coarser timesteps for G

Keeping the resolution in k-space the same in both solvers, F & G, a larger timestep in G, dt_G, was used in place of dt_F, the timestep in F. Employing this technique for the coarse solver has given significant speedup in other systems. For our case, convergence was observed till $dt_G = 10dt_F$. The solutions diverged if dt_G was increased more than that. The plot of the total energy in k-space as a function of time, at different iterations, is shown in Fig. 4.9. In this plot, k represents the parareal iteration. It can be seen that the solutions corresponding to iterations greater that $k = 2$ almost lie on one another, highlighting the convergence of the system. The converged solution using the parareal algorithm in this case also matches with the serial solution using the fine solver, denoted as "serial" in Fig. 4.9. The solution using the coarse solver, denoted as "coarse", is also shown on Fig. 4.9 as one deviating from the fine solver solution, with increasing time.

However, convergence, when observed, was attained after about five iterations with 20 processors. However, the wallclock time for such a computaion was greater than that for a serial run with the fine solver. Thus, although this technique for G is clearly a failure in terms of the speedup, it highlights two significant points. First, convergence is attainable despite the sensitivity of the solutions to initial values. Second, this method of using coarse timesteps may be combined with other techniques for G to attain speedup using the parareal algorithm.

4.5.2 G with reduced grid in k-space

Since the calculations were performed with a pseudo spectral code, the resolution of the grid in k-space provided some space for improvement. The grid spacing was kept constant, but the total grid size was reduced to pursue such a method. Fig. 4.10 illustrates this procedure. This technique also resulted in a reduction in the total number of modes used for calculation in G or the coarse solver, thus demanding a lesser computation time.

On transfer from reduced grid size to original grid size it was necessary to give the spectrum a shape similar to that obtained from a serial run. So, the missing points were filled with a function such that the energy of these points decreased with increasing k.

This technique resulted in solutions that were convergent, and the computation time for G was significantly lesser than that for the fine solver. Keeping in mind the Courant Friedrichs Lewy condition, larger time steps, dt_G was combined with a reduced k-space to optimize G as a coarse solver for the parareal algorithm.

4.5.3 G with coarse grid in k-space

This technique was similar to the previous one in the sense that it also involved transforming the grid space for calculations with G. Fig. 4.11 illustrates such a procedure, where modes were skipped in k-space during the calculation with the coarse solver, G, thus increasing the grid spacing but keeping the total grid size constant. The *missing points* on the grid were filled in with a complex spline. The outcome of this technique is illustrated in Fig. 4.12 and Fig. 4.13. The figures represent the spectrum in k space or Fourier space. The energy is plotted along the z axis. The points on the x−y plane reprsent the modes. The maximum energy is concentrated in the low k modes.

However, as is evident from Fig. 4.12 and Fig. 4.13, the maximum energy in the former is much greater than than that in the latter. Skipping points in k-space for G also significantly reduced the total energy of the system although the shape of the spectrum remained intact. This was due to the fact that there were a few high energy modes in k-space, and when they were missing in the coarse grid, splining averaged the surrounding low energy modes to fill those points, resulting in modes with much lower energy. Hence, this technique led the coarse solver, G, to generate a solution which was very different (with much less energy in the total system) from the desired solution from the fine solver, F. This caused a violation of the conditions for the choice of G [1], [3], which is required for the stability and convergence of the algorithm. The parareal scheme sets limits on the inaccuracy of the solution of coarse solver [1], [3].

Although this particular choice of the coarse solver was ineffective in our case, it could be useful in systems where the energy is not concentrated on a few selective modes, but is more uniformly distributed. Complex splining can then be used to fill the *missing points* of the grid space and still converge with the parareal algorithm.

4.5.4 G with a different solver

Another choice for the coarse solver could be to use a completely different propagator for the equation than that is used for the fine solver. The Gear type solver avaiable with the VODPK package [19], which was used for our fine solver runs, was replaced by the Runge Kutta solvers of both 2^{nd} and 4^{th} orders. The Runge Kutta solvers proved to be a good choice as they were much faster than the the fine solver, but at the same time, allowed convergence for the parareal scheme.

The Runge Kutta solvers were implemented by combining the techniques described in Sections 4.5.1 and 4.5.3. This gave an even faster coarse solver, thus contributing to overall computational gain.

4.5.5 G with reduced physics

If the coarse solver employs a reduced physics model as compared to the fine solver, computation time may be expected to decrease. However, for the turbulent system discussed here, simplification of the model in G generated diverging parareal solutions. If V_D or the diamagnetic drift velocity in Eq. (4.4) was neglected in the coarse solver, the computation time was reduced, but the method was not fast enough to generate computational gain. The fact that the parareal scheme attains convergence after a number of iterations involving successive use of F (in parallel) and G (in serial), the computation time for G has to be significantly lesser than F in order to observe computational gain. Moreover, although initial convergence was observed over a very short simulation time, solutions diverged in case of longer runs.

4.6 Conclusion

The parareal algorithm can be successfully applied to more than one simulation of fully developed turbulence. The drift wave model for these simulations include the polarization drift nonlinearity and a combination of both the polarization and the $\mathbf{E} \times \mathbf{B}$ nonlinearities. However, having reduced energy in the high k modes as compared to the rest of the system allows convergence in cases where a reduced grid size in k-space is used for the coarse solver, G. Calculations with a reduced grid size in G gives a better computational gain.

The other conclusion is that the convergence of these simulations do not entirely depend on the numerics, but on the physics of the coarse solver as well. So, an optimum choice of G is based on the dynamics of the system, along with computational speed. However, as also observed in [2], a good G is not necessarily a fast one, but one for which the convergence rate is reasonably high as well.

This work expands the prospect for the applicability of the parareal algorithm to other turbulent systems. This fact serves as a motivation for work on the improvement of the efficiency of the parareal algorithm. For example, reuse of processors that currently remain idle after attainment of convergence may greatly increase the efficiency.

Table 4.1. The value of b is nearly constant for different coarse solvers used for the same reduced grid size in G.

G	$k_{\max}$	dt_F/dt_G	n	c	b
RK2	72	8	44	160	1200 ± 160
RK2	72	8	88	80	1200 ± 80
RK2	72	8	200	80	1067 ± 80
RK2	72	8	320	80	1227 ± 80
RK2	72	8	640	16	1139 ± 16
RK4	72	40	88	80	1067 ± 80
RK4	72	40	200	80	1062 ± 80
VODPK	72	4	200	80	1200 ± 80
VODPK	72	4	280	80	1415 ± 80

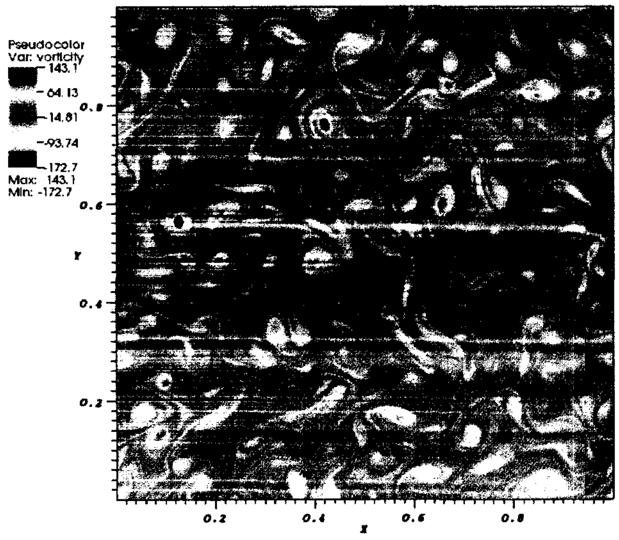


Figure 4.1. Eddies are isotropic for the model with only the polarization drift nonlinearity.

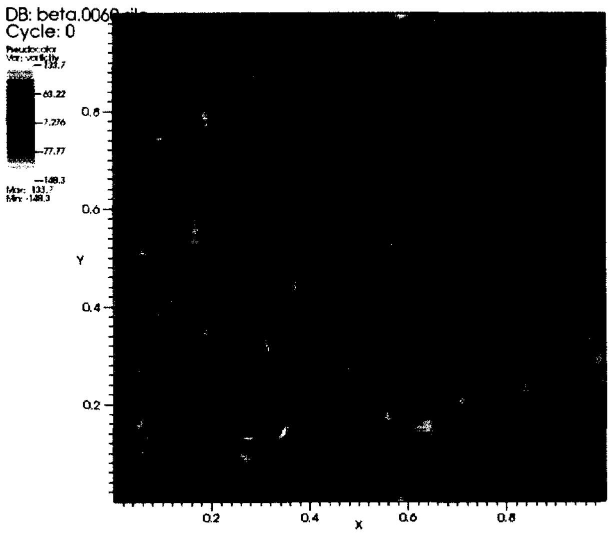


Figure 4.2. Vorticity field for the model with both the polarization drift nonlinearity and $\mathbf{E} \times \mathbf{B}$ nonlinearity.

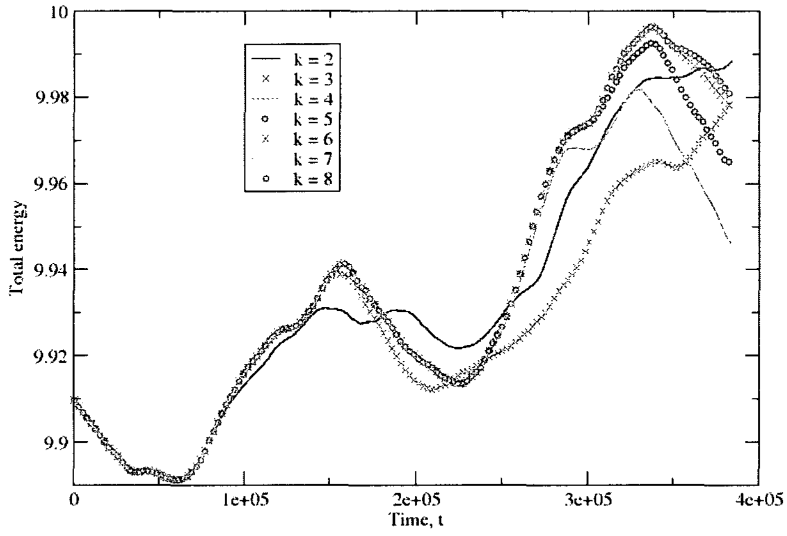


Figure 4.3. Convergence of the total energy of the system for the case with polarization drift nonlinearity. This is a simulation of 384000 timesteps with 960 processors.

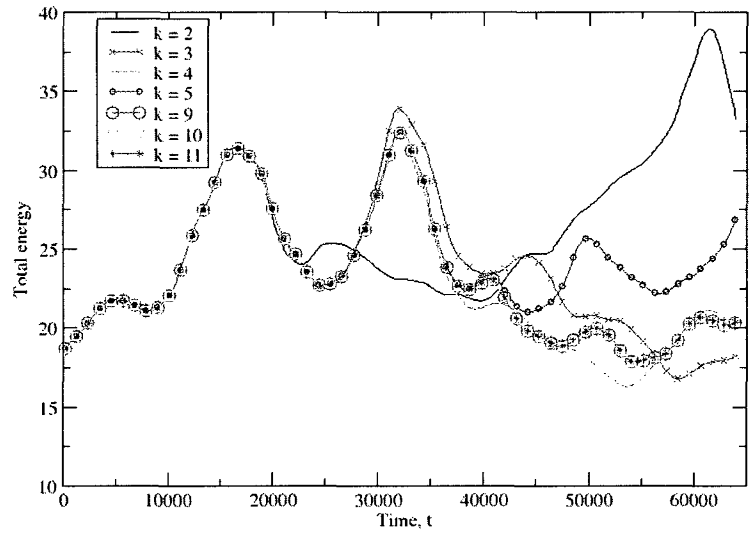


Figure 4.4. Convergence of the total energy of the system for the case with both polarization drift nonlinearity and $\mathbf{E} \times \mathbf{B}$ nonlinearity. This is a simulation of 64000 timesteps with 320 processors.

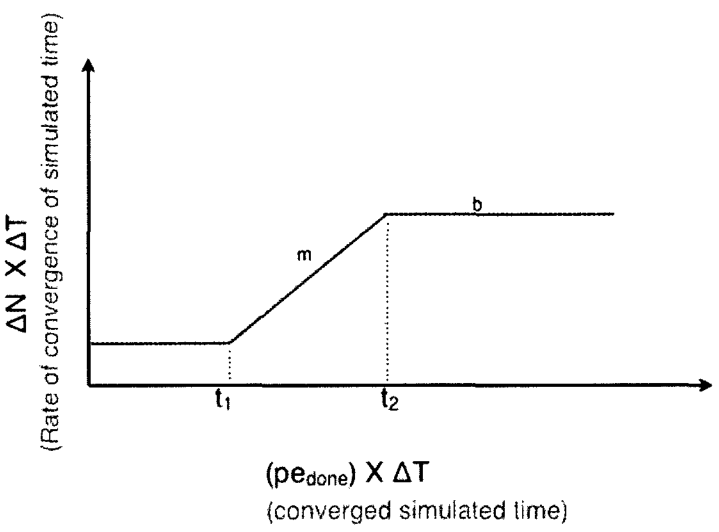


Figure 4.5. Sketch of the typical features of the rate of convergence shows that the rate of convergence is low till simulation time t_1 is converged, and then gradually attains a value, b, after time t_2 is converged.

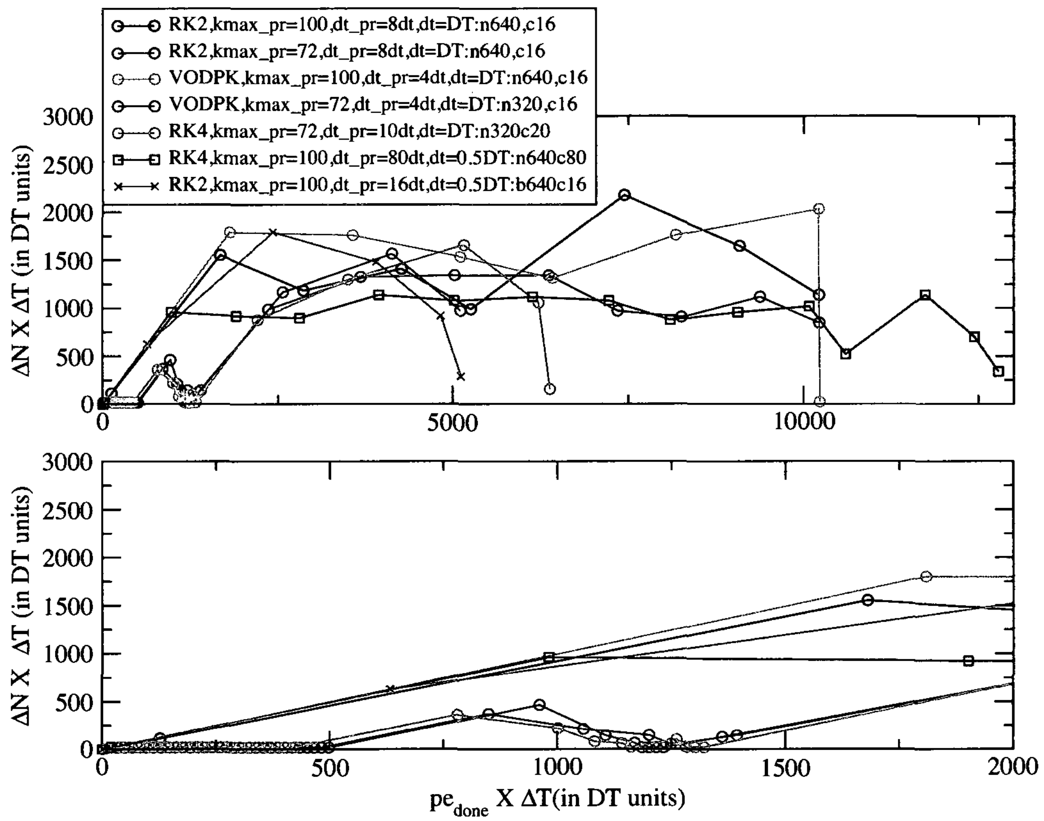


Figure 4.6. The convergence rate is similar if the reduced grid size for the coarse solver, G, is the same, irrespective of the solver for the differential equation.

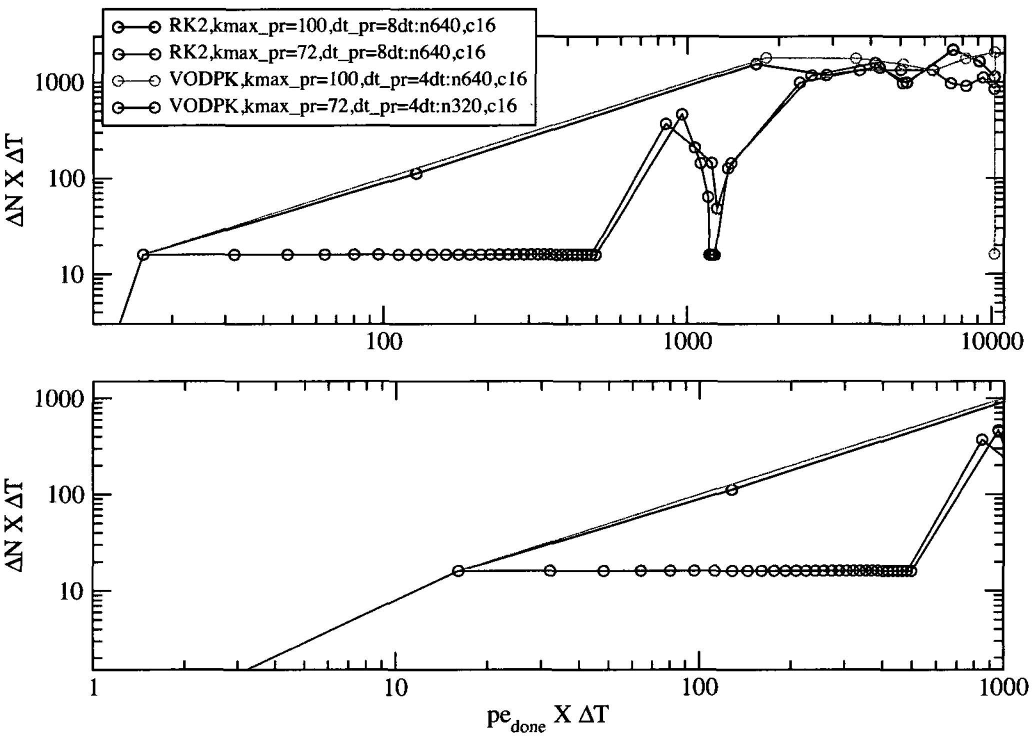


Figure 4.7. Plot similar to Fig. 4.6, but in log-log, to highlight the flatness of the convergence rate at different regions of simulation time.

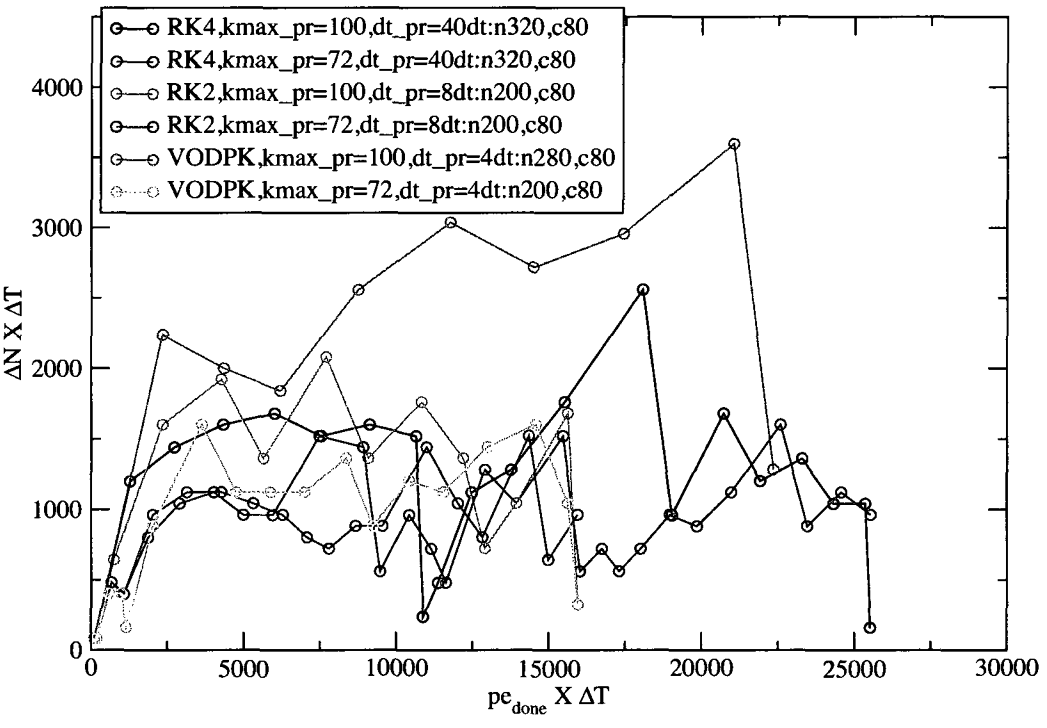


Figure 4.8. For large chunk sizes, the convergence rate for $t < t_1$ is absent, only $t > t_2$ is present.

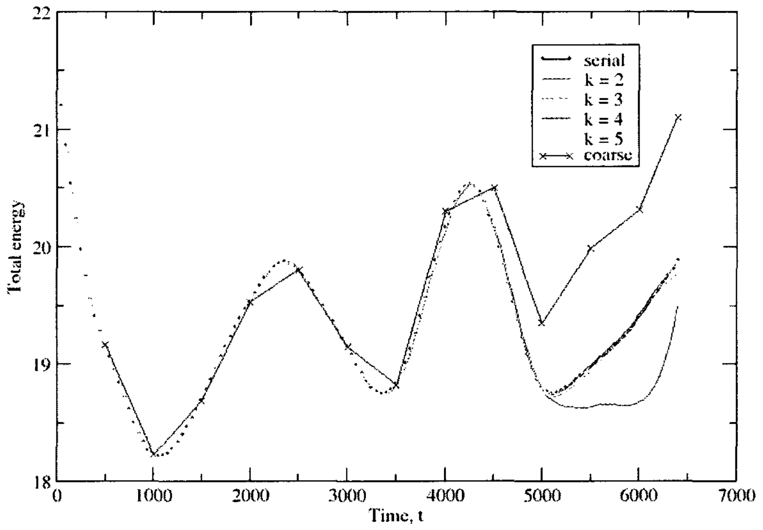


Figure 4.9. Convergence with larger timesteps in G. The total energy in k-space is plotted as a function of time, for different iterations, represented by F.

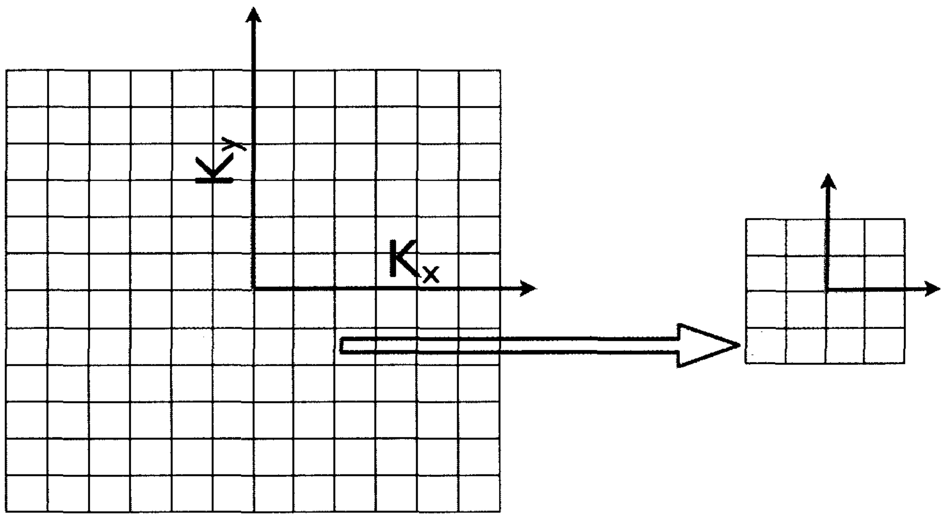


Figure 4.10. A reduced grid in k-space was attempted to use for G to speed up the calculation.

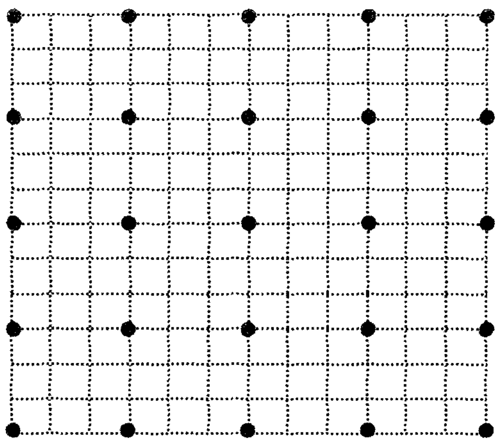


Figure 4.11. A coarse grid in k-space was attempted to use for G to speed up the calculation.

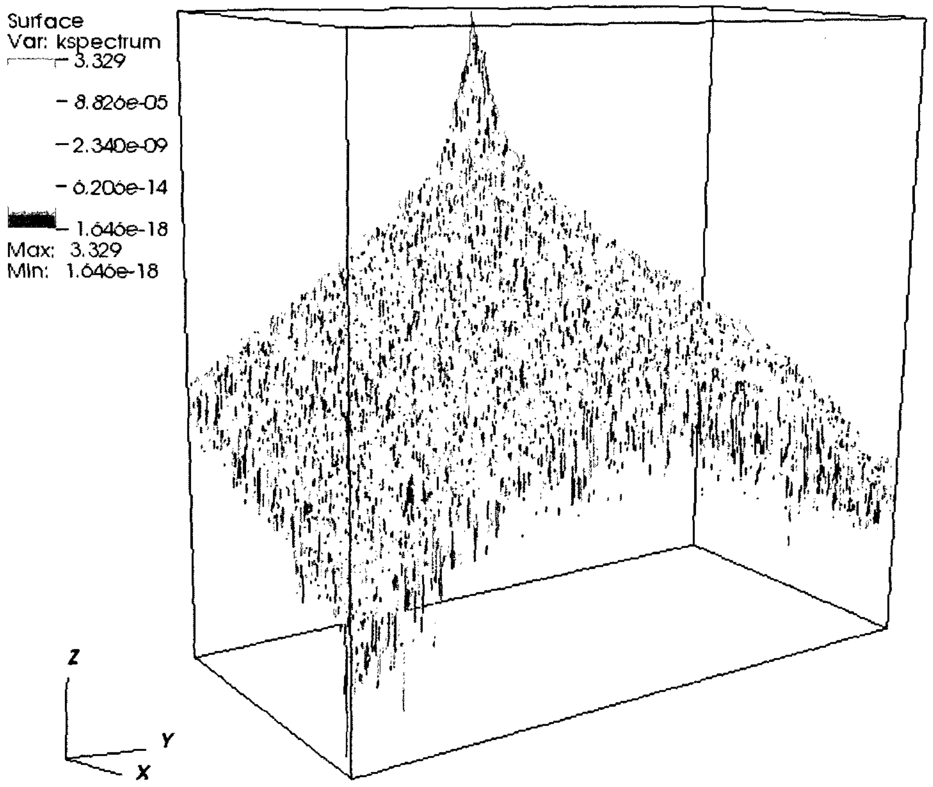


Figure 4.12. The total energy in k-space when calculated on the fine grid.

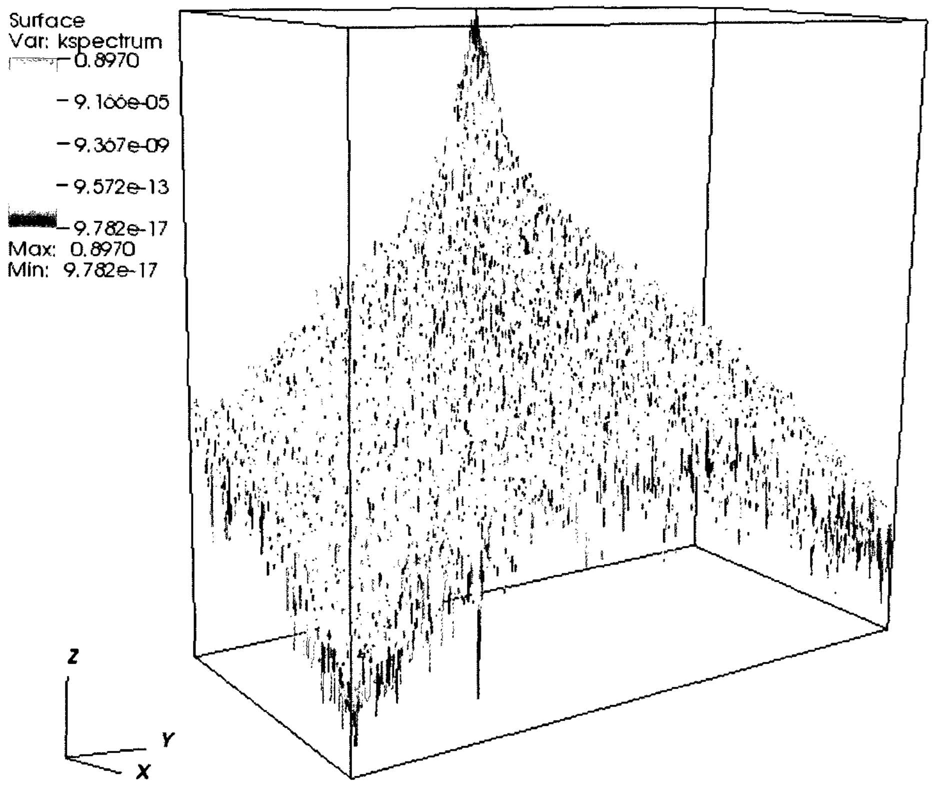


Figure 4.13. The total energy in k-space after the spectrum was transferred from the coarse grid to the fine one, by splining.

Bibliography

[1] J. Lions, Y. Maday, G. Turinici, A "parareal" in time discretization of pde's, Comptes Rendus de l'Academie des Sciences - Series I - Mathematics 332 (7) (2001) 661-668.

[2] D. Samaddar, D. E. Newman, R. Sanchez, Parallelization in time of numerical simulations of fully developed plasma turbulence using the parareal algorithm, J. Comp. Phys. (2009) (submitted).

[3] L. Baffico, S. Bernard, Y. Maday, G. Turinici, G. Zerah, Parallel in time molecular dynamics simulations, Phys. Rev. E 66 (5) (2002) 057706.

[4] I. Garrido, M. S. Espedal, G. E. Fladmark, A convergent algorithm for time parallelization applied to reservoir simulation, in: Proceedings of Fifteen International Conference on Domain Decomposition Methods, Vol. 40, Springer, Berlin, (2004) 469-476.

[5] P. F. Fischer, F. Hecht, Y. Maday, A parareal in time semi-implicit approximation of the navier-stokes equations, in: Proceedings of Fifteen International Conference on Domain Decomposition Methods, Vol. 40, Springer Verlag, (2004), pp. 433-440.

[6] G. Bal, Y. Maday, A parareal time discretization for nonlinear PDEs with application to the pricing of an American put, Vol. 23 of Lect. Notes Comput. Sci. Eng., Springer, Berlin, 2002, pp. 189-202.

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

[8] M. J. Gander and E. Hairer, Nonlinear convergence analysis for the parareal algorithm, in: Domain Decomposition Methods in Science and Engineering XVII, Vol. 60, Springer, (2008), pp. 45-56.

[9] Y. Liu, J. Hu, Modified propagators of parareal in time algorithm and application to Princeton Ocean model, Internat. J. Numer. Methods Fluids 57 (12) (2008) 1793-1804.

[10] G. Staff, G. Rnquist, Stability of the parareal algorithm, in: Proceedings of Fifteen International Conference on Domain Decomposition Methods, Springer Verlag, (2003), pp. 449-456.

[11] G. Bal, On the Convergence and the Stability of the Parareal Algorithm to Solve Partial Differential Equations, Vol. 40 of Lect. Notes Comput. Sci. Eng., Springer, Berlin, (2005), pp. 425-432.

[12] M. Gander and S. Vandewalle, Analysis of the Parareal Time-Parallel Time-Integration Method, Siam J. Sci. Stat. Comput. 29 (2) (2007) 556-578.

[13] P. K. Kundu and I. M. Cohen, Fluid Mechanics, Elsevier Academic Press, (2004).

[14] U. Frisch, Turbulence, The Legacy of A. N. Kolmogorov. Cambridge, U.K.: Cambridge Univ. Press, (1995).

[15] F. Takens, Detecting Strange Attractors in Turbulence, Vol. 898 of Lecture Notes in Mathematics, Springer, Berlin, 1981, p. 366.

[16] D. E. Newman, P. W. Terry, P. H. Diamond, A two-nonlinearity model of dissipative drift wave turbulence, Phys. Fluids B 4 (3) (1992) 599-610.

[17] Y. M. Liang, P. H. Diamond, X. H. Wang, D. E. Newman, P. W. Terry, A two-nonlinearity model of dissipative drift wave turbulence, Phys. Fluids B 5 (4) (1993) 1128-1139.

[18] D. E. Newman, P. W. Terry, P. H. Diamond, Y. M. Liang. The dynamics of spectral transfer in a model of drift wave turbulence with two nonlinearities, Phys. Fluids B 5 (4) (1993) 1140-1153.

[19] A. Hindsmarsh, Gear: Ordinary equation system solver, in: Report UCID-30001, Rev. 3, LLNL, 1974.

Chapter 5
Conclusions

The entire work of this thesis has focused on improving the numerical techniques used to understand turbulence and turbulent transport, with special application to fusion plasmas. Perfectly modeling turbulence is still and will for the foreseeable future be an outstanding problem [1], [2]. Capturing the physics of plasma turbulence is even more challenging, due to the presence of charged fluids and electro magnetic fields [3], combined with the complex dynamics already seen in the turbulence in neutral fluids. This thesis has sought to make a contribution in improving the present computational strategies employed in two types (a 1D model and a 2D model) of turbulence simulations.

5.1 Application of the parareal algorithm to turbulence

The successful implementation of the parareal technique [4] to turbulence modeling serves as the major achievement of this thesis. The value of this very non intuitive technique lies in the fact that it solves a time-dependent initial value problem by time parallelization. A priori success could not be guaranteed, in fact, not apparently expected, due to the very high sensitivity of turbulent systems to initial conditions, combined with exponential divergence of lagrangian trajectories.

Turbulence simulations typically involve a wide range of timescales often spanning more than 10 orders of magnitude. Resolving the micro turbulence of individual eddies makes long time simulations virtually impossible due to the extremely large wall clock times required for these calculations. The only alternate approach is to use effective transport coefficients and assume unchanging or frozen plasma profiles to make long simulations. Modern supercomputers do offer a large number of CPUs (of the order of 100000), but even the state of the art fluid codes for turbulence, using space parallelization, scale only upto a few thousand processors.

Prior to this present research, the parareal algorithm had been applied to many "simple" ("simple" when compared to turbulence) problems [5], [6], [7], [8], [9], [10], [11], [12], [13], [14]. The seminal paper on this subject, by *Lions et al.*, in 2002 [4], is considered a pathbreaking formulation for numerically solving time dependent problems. However, despite its success with problems of much less complexity, the parareal algorithm, to the best of our knowledge, was never successfully applied to fully developed turbulence simulations.

Although in this work, the algorithm has been applied to systems of plasma turbulence [15], [16], [17], the models involved are very similar to a much wider range of problems. The parareal scheme has been shown to work for a dissipative trapped electron mode (DTEM) model for drift wave turbulence, involving the E $\times$ B nonlinearity [18] or the polarization nonlinearity or both. The model with the polarization nonlinearity is similar to the Hasegawa-Wakatani model [19], [20], also described in [16]. The polarization nonlinearity is similar to the nonlinear term in the two dimensional Navier Stokes equation for incompressible fluids. The model is also analogous to the quasi-geostrophic equation used to describe planetary flows. The E $\times$ B nonlinearity results in a direct cascade of energy, while the reverse is true for the polarization nonlinearity.

The parareal algorithm involves two propagators, namely, a coarse solver and a fine solver. The coarse solver employed for the algorithm used a reduced grid in Fourier space, implying calculations without the highest modes in k-space, which are otherwise present in the actual simulation. Combined with the reduced grid, the coarse solver also used a coarser timestep for faster computation with the Runge Kutta solver. The Runge Kutta solver was a replacement for the more accurate, but computationally intensive, propagator used by the fine solver. The parareal algorithm not only yielded converged solutions with this case, but also led to a computational gain close to a magnitude, which is very promising for a system as complex as this.

This thesis presents a detailed account of various choices for the coarse solver that may be employed for the parareal algorithm. Some of these options have not been effective for the turbulent system explored in this work, but those choices may prove to be useful in some other cases. The potential for success for various solvers, depending on the physics as well as the numerics of a particular problem are discussed.

This work also introduces a phenomenological model for the convergence rate of the parareal solution in case of the turbulent system involved. The purpose of this model is to attain predictability about the different parameters related to the parareal simulations and the corresponding convergence to achieve optimum performance. The plots for the gain in case of strong scaling and for the work done per processor in case of weak scaling match reasonably well with the theoretical predictions made by the model. This research also delves more into the details of the characteristics of convergence of the parareal solutions, and further explores the model for the convergence rate for the parareal scheme. A systematic study is made for various options that may be used as the coarse solver for the parareal algorithm, to estimate the roles and dependencies of the different parameters involved in the convergence.

5.2 Noise in transport model

As another contribution to the improvement of turbulence simulations, a 1D transport model was addressed. Specifically, the effect of noise in such a transport model has been studied as the first part of this thesis. Prior to the start of this research, the oscillatory behavior of the density fluctuation in a 1D envelope model was extensively studied in [21], [22]. The simulations of fusion plasma, based on this model, already showed the transition to a high confinement region due to E $\times$ B shear flow suppression of turbulence. In [22], it was also observed that a regime existed where the damping of turbulence due to shear flow was comparable to the growth rate. The onset of the oscillations was also found to be in the latter region.

In the previous work just discussed [21], [22], the simulations made reasonable agreement with experimental results at the Tokamak Fusion Test Reactor [23]. However, the propagation length of the observed waves was unphysically large. The effects of noise as a surrogate for turbulent decorrelation on the propagation distance of the waves were explored. It became apparent that the addition of noise to the system led to both enhancement as well as reduction of the correlation of the waves, depending on the amount of noise applied. This result, though very interesting, and possibly seen for the first time in an envelope model of turbulent plasma, agrees well with the numerical observations made in other reaction diffusion systems [24], [25], [26]. In our simulations, decorrelation particularly occurred when a high amplitude of perturbation was applied to a large number of sites or points in space. This result implies that noise may be used as a surrogate for turbulent decorrelation in real turbulence models.

5.3 Future work

The research conducted as part of this thesis ushers in a lot of opportunities in the realm of computational study of turbulence. The application of the parareal algorithm discussed here involves three models of fusion plasma. However, this can and should be extended to other turbulence models as well.

More options for coarse solvers will need to be explored, particularly for new models, for optimum convergence rate and speedup.

Time parallelization has only been explored here. However, combining this technique with space parallelization is likely to improve the computational gain. Moreover, in the current scheme, processors that have attained convergence remain idle for the rest of the parareal cycles, while calculations are performed on other processors. Reuse of these idle processors may greatly contribute to increasing the efficiency. Thus, a combination of space and temporal parallelization, accompanied by processor reuse in the parareal technique may lead to utilization of processors and supercomputing resources that has not yet been achieved.

Bibliography

[1] P. K. Kundu and I. M. Cohen, Fluid Mechanics, Elsevier Academic Press, (2004).

[2] U. Frisch, Turbulence, The Legacy of A. N. Kolmogorov. Cambridge, U.K.: Cambridge Univ. Press, (1995).

[3] B. A. Carreras, Progress in anomalous transport research in toroidal magnetic confinement devices, IEEE Trans. Plasma Sci. 25, (1997) 1281.

[4] J. Lions, Y. Maday, G. Turinici, A "parareal" in time discretization of pde's, Comptes Rendus de l'Academie des Sciences - Series I - Mathematics 332 (7) (2001) 661-668.

[5] L. Baffico, S. Bernard, Y. Maday, G. Turinici, G. Zerah, Parallel in time molecular dynamics simulations, Phys. Rev. E 66 (5) (2002) 057706.

[6] G. Bal, Y. Maday, A parareal time discretization for nonlinear PDEs with application to the pricing of an American put, Vol. 23 of Lect. Notes Comput. Sci. Eng., Springer, Berlin, 2002, pp. 189-202.

[7] G. Staff, G. Rnquist, Stability of the parareal algorithm, in: Proceedings of Fifteen International Conference on Domain Decomposition Methods, Springer Verlag, (2003), pp. 449-456.

[8] G. Bal, On the Convergence and the Stability of the Parareal Algorithm to Solve Partial Differential Equations, Vol. 40 of Lect. Notes Comput. Sci. Eng., Springer, Berlin, (2005), pp. 425-432.

[9] M. Gander and S. Vandewalle, Analysis of the Parareal Time-Parallel Time-Integration Method, Siam J. Sci. Stat. Comput. 29 (2) (2007) 556-578.

[10] I. Garrido, M. S. Espedal, G. E. Fladmark, A convergent algorithm for time parallelization applied to reservoir simulation, in: Proceedings of Fifteen International Conference on Domain Decomposition Methods, Vol. 40, Springer, Berlin, (2004), pp. 469-476.

[11] P. F. Fischer, F. Hecht, Y. Maday, A parareal in time semi-implicit approximation of the navier-stokes equations, in: Proceedings of Fifteen International Conference on Domain Decomposition Methods, Vol. 40, Springer Verlag, (2004), pp. 433-440.

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

[13] M. J. Gander and E. Hairer, Nonlinear convergence analysis for the parareal algorithm, in: Domain Decomposition Methods in Science and Engineering XVII, Vol. 60, Springer, (2008), pp. 45-56.

[14] Y. Liu, J. Hu, Modified propagators of parareal in time algorithm and application to Princeton Ocean model, Internat. J. Numer. Methods Fluids 57 (12) (2008) 1793-1804.

[15] D. E. Newman, P. W. Terry, P. H. Diamond, A two-nonlinearity model of dissipative drift wave turbulence, Phys. Fluids B 4 (3) (1992) 599-610.

[16] Y. M. Liang, P. H. Diamond, X. H. Wang, D. E. Newman, P. W. Terry, A two-nonlinearity model of dissipative drift wave turbulence, Phys. Fluids B 5 (4) (1993) 1128-1139.

[17] D. E. Newman, P. W. Terry, P. H. Diamond, Y. M. Liang. The dynamics of spectral transfer in a model of drift wave turbulence with two nonlinearities, Phys. Fluids B 5 (4) (1993) 1140-1153.

[18] D. Samaddar, D. E. Newman, R. Sanchez, Parallelization in time of numerical simulations of fully developed plasma turbulence using the parareal algorithm, J. Comp. Phys. (2009) (submitted).

[19] A. Hasegawa and K. Mima, Phys. Rev. Lett. 39, (1977) 205.

[20] A. Hasegawa, C. G. Maclennan and Y. Kodama, Phys. Fluids 22, (1979) 2122.

[21] D. E. Newman, B. A. Carreras, D. Lopez Bruna, P. H. Diamond, and V. B. Lebedev, Dynamics and control of internal transport barriers in reversed shear discharges, Phys. Plasmas 5, (1998) 938.

[22] D. L. Bruna, D. E. Newman, B. A. Carreras and P. H. Diamond, Fluctuation level bursts in a model of internal transport barrier formation, Phys. Plasmas 6, (1999) 854.

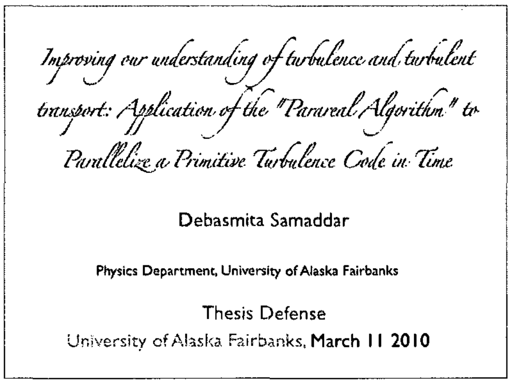
[23] G. Bateman, Phys. Fluids B, 4, (1992) 634.

[24] S. Kadar, J. Wang, and K. Showalter, Nature (London) 391, (1998) 770-772.

[25] J. F. Lindner, S. Chandramouli, A. R. Bulsara, M. Locher, and W. L. Ditto, Phys. Rev. Lett. 81, (1998) 5048-5051.

[26] C. Zhou, J. Kurths, Z. Neufeld, and I. Z. Kiss, Phys. Rev. Lett. 91,(2003) 150601.

Appendix A
Slides from thesis defense



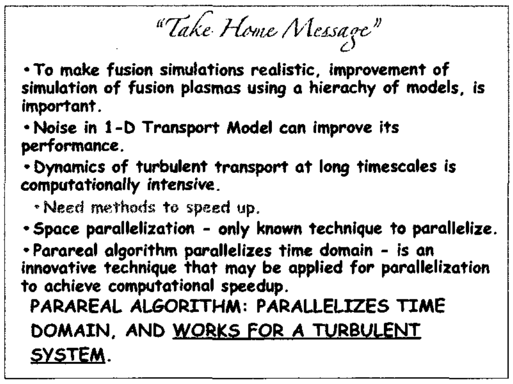
Improving our understanding of turbulence and turbulent transport: Application of the "Parareal Algorithm" to Parallelize a Primitive Turbulence Code in Time

Debasmita Samaddar

Physics Department, University of Alaska Fairbanks

Thesis Defense

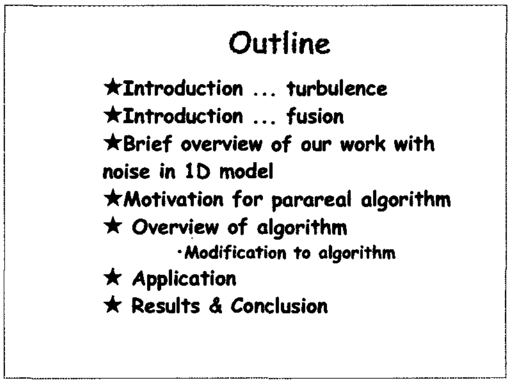
University of Alaska Fairbanks, March 11 2010



"Take Home Message"

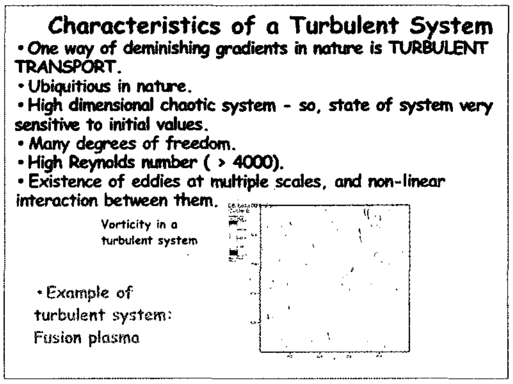
- **To make fusion simulations realistic, improvement of simulation of fusion plasmas using a hierachy of models, is important.**
- **Noise in 1-D Transport Model can improve its performance.**
- **Dynamics of turbulent transport at long timescales is computationally *intensive*.**
  - Need methods to speed up.
- **Space parallelization - only known technique to parallelize.**
- **Parareal algorithm parallelizes time domain - is an innovative technique that may be applied for parallelization to achieve computational speedup.**

PARAREAL ALGORITHM: PARALLELIZES TIME DOMAIN, AND WORKS FOR A TURBULENT SYSTEM.



Outline

★**Introduction ... turbulence**
★**Introduction ... fusion**
★**Brief overview of our work with noise in 1D model**
★**Motivation for parareal algorithm**
★ **Overview of algorithm**
 ·**Modification to algorithm**
★ **Application**
★ **Results & Conclusion**



Characteristics of a Turbulent System

- **One way of deminishing gradients in nature is TURBULENT TRANSPORT.**
- **Ubiquitious in nature.**
- **High dimensional chaotic system - so, state of system very sensitive to initial values.**
- **Many degrees of freedom.**
- **High Reynolds number (> 4000).**
- **Existence of eddies at multiple scales, and non-linear interaction between them.**

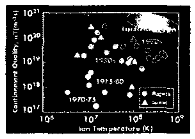
Vorticity in a turbulent system

- Example of turbulent system: Fusion plasma

For Fusion :

- Typically, plasma at 100 million K
- Maximize nTτ (Lawson criterion)

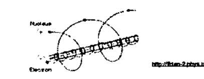
$$\begin{tikzcd} & nT\tau \arrow[dl] \arrow[d] \arrow[dr] & \\ \text{Density} & \text{Temperature} & \text{Confinement Time} \end{tikzcd}$$



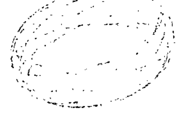
Confinement :

- *Gravitational confinement - as in stars*
- Magnetic Confinement of plasma
- Inertial confinement

Magnetic Confinement

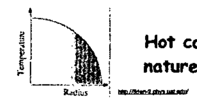


Most devices are toroidal shaped.



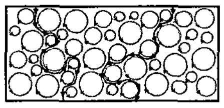
Used in ITER, DIII-D, JET, JT-60 ...

Problems ...



Hot core, cool walls - but nature abhors gradients

Turbulence relaxes the gradient.
Turbulent transport reduces confinement time.



Challenge : TO CONTROL THE TURBULENT TRANSPORT

Hierachy of numerical modelling - to study turbulence

- 1-D - Envelope models, simple, fast, but dynamics may not always be realistic. However, often used for real experiments.
- 2-D - Models using primitive equations.
- 3-D fluid model & 3-D particle model - Also use primitive equations. Realistic, but, much more complicated computationally.

Model lowest in Hierachy:
Noise in Reaction-Diffusion Equation like Transport Model

An Overview

- Transport model:

$$\frac{\partial n}{\partial t} = S_{NBI} + S_{w} + \frac{1}{r}\frac{\partial}{\partial r}\left[rD_n\frac{\partial n}{\partial r}\right]$$

$$\frac{3}{2}\frac{\partial nT_i}{\partial t} = \frac{1}{r}\frac{\partial}{\partial r}\left[r\left(\chi_i n\frac{\partial T_i}{\partial r} + \frac{5}{2}D_n T_i\frac{\partial n}{\partial r}\right)\right] - D_n\frac{1}{n}\frac{\partial n}{\partial r}\frac{\partial nT_i}{\partial r} - Q_{NBI} + Q_e(T_e - T_i)$$

$$\frac{3}{2}\frac{\partial nT_e}{\partial t} = \frac{1}{r}\frac{\partial}{\partial r}\left[r\left(\chi_e n\frac{\partial T_e}{\partial r} + \frac{5}{2}D_n T_e\frac{\partial n}{\partial r}\right)\right] + D_n\frac{1}{n}\frac{\partial n}{\partial r}\frac{\partial nT_e}{\partial r} + Q_{OH} + Q_{ohm} + Q_e(T_i - T_e)$$

- System is closed with a Fluctuation envelope equation

$$\frac{\partial \varepsilon}{\partial t} = \left\{\gamma - \alpha_1\varepsilon - \alpha_2\left[\frac{r}{q}\frac{\partial}{\partial r}\left(\frac{q}{r}\frac{E_r}{B_\theta}\right)\right]^2\right\}\varepsilon + \frac{1}{r}\frac{\partial}{\partial r}\left[rD_\varepsilon\frac{\partial \varepsilon}{\partial r}\right]$$

and the equation for E_r [illegible]

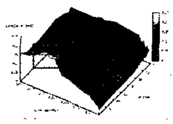
Waves in the model

- Transition model - used to successfully explore transport dynamics, and has physical propagating waves.
- But, waves propagate unphysically far possibly due to lack of turbulent decorrelation in an envelope model.
- Noise added to reaction diffusion equations has been found to significantly effect wave propogation.
- So we added noise to see its effect

The noise was varied in 2 ways:

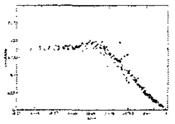
- Amplitude of Noise
- Number of sites perturbed

Results - Key Points



• In a region of low noise amplitude, correlation length increases with amplitude, consistent with noise enhanced propagation in other systems !

• A large number of sites perturbed by a large amplitude noise - reduced correlation



$noise = (amplitude)^{\frac{3}{2}} \times (pthresh)$

Paper in progress

Next Level of Model Hierachy - 2D Primitive Equation Model

Parareal Algorithm -

a new technique to parallelize turbulence codes

Motivation

• Simulations of fusion plasma is numerically very challenging - e.g., studying long time evolution (hundreds or thousands of eddy decorrelation times) of the transport dynamics is needed to understand turbulent transport on profile evolution timescales (in many plasmas and neutral fluids).

• Space parallelization is not enough.

• Is time parallelization an option? Well, parareal algorithm has been reported to have given significant speedup for simple codes.

• Had not (yet) been applied to a turbulence code.

Overview of the Parareal Algorithm ...

Parareal Algorithm : Distinct in many ways

• Algorithm first proposed by Lions et al. in 2001.

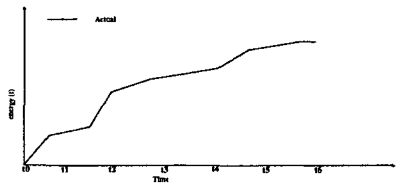
• Parallelizes in time, despite the sequential nature of the time domain.

• Very non-intuitive as this is an initial value problem, and the result of each time step should depend on that of the previous timestep. <u>However, in this case, "timesteps" (chunks) are solved in parallel.</u>

• Decomposition of space domain - hithereto been the only approach to solve partial differential equations using parallel processors.

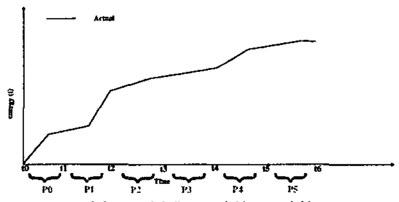
• Uses predictor - corrector approach.

F is a propagator evolving the function (energy(t)) from initial time, to, to a later time ...



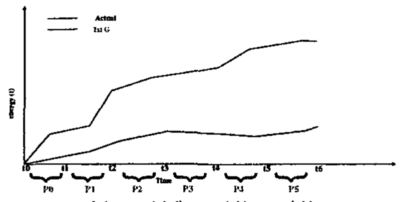
$$\lambda_{i+1}^{k+1} = G_{\Delta t}\left(\lambda_{(i)}^{k+1}\right) + F_{\Delta t}\left(\lambda_{(i)}^{k}\right) - G_{\Delta t}\left(\lambda_{(i)}^{k}\right)$$

F is a propagator evolving the function (energy(t)) from initial time, t_0, to a later time ...



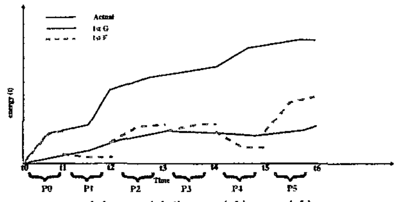
$$\lambda_{i+1}^{k+1} = G_{\Delta t}\left(\lambda_{(i)}^{k+1}\right) + F_{\Delta t}\left(\lambda_{(i)}^{k}\right) - G_{\Delta t}\left(\lambda_{(i)}^{k}\right)$$

F is a propagator evolving the function (energy(t)) from initial time, t_0, to a later time ...
G - faster but inaccurate propagator



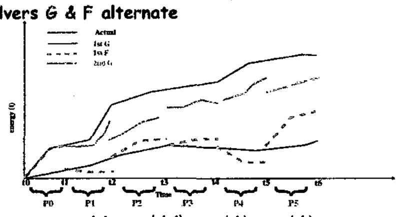
$$\lambda_{i+1}^{k+1} = G_{\Delta t}\left(\lambda_{(i)}^{k+1}\right) + F_{\Delta t}\left(\lambda_{(i)}^{k}\right) - G_{\Delta t}\left(\lambda_{(i)}^{k}\right)$$

F is a propagator evolving the function (energy(t)) from initial time, t_0, to a later time ...
G - faster but inaccurate propagator



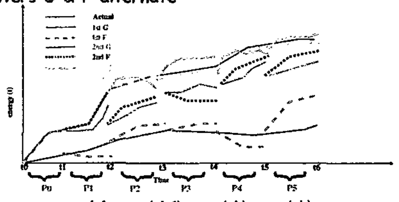
$$\lambda_{i+1}^{k+1} = G_{\Delta t}\left(\lambda_{(i)}^{k+1}\right) + F_{\Delta t}\left(\lambda_{(i)}^{k}\right) - G_{\Delta t}\left(\lambda_{(i)}^{k}\right)$$

F is a propagator evolving the function (energy(t)) from initial time, t_0, to a later time ...
G - faster but inaccurate propagator
Solvers G & F alternate



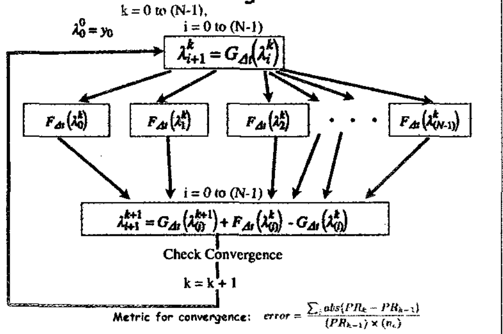
$$\lambda_{i+1}^{k+1} = \underset{\text{New } G}{\underset{\uparrow}{G_{\Delta t}\left(\lambda_{(i)}^{k+1}\right)}} + F_{\Delta t}\left(\lambda_{(i)}^{k}\right) - \underset{\text{Old } G}{\underset{\uparrow}{G_{\Delta t}\left(\lambda_{(i)}^{k}\right)}}$$

F is a propagator evolving the function (energy(t)) from initial time, t_0, to a later time ...
G - faster but inaccurate propagator
Solvers G & F alternate



$$\lambda_{i+1}^{k+1} = \underset{\text{New } G}{\underset{\uparrow}{G_{\Delta t}\left(\lambda_{(i)}^{k+1}\right)}} + F_{\Delta t}\left(\lambda_{(i)}^{k}\right) - \underset{\text{Old } G}{\underset{\uparrow}{G_{\Delta t}\left(\lambda_{(i)}^{k}\right)}}$$

Basic Algorithm



k = 0 to (N-1),
$\lambda_0^0 = y_0$
i = 0 to (N-1)
$\lambda_{i+1}^{k} = G_{\Delta t}\left(\lambda_i^k\right)$
$F_{\Delta t}\left(\lambda_0^k\right)$
$F_{\Delta t}\left(\lambda_1^k\right)$
$F_{\Delta t}\left(\lambda_2^k\right)$
. . .
$F_{\Delta t}\left(\lambda_{(N-1)}^k\right)$
i = 0 to (N-1)
$\lambda_{i+1}^{k+1} = G_{\Delta t}\left(\lambda_{(i)}^{k+1}\right) + F_{\Delta t}\left(\lambda_{(i)}^{k}\right) - G_{\Delta t}\left(\lambda_{(i)}^{k}\right)$
Check Convergence
k = k + 1

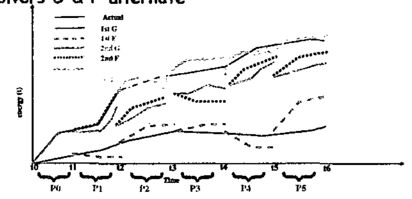
Metric for convergence: $error = \frac{\sum_i abs(PR_k - PR_{k-1})}{(PR_{k-1}) \times (n_c)}$

A Modification to the Algorithm

In standard parareal algorithm, at the kth iteration, the sequential solver, G is started from processor P_k. However, if the algorithm is working successfully, n processors have attained convergence at the kth iteration, where n > k. We have used this characteristic to improve the algorithm.

We replaced P_k by P_n as the starting point for G and all subsequent calculations at iteration k. This change led to a quicker convergence of the entire time series.

F is a propagator evolving the function (energy(t)) from initial time, t_0, to a later time ...
G - faster but inaccurate propagator
Solvers G & F alternate



$$\lambda_{i+1}^{k+1} = \underbrace{G_{\Delta t}\left(\lambda_{(i)}^{k+1}\right)}_{\text{New } G} + F_{\Delta t}\left(\lambda_{(i)}^{k}\right) - \underbrace{G_{\Delta t}\left(\lambda_{(i)}^{k}\right)}_{\text{Old } G}$$

Success of Algorithm Depends on Multiple Factors

Algorithm always converges if k=N.
But, success in achieving significant speedup if

- k<<N.

G is much cheaper than F.

- "Good" G: Solutions converge
- "Bad" G: Solutions diverge
- No "fixed recipe" for G !
- Despite solutions being very sensitive to initial conditions for turbulent or chaotic systems - it is possible to choose G.

Application of the Parareal Algorithm ...

Parareal Algorithm has been successfully applied to the Lorenz System

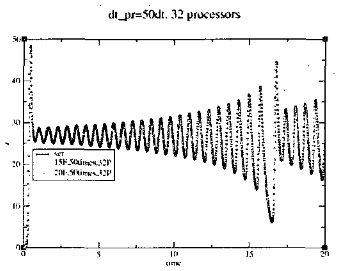
Ref: M.J. Gander & E. Hairer(2008)

$$\frac{dx}{dt} = \sigma(y - x)$$
$$\frac{dy}{dt} = x(\rho - z) - y$$
$$\frac{dz}{dt} = xy - \beta z$$

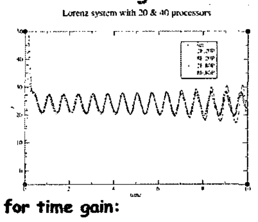
$\sigma = 10$

$\varrho = 28$

$\beta = 8/3$

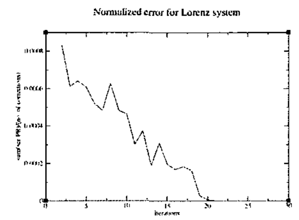


In Region explored, varying Chunk-Size does not affect k, required for Convergence



Estimation for time gain:
If convergence in k=10 with 180 processors & time for G is neglected,
time_{serial} = 18 * $\text{time}_{parareal}$ Ref: M.J. Gander & E. Hairer(2008)

Normalized error decreases with increasing Iterations



As the solutions converge towards the serial solution, the normalized error reduces below prescribed tolerance.

Moving from the "simple" Lorenz system to a much more complex Turbulent system . . .

Two-nonlinearity Model for DTEM

$$\frac{\partial}{\partial t}\left(1-\rho_s^2\nabla_\perp^2\right)\tilde{n}+D\frac{\partial^2\tilde{n}}{\partial y^2}+V_D\frac{\partial\tilde{n}}{\partial y}- -L_nD\left[\nabla_\perp\left(\frac{\partial\tilde{n}}{\partial y}\right)\times\mathbf{z}\right]\cdot\nabla_\perp\tilde{n}+ +\rho_sc_s\left(\nabla_\perp\tilde{n}\times\mathbf{z}\right)\cdot\nabla_\perp(\rho_s^2\nabla_\perp^2\tilde{n}))=0$$

$$D=\frac{3\sqrt{\epsilon}(\rho_sc_s)^2}{2L_nL_T\nu_{eff}}$$

$$V_D=c_s\rho_s/L_n$$

- Evolution solved by spectral code in k_x, k_y space, often of size 385 x 385:

$$i\frac{\partial\tilde{n}_k}{\partial t}-\frac{V_Dk_y+ik_y^2D}{1+k^2\rho_s^2}\tilde{n}_k+ +\frac{i}{1+k^2\rho_s^2}\left(\alpha_{E\times B}N_k^{E\times B}+\alpha_PN_k^{POL}\right)=$$

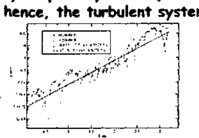
Control knobs
$\beta, \alpha_{E\times B}, \alpha_P$

- to which sources T_k (in k space) and sinks S_k may be added
- an external shear at (k_x=0, k_y=±1) may be added to explore its effects on turbulence transport.

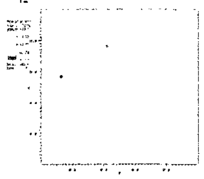
Moving from the "simple" Lorenz system to a much more complex Turbulent system ...

- 296450 non-linearly coupled equations (the Lorenz system had 3 such equations) ... hence, the turbulent system is much more complex

Has positive Lyapunov exponent

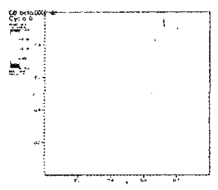


Vorticity in turbulent system with polarization non linearity

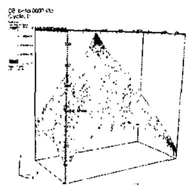


Turbulent system with E X B non linearity

Vorticity in turbulent system with EXB non linearity



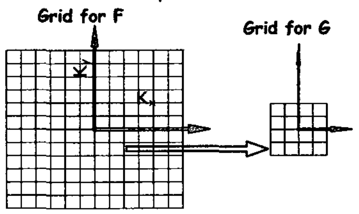
Power Spectrum in k space



Selecting Optimum Coarse Solver is Important

Different approaches can/should be explored to find G.
Any one of them, or a combination of them, may work :
- Some of the physics may be ignored when solving with G, to achieve speedup.
- G can be same as F, but may be solved over a coarser k-mesh (or spatial grid).
- G may be same as F, but may be solved with a larger timestep (dt) and less accuracy.
- Use a different G.

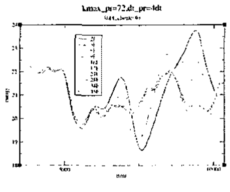
Reducing the Grid Size in K-space speeds G.



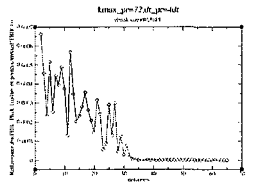
Since most energy is concentrated at core,
only the core portion of the grid used in F was used in G.
But, core should be big enough to include dissipation scales.
Also, reduced grid allows larger dt for evolution.

Convergence is found with reduced k-space & larger dt in G

Turbulent system with E X B non linearity



Convergence at 34 iterations for 80 processors.

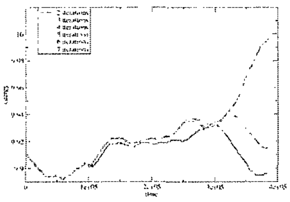


The error falls off with iterations.

Total energy of the system is used for convergence measurement.

Convergence is again found with reduced k-space & larger dt in G

Turbulent system with polarization non linearity

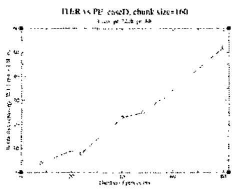


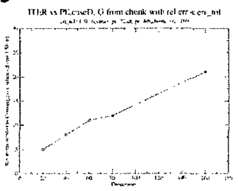
Convergence at 7 iterations for 640 processors.

Total energy of the system is used for convergence measurement.

Modification to the Algorithm ...

Modification to the Algorithm significantly reduces the Number of Iterations needed for Convergence





For a fixed chunk size=160, convergence occurs at 22 iterations for 160 processors with the modification to the scheme. With the unmodified version, for the same chunk size with 80 processors, 53 iterations were required for convergence!

A Theoretical Analysis of Efficiency ...

The Scaling varies with the Chunk Size per Processor as well as the total Simulation Time

Strong Scaling:

- Total simulation time (ntstep) constant.
- Number of processors varied.
- So, the chunk size decreases linearly with increasing number of processors.
- Ideally, $T_{PA} = 1/N$.

Weak Scaling:

- Chunk size constant.
- Number of processors varied.
- So, the total simulation time (ntstep) increases linearly with the processors.
- Ideally, T_{PA} = constant .

Strong Scaling : Model predicts Two Regimes with Number of Processors exist for the Gain due to Parallelization

The gain, H_{PA} may be defined as :

$$H_{PA} = \frac{T_F^{ser}}{T_{PA}} = \left[k(N)\left(\frac{1}{\beta} + \frac{1}{N}\right)\right]^{-1} \text{ where } \beta = \frac{T_F^{ser}}{T_G^{ser}} \; \& \; k(N) \simeq N^{\alpha}$$

So $$H_{PA} = \left[N^{\alpha}\left(\frac{1}{\beta} + \frac{1}{N}\right)\right]^{-1}$$

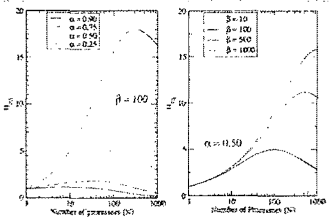
Hence, the number of processors giving maximum gain:

$$N_{max} = \beta\left(\frac{1-\alpha}{\alpha}\right), \alpha \neq 0$$

Hence, the two regimes are:

$$H_{PA} \simeq N^{1-\alpha}, N << \beta \qquad H_{PA} \simeq \beta N^{-\alpha}, N >> \beta$$

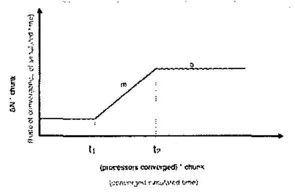
H_{PA} depends on β & α



For a fixed β, H_{PA} decreases with increasing α.

For a fixed α, H_{PA} increases linearly with β.

Convergence rate has two parameter regimes



Convergence rate is low for $t < t_1$ and is b for $t > t_2$

Replacing old Solver in G with Runge-Kutta increases β

Time for typical F runs for NTSTEP=25600:25.92hrs=1555.2mins

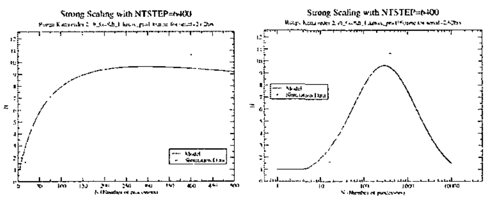
Timings for serial runs in G using Runge Kutta with kmax=72 (for NTSTEP=25600):

G solver	Time (mins)	Beta
RK2,dt_G=2dt_F	14.56	106.81
RK2,dt_G=8dt_F	3.625	429
RK2,dt_G=16dt_F	1.82	854.5
RK2,dt_G=32dt_F	UNSTABLE (NANs)	
RK4,dt_G=2dt_F	29.2	53.26
RK4,dt_G=8dt_F	7.25	214.51
RK4,dt_G=16dt_F	3.651	426
RK4,dt_G=32dt_F	1.82	854.5
RK4,dt_G=40dt_F	1.46	1065.2
RK4,dt_G=64dt_F	0.91	1709
RK4,dt_G=80dt_F	UNSTABLE (NANs)	

Balancing β & k(N) maximizes H

cases with kmax_pr=72	Iterations to converge	Time for convergence(hr)
VODPK in G	11	2.88
RK order4,dt_pr=2dt_ser	11	2.04
RK order4,dt_pr=4dt_ser	11	1.76
RK order4,dt_pr=8dt_ser	10	1.355
RK order4,dt_pr=16dt_ser	11	1.43
RK order4,dt_pr=40dt_ser	11	1.326
RK order2,dt_pr=2dt_ser	10	1.58
RK order2,dt_pr=4dt_ser	10	1.375
RK order2,dt_pr=8dt_ser	10	1.245
RK order2,dt_pr=16dt_ser	12	1.47

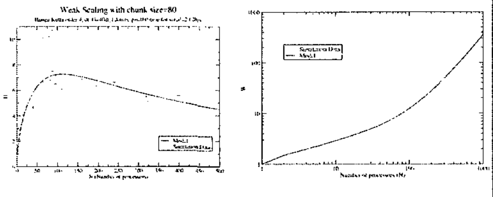
Strong Scaling : Using Runge-Kutta solver in G greatly enhances the gain, H



H = T_ser / T_PA

A gain of 10.69 is achieved with 400 processors.

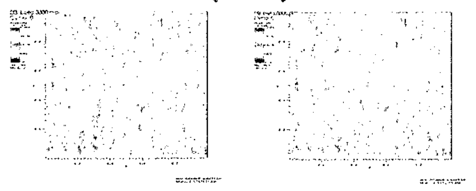
Weak Scaling : Using Runge-Kutta solver in G greatly enhances the gain, H



H = T_ser / T_PA   W = T_PA / T_ser(chunk)

A gain of 8.805 is achieved with 88 processors.

Serial & Parareal Solutions : Similar (Movies)



Serial Solution   Parareal Solution

Conclusions

- Improvement of the 1D model & implementation of the parareal algorithm to 2D model enhance our ability to understand turbulence & turbulent transport.
- Noise added to a 1D reaction diffusion model can act as a decorrelation surrogate.
- With the success of the parareal algorithm, the study of the dynamics of turbulent transport at transport timescales and other relevant problems in fusion plasma may be made possible.
- Parallelization in time - possible for a turbulent system and Parareal Algorithm works for such a case. Two cases have been explored.
- For a successful application of the parareal algorithm to a turbulence code, a proper choice of the coarse solver, G, is critical.
  - G with a reduction of grid size & use of a Runge Kutta solver appears to work best for our case.
- A successful implementation of the parareal algorithm may greatly enhance computational efficiency, and larger scalability with respect to processor numbers.

Future Work

Further improvement can be pursued by:

- new strategies to improve G (L.Chacon, ORNL),
- hybrid parallelization (space + time),
- reuse of processors already having attained convergence,
- portable parareal framework (L.Berry, W. Elwasif, ORNL)

Acknowledgements:

* D. E. Newman & R. Sánchez - without their tireless help and input - this project was/is impossible.

* Consultants & Staff at ARSC (Arctic Region Supercomputing Center) for their incredible patience, help & enthusiasm.

* ARSC for computer time.

References

- Y. Liang et al, Phys. Fluids B 5, 1128 (1993)
- D. E. Newman et al, Phys. Fluids B 5, 1140 (1993)
- J.A. Mier et al, Physical Review Letters (accepted, 2008)
- J.L. Lions et al, C.R. Acad. Sci. Paris, Serie I, 332 (2001), pp. 661668
- L. Baffico et al, Phys. Rev. E 66, 057701 (2002)
- M. Gander & S. Vandewalle, SIAM J. on Sc. Comp. 29(2) (2007)
- Y. Maday & G. Turinici, Proceedings of the 15th International Conference on Domain Decomposition Methods, Berlin, July 21-25, 2003
- P. F. Fischer et al, A Parareal in Time Semi-implicit Approximation of the Navier-Stokes Equations
- M.J. Gander & E. Hairer, Nonlinear Convergence Analysis for the Parareal Algorithm (Domain Decomposition Methods in Science and Engineering XVII)

Thank you.