A POSTERIORI ERROR ESTIMATOR FRAMEWORK FOR PDE'S

A Dissertation

Presented to

the Faculty of the Department of Computer Science

University of Houston

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

By

Christophe Picard

December 2007

A POSTERIORI ERROR ESTIMATOR FRAMEWORK FOR PDE'S

Christophe Picard

APPROVED:

Marc Garbey, Chairman
Department of Computer Science

Thierry Colin
Departement de Mathematiques appliquees,
Universite de Bordeaux 1

Edgar Gabriel
Department of Computer Science

Venkat Subramanian
Agile Developer, Inc.

Shishir Shah
Department of Computer Science

Dean, College of Natural Sciences and Mathematics

Acknowledgements

A doctoral dissertation is never the achievement of one person alone but of people that are collaborating at different levels to produce an innovative work.

The first person I would like to thank is my adviser, Dr. Marc Garbey, for his guidance and supervision during these last four years. He gave me the freedom and time to explore some ideas on my own, and he always gave great support in overcoming all the crises. He also shares his passion for his work with no bounds. I hope that one day I will become as passionate, and as a good adviser to others as he has been to me.

I am also grateful to Dr. Thierry Colin, my co-adviser, who gave me the opportunity of doing this dissertation in collaboration with the University of Bordeaux. Without him, this dissertation could not have been possible.

I would like to express my gratitude to my exceptional doctoral committee. I wish to thank Dr. Venkat Subramanian for the invaluable experience he shared with me and which greatly improved the quality of my dissertation work. I also thank Dr. Shishir Shah and Dr. Edgar Gabriel for their different input and encouragement.

I want to express my appreciation for the former and current students of the Modeling and Computational Science group with whom I have interacted and shared many experiences all these years: Francois Pacull, Bilel Hadri, Hatem Ltaief, Mallek Smaoui, Nicolas De Brye, Benoit Megrat, Guillaume Garbey and Guillaume Tran-Son-Tay. I would also like to thank Dr. Petri Fast for giving me an original point of view on my research. I want to extend my gratitude to Victoria Hilford and Justin Hansen who took the time to read this work and suggest corrections.

I wish to extend my recognition to Marie-Noelle Garbey who cared and gave me her precious support.

To the friends and relatives who believed in me all these past years, I say thank you.

Finally, I would like to thank Prof. Bertrand Thomas who first gave me his taste and passion for mathematics; it has kept me motivated all these years.

This work was sponsored by Sandia National Laboratory, the multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

A POSTERIORI ERROR ESTIMATOR FRAMEWORK FOR PDE'S

An Abstract of a Dissertation

Presented to

the Faculty of the Department of Computer Science

University of Houston

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

By

Christophe Picard

December 2007

Abstract

The Least Square Extrapolation (LSE) method for solution verification was introduced in 2002. Since then, the method has forked into three areas. The method was extended to the study of stiff elliptic problems. Techniques developed in this new framework allow having a rigorous upper-bound error estimator to predict very fine grid solutions. This work is applicable to the pressure solver in the Immersed Boundary Method. But the equations are still steady. In this dissertation, we expanded the method to two more types of problems. First, the LSE method was extended to parabolic equations by using coarse grid solutions that have different meshes in space and time, with minimum overhead on memory. Finally, we designed a new method that offers a general framework to do solution verification efficiently by processing the underlying set of discrete (non)-linear equations without using *a priori* information on the approximation theory framework that is applied to solve the PDE. A library was developed to compute the optimized extrapolation using a surface response methodology. Any 3D Navier-Stokes code can be plugged into it to compute the optimum without knowledge of the internal structure of the code. This work has evolved by establishing the conditioning number of the problem in a reduced space that approximates the main feature of the numerical solution, thanks to a sensitivity analysis. Overall our method produces an *a posteriori* error estimation in this reduced space of approximation. Standard benchmark problems showed that more information can be extracted than by using Richardson Extrapolation.

Contents

List of Figures

Chapter 1

Introduction

1.1 Concept of solution verification

On December 26^{th}, 1999, the winter storm Lothar (Figure 1.1) passed over France and Germany causing more than $6.2 billion in damages and more than 100 deaths. The short-range weather forecasting model used by the German Weather Service had failed to predict the path of the storm, increasing the damages. Later analysis concluded that the automated system had found some outliers data that were unusual for European weather and had discarded them. Later simulations were unable to correctly predict the impact of the storm. Research on weather forecasting is still an active topic.

While not exactly a software problem, the Intel Pentium bug of 1994 is one of the most well-known bugs in history. An error in the look-up table for floating-point

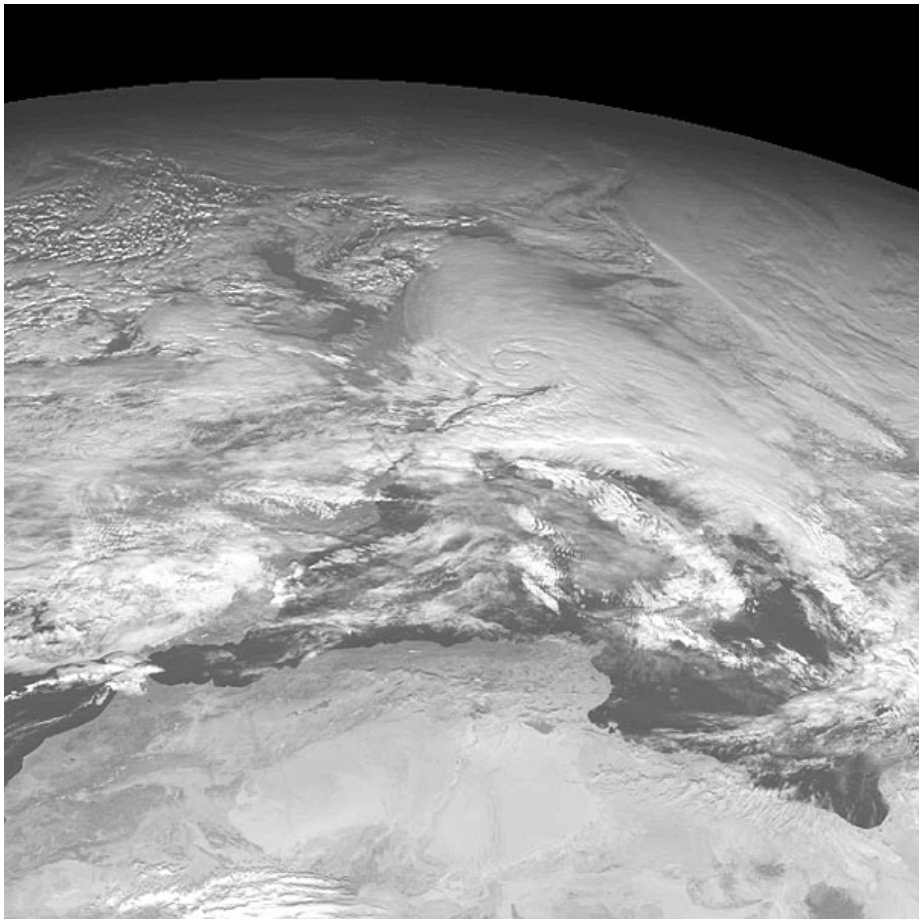

Figure 1.1: Lothar winter storm system over Europe.

divisions created errors ranging from one out of 10,0000 to one out of one quadrillion. The cost of this bug was evaluated at $400 million.

In September 1999, Mars Climate Orbiter crashed on Mars instead of reaching a safe orbit. The underlying cause of this accident was a failure to convert English measures to metric measures.

On February 25^{th}, 1991, at Dhahran, Saudi Arabia, a Patriot missile defense

system failed to track and intercept an incoming Iraqi Scud. The Scud hit an army barrack, killing 28 soldiers. The post-accident investigation found that the failure was due to an inaccurate calculation of the time since boot, due to computer arithmetic errors.

On August 23^{rd}, 1991, in North Sea, during the lowering process below the ocean surface, the hull of the Sleipner A oil platform sprang a leak and sank into the 220 m fjord (Figure 1.2). This event triggered seismic waves at the magnitude of 3.0 on the Richter scale. Post-accident investigation concluded that the loss was due to a failure in a cell wall, leading to a leakage that the deballasting pumps could not handle. The root cause for this failure was an inaccurate finite element approximation of the linear elastic model of the tricell in the program NASTRAN. The shear stresses on the ballast chambers were underestimated by 47%. The investigations that followed evaluated the cost of the accident to be \$700 million. After the accident, a more careful finite element analysis has been capable of predicting the water depth at which failure occurred.

Each of these examples illustrates differently a particular type of problem. The failure in predicting the storm system in Europe is a consequence of a defective model. The problem of the Intel chip is an example of accuracy error. The crash of the Mars probe is a coding error.

The last two examples are examples of arithmetic errors that could have been detected by careful error analysis.

They all illustrate the impact of simulations in the real world. As mathematical

Figure 1.2: Sleipner A offshore platform under construction

models gain in complexity and as the available computing resources increase, it is important to consider the reliability of simulations when making decisions.

As mentioned in [1, 2], simulations share common points with both the experimental approach and theory. However, it also has its own particularities. For instance, simulation allows testing and improvement of theoretical models. On the other hand, simulation is experimental since an engineer can modify the input at will to generate new data. As such, most people often have considered it to be a peer methodology of the two more traditional approaches.

One of the strengths of simulation is to enable a way of performing any virtual experimentation for problems or systems for which real experimentation is not possible at a reasonable cost. Weather forecasting is such an application; it is hardly possible, if not impossible, to reproduce a specific atmospherical state in a controlled environment. It is therefore essential to ensure the reliability of solutions produced by simulations.

Questions that arise include these:

- Are there any tools that allow applied mathematicians to verify multiphysics and multiscale problems?
- If yes, what are these tools ?
- What are the industry standards for these tools?

These questions lead to the concepts of validation and verification.

It is important to make a distinction between code validation, code verification and solution verification. From [3], whenever code verification applies to simulation codes, it consists of finding and removing mistakes in the source code, finding and removing errors in the numerical algorithms, and improving software quality assurance practice. Code validation performs the assessment of the accuracy of the model compared to its intended usage. Finally, solution verification deals with the accuracy of input and output data and provides an estimate of the error for numerical solutions. In other words, the goal of solution verification is to give a quantitative evaluation of the numerical error of a given solution to partial differential equations

(PDEs). An exhaustive and rigorous analysis of this error is not always possible for complex PDEs. Some of the methods addressing this problem are explicit discretization, robustness and convergence studies, formal error estimation procedures, and inferences from test problem suites and from previous experiments.

In order to estimate errors in a numerical solution to a PDE, two methods are at our disposal: *a priori* and *a posteriori* error estimates. *A priori* error estimates use only information about the discretization, the initial condition, and the boundary conditions. *A posteriori* error estimates make use in addition of anterior numerical solutions. In the following, we will focus only on *a posteriori* error estimates.

1.2 Concept of *a posteriori* error estimates

1.2.1 *A priori* error estimates

The first step in estimating the error produced by a numerical method is of a predictive nature. Indeed, one can attempt to estimate the error before actually performing the computation by analyzing the partial differential equation, the initial condition, and the boundary conditions. These types of estimations are *a priori* error estimates. The goal is to successively bound the error produced by each step of the approximation, and then by accumulation phenomena obtain a global error estimate.

In the context of solution verification, *a priori* error estimates serve as guidelines, since they cannot give any numerical evaluation of the errors [3].

One of the underlying requirements in the computation of an *a priori* error estimate is the consistency of the discretization method. One of the constraints to obtain consistency in the discretization is the smoothness of the PDE solution. The lack of regularities on the discretization limits the application of truncation error estimates. For instance, in [4] the authors show that for some incompressible transient flows the solutions lack smoothness, making *a priori* estimates inaccurate.

An added condition for the convergence is the stability of the discretization. Under limited conditions, among which is linearity, a problem converges to the correct solution if and only if it is stable. So being able to prove that a complex non-linear discretization is stable will lead to convergence properties, enabling the use of *a priori* error estimates for the problem.

A priori error estimates have only theoretical meaning. They show that the approximation method is correct in principle. However, in practice we are strongly interested in the error for the concrete approximation on the concrete mesh. Moreover, it is possible to merge *a priori* information with data acquired from previous computations. This leads to the notion of *a posteriori* error estimates.

1.2.2 Error estimates and Richardson Extrapolation

A definition of *a posteriori* taken from word-net is

A posteriori: involving reasoning from facts or particulars to general principles or from effects to causes.

Consequently, *a posteriori* error estimates for PDEs could be defined as methods

for evaluating the error present in an approximated solution of a partial differential equation from

- *a priori* information;
- computational results of a previous numerical solution using the same numerical algorithm on the same PDE from initial and boundary data; and
- information extracted, such as estimates or convergence characteristics.

In using *a posteriori* error estimates, the goal is to detect errors in the discretization. In order to detect errors introduced by the model, the original data, or the code, engineers should refer to other techniques. The governing differential equations and the approximated solutions are means of providing an accurate estimate of the amount of error in the problem domain.

A common aspect among a large majority of methods is their origin in Richardson Extrapolation. It combines several meshes to approximate the error. These methods rely on the assumption that a Taylor series expansion estimating the numerical error exists such that:

$$e_{u_i} = u - u_{h_i} = \sum_{k=1}^{n} \alpha_k h_i^{p_k}, \tag{1.1}$$

where u denotes the exact solution, u_{h_i} is an approximated solution on specific mesh i, n is the order of the expansion, α_k is a constant in the Taylor expansion for the given grid, h_k is a parameter that characterizes the coarseness of the mesh on which the approximation is performed, and p_k is giving the order of accuracy of the method. This leads to an equation with $2n+1$ unknowns, hence the need to have $2n+1$ grids

with a numerical solution to obtain an approximation of the error.

As specified in [5], most approaches have considered only the asymptotic range; hence, only the first term of the Taylor expansion is required :

$$e_{u_i} = u - u_{h_i} = \alpha_1 h_i^{p_1}, \tag{1.2}$$

leaving only three unknowns to be estimated.

1.2.3 Richardson Extrapolation for known order

In this section, we look at a basic formulation of Richardson Extrapolation. Let E be a normed linear space, $||.||$ its norm, $u \in E$, $p > 0$, and $h_i \in (0, h_0)$. $u_i \in E$, $i = 1 \dots 3$ have the following asymptotic expansion:

$$u_i = u + \alpha h_i^p + \delta_i, \tag{1.3}$$

with C a constant independent of h_i, and $||\delta_i|| = O(h_i^p)$.

Let $w_i = \frac{h_i}{h_{i+1}}$.

For known p, the Richardson Extrapolation formula

$$u_i^r = \frac{w_i^p u_{i+1} - u_i}{w_i^p - 1},\ i = 1, 2 \tag{1.4}$$

provides improved convergence: $||u - u_i^r|| = o(h_i^p)$. An *a posteriori* error estimate on u_i is then

$$||u_i - u_i^r||. \tag{1.5}$$

In practice, Richardson Extrapolation applies to grid functions rather than to continuous functions. Let E_i be a family of normed linear spaces, associated with a

mesh M_{h_i}. Equation 1.6 is the asymptotic model for the discrete grid functions:

$$U^i = v + C_i h_i^p + \delta_i, \tag{1.6}$$

with $C_i = (1 + \varepsilon_i)C$, and $\varepsilon_i = O(1)$. δ_i is a model for the h independent numerical perturbation induced by consistency errors and arithmetic error. The Richardson Extrapolation

$$V_2^r = \frac{w_2^p U_3 - U_2}{w_2^p - 1}, \tag{1.7}$$

defined on grid points of M_2 has then for error E_2,

$$E_2 = v - V_2^r = \frac{1}{w_2^p - 1}\left((\delta_2 - w_2^p \delta_3) + C\,(\varepsilon_2 - \varepsilon_3)h_3^p\right). \tag{1.8}$$

The numerical perturbation is amplified by a factor $\frac{w_2^p+1}{w_2^p-1}$. Richardson Extrapolation provides then an *a posteriori* error estimate on U^i that is simply

$$||V_2^r - U_i||,\ i = 1 \ldots 3. \tag{1.9}$$

Richardson Extrapolation also supplies a formula to compute an approximation of a finer mesh solution, for example:

$$U_4 \approx V_2^r + C h_4^p, \tag{1.10}$$

where C is obtained from the identity

$$U_3 = V_2^r + C h_3^p. \tag{1.11}$$

For complex applications, it is no possible to know or to satisfy closely the order of convergence on the computational grid. One may use the estimate:

$$p \sim \log_2 \frac{||u_1 - u_2||}{||u_2 - u_3||}. \tag{1.12}$$

An entirely similar analysis can be applied to the non-embedded refined grid solution U^i in a normed linear space E_i, associated with a mesh M_{h_i}, provided that one projects all grid functions to a fine grid M^0 with an interpolation procedure. However, this interpolation introduces an additional error term to add to the δ_i term of Equation 1.6. This error still remains much less than the expected convergence accuracy h_i^p.

In practice, all pointwise Richardson Extrapolation formulas, particularly in Equation 1.6, are sensitive to numerical perturbation.

Richardson Extrapolation allows an easy description of the concept of *a posteriori* error estimates, but it is limited in handling problems that are more complex. For instance, Richardson Extrapolation alone might not be suitable for problems with noisy data, for stiff problems or for meshes without enough regularity. In the next section, an overview of different *a posteriori* error estimates is made.

1.3 Overview of methods in solution verification

A standard and well-known procedure to establish *a posteriori* estimates is to solve local residual problems. The so-called "equilibrated residual method" is one of the most reliable and accurate techniques [6, 7, 8]. However, theory for this method has been essentially limited to linear PDEs. Examples of applications in the literature include the Poisson problem, the linear singular perturbation problem with simple boundary layers, the Stokes and Oseen equation, and linear elasticity theory. This family of techniques has the advantage of being mathematically rigorous in the finite

element framework. It can be generalized, up to a certain point and on a case per case basis, to non-linear elliptic problems or hyperbolic equations [9, 10, 11, 12, 13, 14, 15, 16, 17, 18]. However, it may not apply to finite volumes computation, where there is no equivalence theorem with finite element formulations. We also notice that the estimates given by the theory do not provide a quantitative free bound on the error. In practice, the unknown constants in the error estimates obtained by this theory have naturally a bad asymptotic behavior for boundary layer problems as the disparity of scales increases, unless one refines the grid. Also, the method has to be significantly modified depending on the existence of additional length scales, such as boundary layers or not, or to take into account the influence of the error in discretization of non-homogeneous boundary conditions [7].

More recently, a general framework for finite element *a posteriori* error control that can be applied to linear and non-linear elliptic problems has been introduced by Patera *et al.* [19, 20, 21, 22]. This new theory focuses on the fact that one is not necessarily interested in the solution u itself, but rather in a linear (local) functional output $Q(u)$ or a stress $\sigma(u)$. For the Incompressible Navier-Stokes equation, it is possible to build *a posteriori* finite element free constant output bounds. The procedure to construct the *a posteriori* estimate on a given triangular mesh of step 'H' uses a fine grid solution 'h' as a reference solution. The construction of upper and lower bounds of linear-functional outputs of the PDE solution demonstrates the efficiency of the method. Examples considered are the Helmholtz problem and the viscous Burgers equation in one space dimension, 1^{st} order linear convection in two-dimensional spaces, and the Navier-Stokes equations written in Boussinesq

approximations in a highly convective flow regime.

This procedure uses the concept of duality that draws on an equivalent dual adjoint formulation of the primal. Then, the adjoint variable associates the error in the functional and the local residual errors of the primal solution. This method is promising and technically impressive. Its implementation seems, at first sight, complex, and is restricted to the finite element framework with appropriate so-called "broken space" to relate the coarse mesh space with the fine mesh approximation. The results on the bounds hold for $H \rightarrow h$ but for all $H < H^*$ where H^* is generally an unknown threshold discretization parameter [19]. A traditional drawback is the fact that the coarse mesh solution H may not be good enough to provide useful information, and that is why the method uses refinement to build confidence in the output results. Therefore, a large Reynolds number is an issue in such a computation. Furthermore, a reliable error estimate on the fine grid solution of step 'h' does not necessarily lead to a reliable error estimate for the true solution, unless one has external knowledge to guess what will be a good, fine grid solution. Notice that [23] and [24] present a practical solution to the error estimate of functional outputs that is used to drive grid refinement on complex CFD problems. In addition, an adjoint-based error correction procedure is very close to the engineering point of view.

It seems that there is a lot of new activity on *a posteriori* estimates based on residual methods for the problem and/or its adjoint in the variational theory framework, but recovery methods that rely on building a better solution to derive an *a posteriori* estimate are still very much in use in the engineering world. Among these methods

are the Richardson Extrapolation technique [5] already discussed in the previous section, and the so-called ZZ Super Convergence Patch Recovery Method [25, 26]. These methods are applicable to linear as well as non-linear problems [27, 28]. As explained in [7] Sect 4.7 p82, this technique may require that the grid resolve the smallest scale.

A third stream of work strongly related to *a posteriori* estimates [29, 30, 31, 32, 33, 34], concerns real complex phenomena that are non-linear, stochastic and multi-scale with no clear cut between the different scaling. Problems with turbulent flows, flow in porous media, or weather prediction are classical examples. The stochastic method of Glimm *et al.* [29] for the prediction of complex phenomena divides them into two components. One is the forward problem, starting from the governing equation and initial data; the other is the inverse problem for minimizing the uncertainties in the model from given observations of the system. The main thrust of the method is to predict functional output of the solution with a coarse grid solution based only on a probability error model that includes error in numerical computation, error in the observation or experiments used to calibrate the model, and error in the data used in the model. Randomness occurs at several levels, such as the specification of the model or the solution process itself. The solution process and the model must support a probabilistic framework; that is why stochastic PDEs are a natural application field. [34] gives a detailed description of the method applied to scale up flow in porous media. To focus on the role of this method for *a posteriori* estimates of the numerical error in direct computation of PDEs, the assumption is that the numerical error can be divided into two components: first, a highly variable component with

sensitive dependence on data, and second, a systematic component with smooth but also unknown dependence on the data. Both components are present in the random process that models the error. The randomness of the error process encompasses the sensitive dependence of the error on the data. Further, the output functional, the objective the method should predict, must not be sensitive to the global error. The error estimate depends on the specific choice of the error statistic model. It seems that this approach is very useful when uncertainties in the model are dominant versus the numerical error, and when a fine grid solution is never accessible. In complex numerical methods such as PDF/Monte Carlo for turbulent flows [35], the construction of the real model for numerical accuracy is a difficult task. It is remarkable that after completion, one may minimize the main component of the error that is random via time averaging. Then Richardson Extrapolation can provide significant improvement on the accuracy of the solution. Nevertheless, the theory developed by Glimm *et al.* [29] is a giant step toward the understanding of the effect of combined observation errors, model error, and numerical simulation error in effective prediction.

In the next section, we are looking briefly at some limitations of this methods.

1.4 Limitations of traditional approaches

The methods described in the previous section are an important axis of research in building and understanding *a posteriori* error estimates. Each method has its strength and weakness, and choosing one particular method results from having a good understanding of the problem at hand and the resources available.

In addition, numerical procedures might produce inadequate and inaccurate solutions or might not capture some detail for complex physical problems. This leads to the notion of under-resolved meshes where evaluating the numerical accuracy is still a challenge [3]. As a matter of fact, in complex modeling, as described in the ASCI project, best grid solutions provided by our best computing resources are fairly under-resolved at least locally [36]. *A posteriori* estimates should now be redesigned to provide solution verification assessment in this context [37, 38].

Furthermore, the analysis of errors is often incomplete because it often does not take into account some basic physical mechanisms on propagation:

- diffusion terms cause slow isotropic error decay, but global error pollution may occur from local irregularities;
- advection terms propagate local errors in the transport direction, but propagate errors to decay exponentially in the crosswind direction;
- reaction terms cause isotropic exponential error decay, but stiff behavior may occur in the coupling error components; and
- errors may occur in the coupling of the three components.

These same ideas lead to the development of efficient, parallel iterative domain decomposition solvers that optimize fast decay of numerical errors introduced at artificial interfaces.

Therefore, it might be useful to keep track of specific error propagation even for models in which diffusion, convection and reaction terms are present; it is impossible

to account for all error interactions by analytical means.

Finally, there are still possibilities of catastrophic failures of *a posteriori* estimates in case of meta-stabilities or bifurcations. A good method for complex modeling should therefore provide numerical indicators of ill-conditioned problems and potential catastrophic errors.

1.5 Objectives

The goal of this work is to present a versatile framework to perform *a posteriori* error estimates for complex PDEs. In order to be able to use this framework, we consider a code that provides a set of discrete approximations of PDEs.

The questions that this work will try to answer are:

1. Given that one can obtain the definition of the residual of the PDE approximation, the existence of a stability estimate on the approximation and two grid solutions, can we automatically find the order of convergence ?
2. Using two or three different grid solutions, not necessarily with uniformly increasing mesh resolution, can we obtain a solution with improved accuracy?
3. Can we derive reliable *a posteriori* error bounds from coarse grid approximations of complex PDE problems?

In addition, there are strong requirements for the solution we want to provide to answer those questions. First, it should be simple enough to implement or modify,

and it should be a stand-alone procedure developed independently from the simulation software. If it is possible to use the existing estimates, the developer could integrate this new error estimate procedure more tightly with the simulation tool. The second aspect driving the development of this *a posteriori* framework is that the arithmetic cost is negligible compared to a direct computation of a very fine grid solution. However, the solution should be general enough that software using variational formulation, finite volumes formulation or finite differences formulation, with irregular meshes or non-linearities, can employ it. Additionally, it should enhance the numerical accuracy and efficiency of simulation with complex physical models and trust in the context of code verification; and it should increase the overall numerical efficiency of the solution procedure when combined with a multilevel procedure.

1.6 Layout

This dissertation is organized in the following way.

Chapter 1 has given an introduction, including a literature review.

In Chapter 2, we present an overview of some flaws of Richardson Extrapolation in the context of fluid mechanics. Then, the focus of the chapter will be on stiff elliptic problems. It is a first generalization of the least square extrapolation method and the introductory work of this dissertation.

In Chapter 3, the focus shifts from elliptic partial differential equations to dependent problems and more specifically to parabolic problems. The introduction

of the time dependency in the framework yields new problems. Several simplified numerical models taken from the literature will illustrate some of the capabilities of the optimized extrapolation method.

In Chapter 4, we present a general framework to perform solution verification on simulation software even if the underlying set of discrete equations is unknown. A by-product of the framework is the evaluation of the conditioning of the problem in a reduced space at a reduced cost.

In Chapter 5, taking into consideration the main drawback of our method, we expose a software solution that takes advantage of a distributed network to perform solution verification. This chapter mainly focuses on implementation features and performance evaluations.

Finally, Chapter 6 provides the conclusion of the dissertation, and presents some possible improvements and investigations.

Chapter 2

Least square extrapolation method

In this chapter, after a brief reminder of some of the limitations of the Richardson Extrapolation method, we present the Least Square Extrapolation method for stiff elliptic problems as done in [39].

2.1 Limitations of Richardson Extrapolation method

As presented in the introduction, Richardson Extrapolation is a technique that was investigated in different communities to improve quantitative accuracy and the order of accuracy. The CFD community is an example of such approaches. Several papers cover the weaknesses and strengths of this method. The goal of this section is to present an overview of some of the weaknesses and to introduce a novel approach that is largely inspired by it.

The first set of limitations was described in [40]. Their analysis was done on Cartesian uniform meshes and thus does not represent a significant part of the simulation methods. Moreover, non-uniform or non-Cartesian grids might emphasize some of the limitations. Here is a summary of the findings :

1. Richardson Extrapolation is limited to flow with large gradients in properties, such as walls or polyphasic flows, for example. It can, in certain cases, worsen the accuracy of the extrapolated solution.

2. High-order extrapolation schemes retain the accuracy of the original solution at the cost of computational complexity.

3. The use of high order extrapolation schemes to retain the same accuracy restricts the elimination of spurious frequency in complex geometry.

The overall conclusion is that smoothness of the solution is a prerequisite even for the simplest flow computations.

Second, in [41], the authors address the problem of oscillatory convergence for finite difference methods. They also suggest some solutions, such as using alternative extrapolation schemes.

In [42], it is shown that to obtain higher order accuracy of the extrapolation, one has to modify the Richardson Extrapolation method.

Thus, assumptions of smoothness and that local error is related to global errors might not always be verified.

2.2 Optimized Extrapolation method for elliptic problems

In this section, we present the fundamentals of the Optimized Extrapolation method for stiff elliptic problems. This work was previously presented in [39].

2.2.1 Stiff elliptic problems

We consider the elliptic problem

$$div(\rho \nabla u(x)) = f(x),\, x \in \Omega \subset I\!R^2 \tag{2.1}$$

$$u = g \; on \; \partial\Omega, \tag{2.2}$$

with $\rho(x) > \epsilon, x \in \Omega,\ \epsilon > 0$. Ω is a polygonal domain.

We assume that the elliptic problem is well-posed and has a unique solution which is relatively smooth.

For the ease of presentation, we use, in this section only, a finite volume approximation of (2.1) on a uniform Cartesian mesh with square cells of size dx. The problem is solved with centered finite volume cells with a classical order scheme. The corresponding linear system is

$$A_h U_h = F_h. \tag{2.3}$$

Equations similar to (2.1) are not purely academic problems.

For example, we can first look at the following problems which exhibits large variations of ρ throughout the domain Ω:

1. The pressure solver in a multiphase flow problem with large ratio of density between fluids: for instance, bubbles of air rising in a liquid may have densities several orders of magnitude different than the density of the fluid itself. The same is true for drops of liquid falling in low-density gas [43, 44].

2. The pressure solver when simulating a flow in porous media through multi-layer materials if the permeability ratio between the layers is too large.

3. The heat equation when studying heterogeneous material having large discontinuity in the thermal conductivity between each media.

In these situations, we consider Ω to be partitioned into two subsets $\Omega_1 \bigcup \Omega_2$, such that

$$||\rho||_{2,\Omega_1} \sim 1, \tag{2.4}$$

$$||\rho||_{2,\Omega_2} \sim \tau, \tag{2.5}$$

with $\tau \ll 1$ or $\tau \gg 1$. The larger the ratio of τ to one is, the larger is the condition number of A, and the harder (2.3) is to solve efficiently with a good approximation of the exact solution.

The second type of problems we are interested in is when the source term $f(x)$ is a collection of dipoles. An example of such a problem arises in the pressure equation in the immersed boundary method of Peskin [45]. Peskin's method is a very elegant technique for the simulation of fluid-structure interaction and is widely used in biological situations, but it suffers from a lack of accuracy. We will now present the optimized extrapolation method in the context of an elliptic problem.

2.2.2 Optimized Extrapolation method

2.2.2.1 Overview of the method

Let G_1 and G_2 be two regular Cartesian meshes used to build two finite volume approximations of the elliptic problem (2.1). Let us denote h_1 and h_2 as the size of the square cells for both meshes, and U_1 and U_2 to be the two corresponding approximations of the continuous solution $u \in (E, ||\ ||)$. We assume the convergence of these approximations, that is

$$U_1,\ U_2 \rightarrow u \text{ in } (\mathrm{E}, ||\ ||) \text{ as } \mathrm{h}_1,\ \mathrm{h}_2 \rightarrow 0. \tag{2.6}$$

A consistent linear extrapolation formula should have the form

$$\alpha U_1 + (1-\alpha)U_2, \tag{2.7}$$

where α is a weight function. In classical RE the α function is a constant. In our optimized extrapolation method α is a space-dependent function. If U_1 and U_2 are in a finite element space $(E_h, ||\ ||)$, α must be such that the linear combination is still in $(E_h, ||\ ||)$.

We formulate the following optimization problem for the unknown function α:

(P_α)**:** *Find* $\alpha \in \Lambda(\Omega) \subset L_\infty$ *such that* $G(\alpha\ U_1 + (1-\alpha)\ U_2)$ *is minimum in* $(E_h, ||\ ||)$,

where G is an objective function to be defined. The Optimized Extrapolated Solution (OES) is then $V_e = \alpha\, U_1 + (1-\alpha)\, U_2$.

For computational efficiency, $\Lambda(\Omega)$ should be a finite vector space of very small dimension compared to the size of matrix A_h defined in (2.3). The objective function G can be any *a posteriori* existing error estimate, if it operates on the space of approximation $(E_h, ||\ ||)$.

In the most general situation, i.e., in the absence of the knowledge of any rigorous *a posteriori* estimate, we choose to minimize the consistency error for the numerical approximation of (2.1) on a fine mesh M^0 of step h^0. The fine mesh M^0 should be set such that it captures all the scales of the continuous solution with the accuracy required by the application. We have *a priori* $h^0 << h_1,\ h_2$. Let us emphasize that h_1 and h_2 do not have to be very different, and that the ratio$\frac{h_1}{h_2}$ does not have to be an integer. Both coarse grid solutions U_1 and U_2 must then be interpolated onto M^0. We will denote $\tilde{U}_1$ and $\tilde{U}_2$ as the corresponding grid functions. One needs seconds to get the approximation of the elliptic operator A_{h^0}. The objective function is then

$$G(U_h^0) = ||A_{h^0}\ U_{h^0} - F_{h^0}||, \tag{2.8}$$

where $U_h^0 = \alpha\ \tilde{U}_1\ +\ (1-\alpha)\ \tilde{U}_2$.

The choice of the space $(E_{h^0},\ ||.||)$ and its norm should depend on the property of the solution. In LSE [46, 47] we chose the discrete L_2 norm on M^0. We will investigate here other possibilities, such as the L_1 or the L_∞ norm.

One of the difficulties encountered with such a two-level extrapolation method is the so-called cancellation problem [46]. In practice, there exist subsets of Ω where U_1 and U_2 are much closer to each other than what the expected order of accuracy based on local error analysis should provide. In such areas the sensitivity of the

extrapolation to the variation of α is very weak and the problem is ill-posed. The optimization computation procedure should consider these subsets as outliers only. A potentially more robust optimization procedure consists of using three levels of grid solutions. The optimization problem is then written

$(P_{\alpha,\beta})$**:** *Find* $\alpha, \beta \in \Lambda(\Omega) \subset L_\infty$ *such that* $G(\alpha\, U_1 + \beta\, U_2 + (1-\alpha-\beta)\, U_3)$ *is minimum in* $(E_h, ||\ ||)$,

where G is an objective function to be defined. The optimized extrapolated solution is then $V_e = \alpha\, U_1 + \beta\, U_2 + (1-\alpha-\beta)\, U_3$.

We notice that if all $U_j,\ j = 1 \ldots 3$, coincide at the same space location there is either no local convergence or all solutions U_j are exact. In such a situation, one cannot expect improved local accuracy from any extrapolation technique. The robustness of the OES method should come from the fact that we do not suppose *a priori* any asymptotic formula on the convergence rate of the numerical method as opposed to RE.

Let us assume that the optimization problem P_α or $P_{\alpha,\beta}$ has been solved and that we have computed V_e either from the two-level or three-level method. We are going to discuss now its application to provide *a posteriori* error estimates.

2.2.2.2 *A posteriori* estimate

Let us denote U_j to be one of the coarse grid approximations at our disposal. A global *a posteriori* estimate of the error $(U_j - u)$ may come in two different ways.

- The first way is the recovery method based on the idea that the optimized extrapolated solution is more accurate than the coarse grid solution. Let us denote $\tilde{U}_j$ the coarse grid solution projected onto the fine grid M^0 via a suitable interpolation procedure. Let us assume that the extrapolated solution is decisively more accurate than that based on interpolation from the coarse grid solution, namely,

$$(V_e - u) << (\tilde{U}_j - u), \text{in } (\mathrm{E_h}, ||\ ||). \tag{2.9}$$

  Then $||V_e - \tilde{U}_2||$ is a good error indicator to assess the accuracy of G_2 solution. We have then

$$(\tilde{U}_j - V_e) \sim (\tilde{U}_j - u), \text{in } (\mathrm{E_h}, ||\cdot||). \tag{2.10}$$

  We will show in our experiments that this method may give a good *lower* bound error estimate. However, we do not know, in general, if hypothesis (2.9) is correct. If G is chosen to be the residual on the fine grid, there is no guarantee that a smaller residual for V_e than for U_2 on the fine grid M^0 will lead to a smaller error.

- The second way is a global *upper* bound that follows from a stability estimate with the discrete operator. Let us assume that the objective function is the residual in the discrete norm $||\cdot||$, namely (2.8). Let us denote U^0 to be the fine grid solution on M^0, and A_0 to be the corresponding linear operator. We have

$$||V_e - U^0|| \ < \ \mu\, G(V_e), \tag{2.11}$$

where $\mu \geq ||(A_0)^{-1}||$.

We conclude then

$$||\tilde{U}_2 - U^0|| \ < \ \mu \, G(V_e) + ||V_e - \tilde{U}_2||. \tag{2.12}$$

The procedure to derive an estimate for μ will be discussed later.

Equation (2.12) is a good global *a posteriori* error estimate provided that

$$||U^0 - u||_2 \ << \ ||U^0 - \tilde{U}_2||_2. \tag{2.13}$$

One way to test hypothesis (2.13) is to measure the sensitivity of the upper bound (2.12) with respect to the choice of the fine grid M^0. This is a feasible test because the fine grid solution is never computed in OES. Our verification procedure then checks that $||U^0 - U_2||_2$ increases toward an asymptotic limit as M^0 gets finer.

We will now present, in detail, the solution procedure to obtain OES and *a posteriori* error estimates. We will assume that G is a linear operator.

2.2.3 Procedure to construct the Optimized Extrapolation

Let $e_i, \ i = 1 \ldots m$ be a set of basis functions of $\Lambda(\Omega)$. The solution process of P_α and/or $P_{(\alpha,\beta)}$ can be broken down into three steps:

1. Interpolation of the coarse grid solution from $G_j, j = 1 \ldots p$ to M^0. We have two coarse grids to interpolate for P_α, respectively, three for $P_{(\alpha,\beta)}$.

2. Evaluation of the objective function

$$G[e_i\ (\tilde{U}_j - \tilde{U}_{j+1})],\ i = 1..m,\ j = 1..p-1,\ \ and\ \ G[\tilde{U}_p] \tag{2.14}$$

on the fine grid M^0.

3. The solution of the optimization problem that has m unknowns for each weight coefficient α and β used in the extrapolation procedure.

In practice, we should keep m much lower than the number of grid points on any coarse grid used. If one chooses the discrete L_2 norm, the optimization problem can be solved easily and the arithmetic complexity of the overall procedure should be of order $Card(M^0)$. In the general case, coding in a stand-alone program independent of the main numerical code might implement the algorithm.

Remark 1: *This procedure can be generalized to non-linear elliptic problems via a Newton-like loop [46, 47].*

We will now discuss the first step of the algorithm.

2.2.3.1 Projection on the fine grid and postprocessing

To compute the objective function properly, the interpolation procedure should preserve the properties of the numerical solution.

For conservation laws, one may require that the interpolation operator should satisfy the same conservation properties. In addition, the method can take into account other constraints related to physical realization. For example, for reacting

flow problems, one can require that the interpolating function preserve the positivity of species. One may use, for example, a transform of the unknown variable,

$$\Psi = \Phi(U), \tag{2.15}$$

such that the inverse map $\Phi^{-1}(\alpha\Psi_1 + (1-\alpha)\Psi_2)$ intrinsically satisfies the constraint.

For positivity, one may use a bijective map Φ from $\mathbb{R}^+$ to $\mathbb{R}$. For mass conservation, in [47] we used the standard stream function transform.

As discussed in [46], the interpolated solutions $\tilde{U}_i$ on the fine grid contain spurious high frequency components. Linear interpolation is much worse than spline interpolation from this point of view. This problem is amplified by the fact that the objective function usually requires the computation of the discrete derivatives of $\tilde{U}_j$.

These spurious frequency components of the interpolated solution are obviously carried on in all linear combinations V_e. The computation of the objective function might then be polluted to the point where minimizing $G(U_h)$ does not guarantee any longer that one minimizes the numerical error.

One postprocessing procedure to overcome this difficulty is to filter out the artificial high frequency components of U_j that cannot be present on the coarse grid G_j. However, for the elliptic problems (Equation 2.1) discretized by Equation 2.3 on M^0, it is convenient to postprocess the interpolated functions $\tilde{U}_j$, by a few steps of the artificial time stepping scheme

$$\frac{V^{n+1} - V^n}{\delta t} \;=\; A_0 V^{n+1} - F_{h_0},\; V^0 = \tilde{U}_j, \tag{2.16}$$

with appropriate artificial time step δt. This will readily smooth out the interpolating

function. We will discuss later a criterion to stop this smoothing relaxation (Equation 2.16).

We will also compare our numerical experiments on Equation 2.1 with the numerical results obtained with linear interpolation and spline interpolation. Let us now discuss the choice of the objective function in OES.

2.2.3.2 Choice of the objective function

In principle, OES should be much cheaper than the computation of the fine grid solution U^0 on M^0. The easiest solution is to choose the objective function to be the residual computed in L_2 norm. This choice presents two essential advantages.

The optimization problem to be solved is a least square problem that is well understood, easy to solve, and easy to process with existing software libraries [48].

For problems with discontinuous solutions, the choice of the L_2 norm might not be best. We can devise an entirely similar approach using, for example, the L_1 norm. That choice might be more relevant for problems having a discontinuous solution. Unfortunately, the minimization of the residual in the L_1 norm is a more difficult task because the objective function is non-differentiable. In this case, we have used the simplex search of Nelder and Mead [49] implemented in Matlab starting from the optimum solution obtained with the L_2 norm. The computation of an estimate of the L_1 norm $||A_0^{-1}||_1$ might be done with an iterative procedure as well. We refer to the papers of Higham [50, 51] that present the method. A well-known procedure that implements this algorithm is available in LINPACK. We have chosen for our

numerical experiment to apply RE to the sequence of estimates $||A_{h_j}^{-1}||_1,\ j = 1, 2, 3,$ to get an estimate of μ. This computation is less expensive than to compute $||A_0^{-1}||_1$ directly. For problems where one is interested in the maximum of the error, we can devise a similar OES using the L_∞ norm.

Finally, one can use standard finite element *a posteriori* estimates [52, 53] in place of the norm of the residual to define G. We will not discuss this approach in this section, since we are working with finite volume discretization for which, to our knowledge, such rigorous estimates for (2.1) are not available.

Remark 2: *We emphasize that all fine scales that are not present in the two or three coarse grid solutions will not be computed properly by the OES V_e. OES tries at best to recover all the scales that are present in the provided coarse grid solutions.*

Remark 3: *In multiscale problems, the numerical error is often dominant in some small local area of the domain where the solution is stiff. One example is a boundary layer or a singularity at some corner of the domain. One criterion to choose the objective function G is to be able to capture such singularities. We refer to the literature in singular perturbation theory to give an extensive review of the choice of the norms to analyze boundary layer problems and its numerical solution - see, for example, [54, 55, 56].*

Let us now discuss the representation of the weight function of $\Lambda(\Omega)$.

2.2.3.3 Representation of the weight functions

We look for a compact representation of the weight function that can capture the main features of the convergence order of the method with very few coefficients. Let us assume for the time being that Ω is a square domain.

As presented in [46], one can use a trigonometric expansion of the weight functions α and β that is adapted to the square domain. The set of trigonometric functions of Equation 2.17 define the space of the unknown weight function:

$$\alpha(x) = \sum_{i=1..m,\ j=1..m} \alpha_{i,j}\ e^i(x_1)\ e^j(x_2), \tag{2.17}$$

with $x = (x_1, x_2)$, $e^0 = 1$, $e^1 = cos(\pi x_{1/2})$ and $e^i = sin((i-2)\pi x_{1/2})$, for i = 3..m.

This set of trigonometric functions allows us to approximate at second order in L^2 norm any smooth non-periodic functions of $C^1[(0,1)^2]$, [57]. One additional advantage of this choice of approximation space for the weight function is that it allows us to interpret easily our numerical result in the frequency space.

We observe in practice that the higher order the expansion defined by Equation 2.17, the more amplified are the spurious modes in the interpolated solution $\tilde{U}_i$. Postprocessing is then particularly important to stabilize the OES.

In this section, we have tested a second alternative that might be better suited to capture the local properties of the convergence rate. First, let us define $e_{i,j}$ to be the set of Q_1 basis functions of the square domain Ω on the very coarse grid of $m \times m$ cells. $e_{i,j}$ is then one at the center of the cell of coordinates (i, j) and zero elsewhere. Second, we transform $e_{i,j}$ into $\tilde{e}_{i,j}$, which is the interpolated function defined on the

fine mesh M^0. Our second solution is then to look for the weight functions as follows:

$$\alpha(x) = \sum_{i=1..m,\ j=1..m} \alpha_{i,j}\ \tilde{e}_{i,j}(x_1, x_2). \tag{2.18}$$

We will compare both representations of the weight coefficients in the numerical experiments.

In the general case, Ω is a polygonal domain that can be embedded, after appropriate rescaling, into a square $(0,1)^2$. Because the unknown weight functions of the OES do not have boundary conditions, we can use exactly the same set of basis functions. However, the OES problem depends only on the part of the interpolated solutions that are the grid points contained inside the domain Ω. With the representation given in (2.18), the OES formulation should not contain any coefficients of the basis function $\tilde{e}_{i,j}$ that are identically null inside Ω.

We have now described all the components of the solution procedure to build the OES.

We are going to show in the next section that the OES method provides robust *a posteriori* global error estimates for the elliptic problem (2.1) with stiff coefficient ρ.

2.3 Results

In this section, we present numerical results for the OES method on stiff elliptic problems.

We are going to consider eight test cases based on the homogeneous Dirichlet problem

$$div(\rho \nabla u(x)) = f(x),\, x \in \Omega \subset I\!R^2 \; and \tag{2.19}$$

$$u = 0 \text{ on } \partial\Omega, \tag{2.20}$$

with the following smooth right-hand side function:

$$f(x) = \exp\left(-2\,(x_1(i) - 0.5)^2 \;-\; 2\,(x_2(j) - 0.5)^2\right). \tag{2.21}$$

The first test case, T_1, used as a basic reference is the Poisson problem into the square $(0,1)^2$. All other test cases correspond to different distributions of the ρ function and/or different geometry of the domain.

Let us define D_1 (respectively, D_2) the disc of center $C = (0.38, 0.48)$ ($C = (0.64, 0.74)$) and radius $R = 0.15$. The coefficient ρ is as follows:

$$\rho(x,y) = 1 \;+\; 0.5\,(\tau - 1)\;(1. + \tanh(-100\,(dist((x,y),C) - R)))\,,\; x \in \Omega, \tag{2.22}$$

where $dist((x,y),C)$ is the distance from the point of coordinates (x,y) to the center C of the disc. ρ is close to τ inside the disc of center C and radius R, and 1 outside.

For the first five test cases $T_i,\; i = 1 \ldots 5$, Ω is the unit square. The test cases T_6 and T_7 correspond to an L-shape domain $(0,1)^2 \setminus (0,0.5)^2$. The last test case, T_8, is for a Poisson problem with a circle of dipole source terms. This last example plays the role of a reference test case in the analysis of the impact of the choice of the norm in the error estimate. Figures 2.1, 2.2, 2.3, and 2.4 show, respectively, the solution on the fine grid M^0 for the test cases T_i, where $i = 2,\; 6,\; 7,\; and\; 8$.

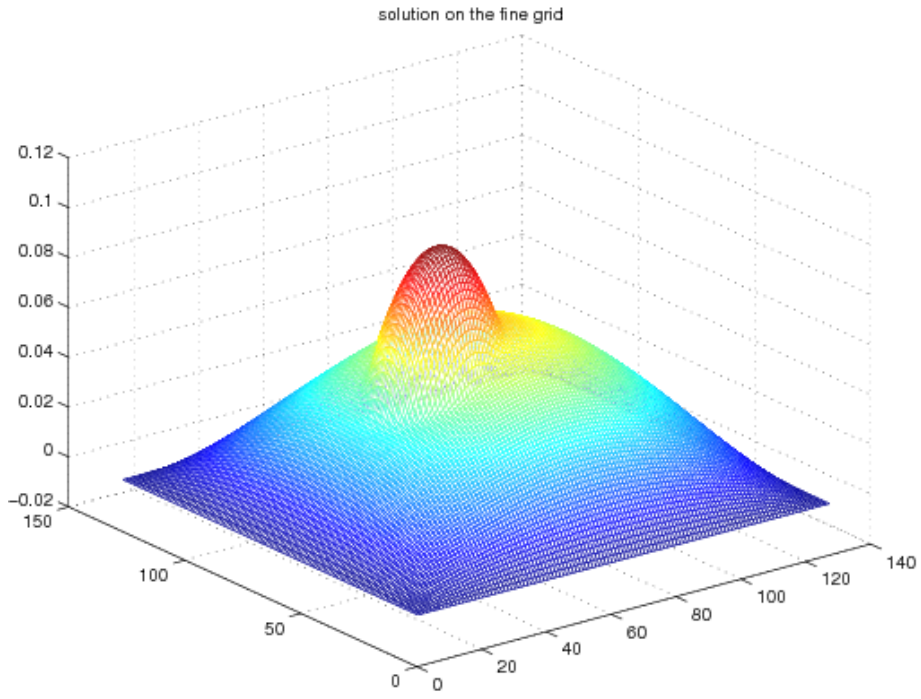

Figure 2.1: Solution of T_2 with $\rho \approx 0.1$ in the disc D_1.

The test cases $T_i,\ i = 1 \ldots 7$, are designed to be representative of the pressure equation for a two-phase flow problem. The disc $D_{1/2}$ is the analog of a bubble of circular shape that has the relative density τ to its medium. Small τ are for bubbles of gas immersed in liquid. Large τ can be interpreted as a liquid drop immersed in air.

We see in Figure 2.1 that small τ gives a high-pressure peak in the disc. Large τ in Figures 2.2 and 2.3, induces a plateau in the pressure that matches the disc contours. In test case T_6 the disc D_1 intersects the wall, and the plateau matches the zero boundary condition. In test case T_7 the disc D_2 stays inside the L shape domain, and we have a strong interaction between the "bubble" and the singularity

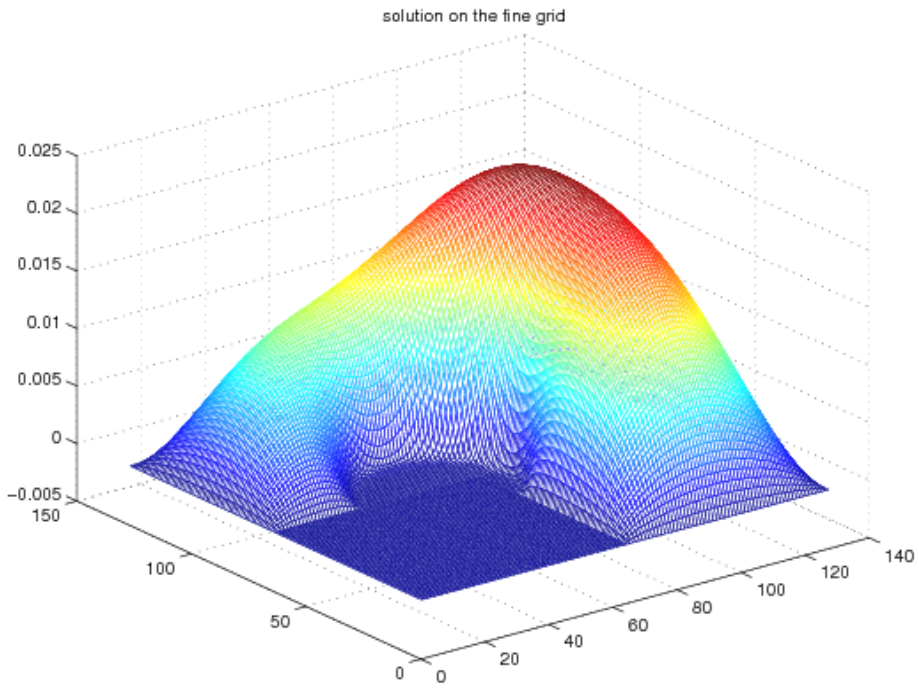

Figure 2.2: Solution of T_6 with $\rho \approx 100$ in the disc D_1.

of the solution at the entry corner.

With homogeneous Dirichlet boundary conditions, complex geometry does not play a significant role for small τ values. In fact, it can be observed that for small τ the solution of the elliptic problem is of order τ outside the disc D.

We have done a large number of experiments with various grid resolutions for $G_i,\ i = 1 \ldots 3$ as well as M^0. We focus this experimental section on the discussion of the following:

- the postprocessing of the interpolated solution, i.e., the number of time steps in the relaxation procedure 2.16;

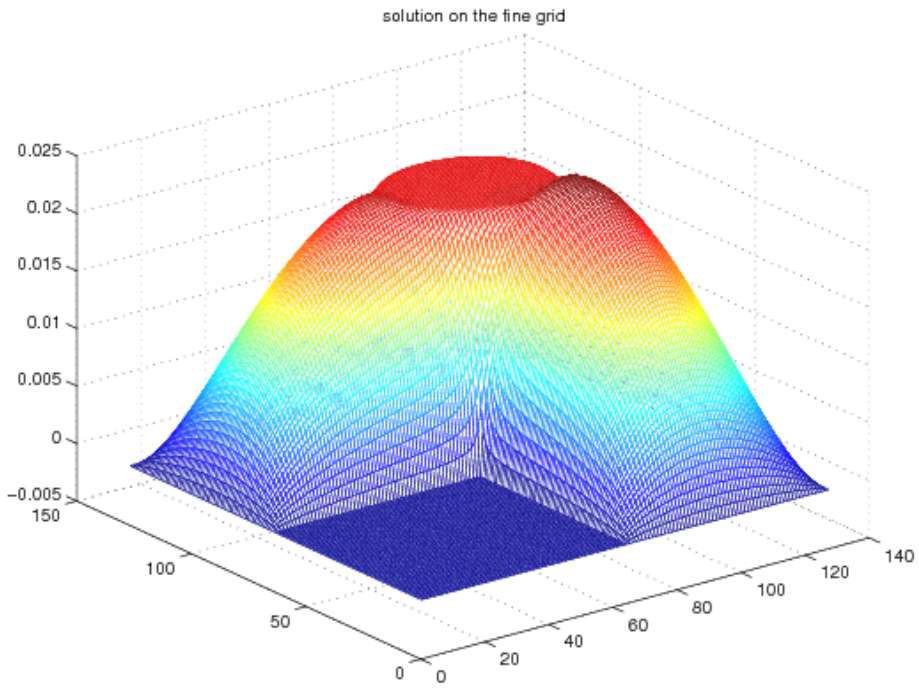

Figure 2.3: Solution of T_7 with $\rho \approx 100$ in the disc D_2.

- the impact of the parameter τ;
- the choice of the interpolating function: linear versus spline;
- the choice of the basis function for $\Lambda(\Omega)$, i.e., Fourier versus interpolated $Q1$; and
- the choice of the norm L_2 versus L_1 in the objective function

$$G(U_h) = ||A_h\, U_h - F_h||. \tag{2.23}$$

The pseudo-language chosen to develop these concepts is Matlab. In all the graphics representing the performance of our method for these eight test cases, we

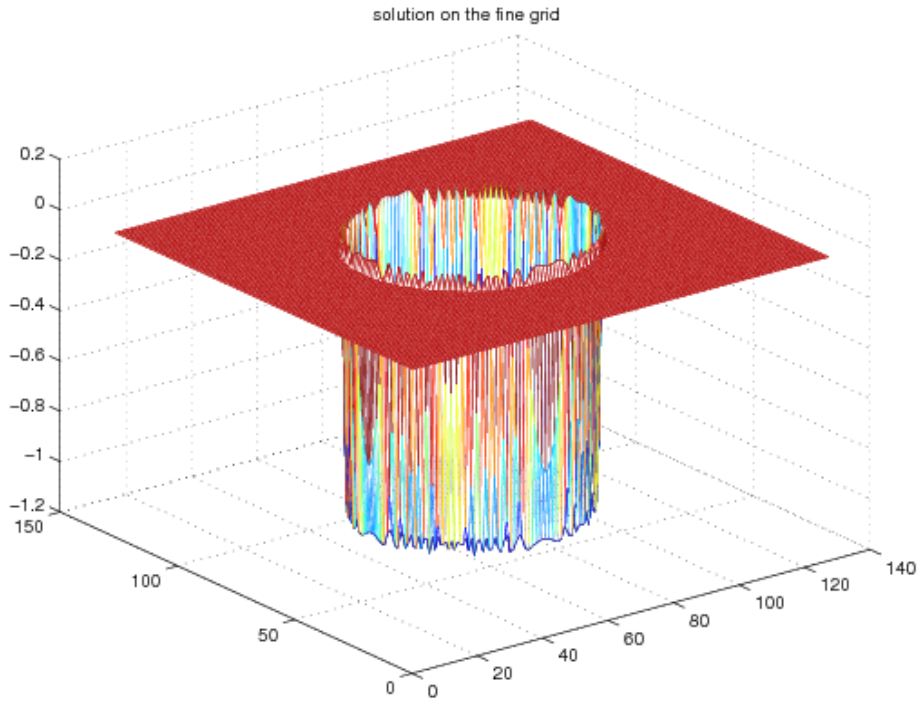

Figure 2.4: Solution of the Poisson problem with a circle of dipole source terms.

have chosen for convenience and comparison purposes to fix the parameters of the method as follows.

- The direct solver solves each test case on a Cartesian grid of the same space step h_i in both space directions. For the coarse grid G_1, G_2, and G_3, we have $h_1 = 1/14$, $h_2 = 1/20$, and $h_3 = 1/26$, respectively.

  This coarse grid does not resolve the sharp transition of the density function ρ defined in (2.22). ρ appears to be almost a step function on these coarse grids.

- The fine grid solution M^0 is chosen to be a grid of step 1/128. To verify the quality of this fine grid solution we use as a benchmark solution the grid M^∞

of space step 1/256.

- The bilinear interpolation or the cubic spline interpolation of Matlab interpolates the solution on M^0.

- The time integration of the heat equation enables the postprocessing of these interpolated solutions:

$$\frac{\partial u}{\partial t} = div(\rho \nabla u(x)) - f(x),\ x \in \Omega \subset \mathbb{R}^2,\ u = 0 \text{ on } \partial\Omega, \tag{2.24}$$

  on the interval of time length of order 10^{-2}. To this end, we use time steps of order $\delta t = 10^{-3}$ in a first order implicit Euler scheme. Typically, each time step requires two iterates of a biconjugate gradient method (BICGSTAB) with an incomplete LU pre-conditioner.

  The fine grid solution M^0 is not resolved by this scheme by all means. We will see that the quality of the OES improves dramatically with this relaxation procedure.

In Figures 2.5,2.6, and 2.7, we use the following conventions. The left graph gives an error estimate based on the recovery method Equation (2.9), while the right graph gives an upper bound of the error based on the global estimate Equation (2.12). The graph in the middle shows the residual obtained in the norm of choice for the OES method.

To check the accuracy of the error prediction, we have computed the M^0 fine grid solution. The curves labeled with '* 'give the error $||\tilde{U}_3 - U^0||$.

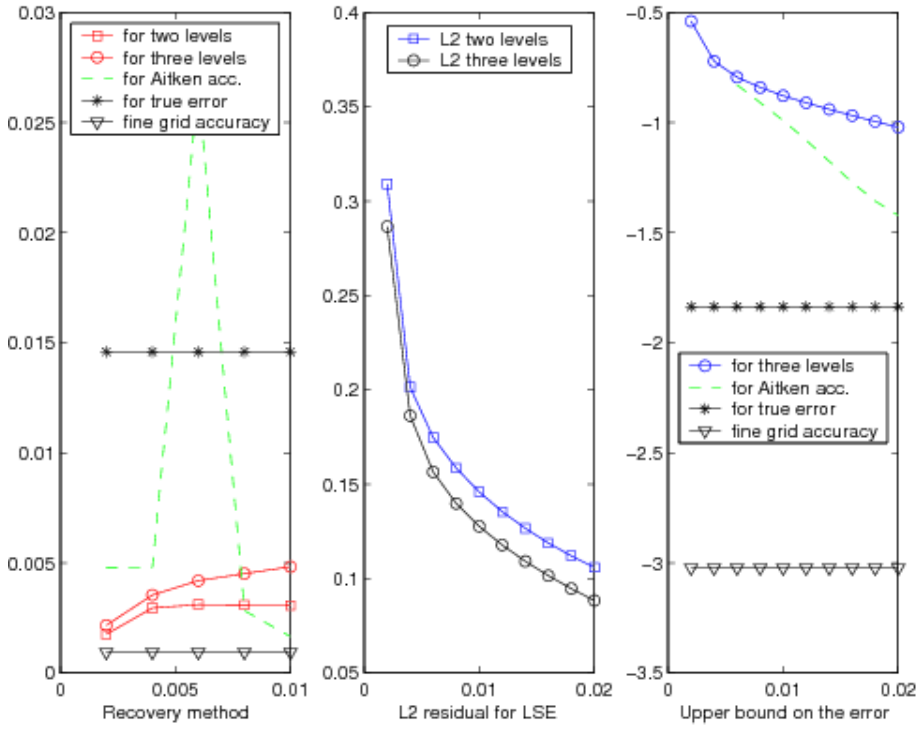

Figure 2.5: Error estimates with T_3. $\tau = 0.01$.

To check the accuracy of the M^0 solution, the curve with 'v' labels gives the error of the M^0 solution versus the ground true solution M^∞, that is $||U^0 - U^\infty||$ on M^0.

We will systematically compare the OES with the two-level method using the coarse grid solutions U_2 and U_3, and with the three-level method based on all three coarse grid solutions. The influence of the time stepping used to postprocess all three projected solutions $\tilde{U}_j$, $j = 1..3$, is demonstrated by representing on the horizontal axis of each graph the time variable of the postprocessing Equation (2.24).

The dashed curved '- -' in the left and right graphs are the Aitken accelerations of the sequences of error prediction versus the time step based on the three-level

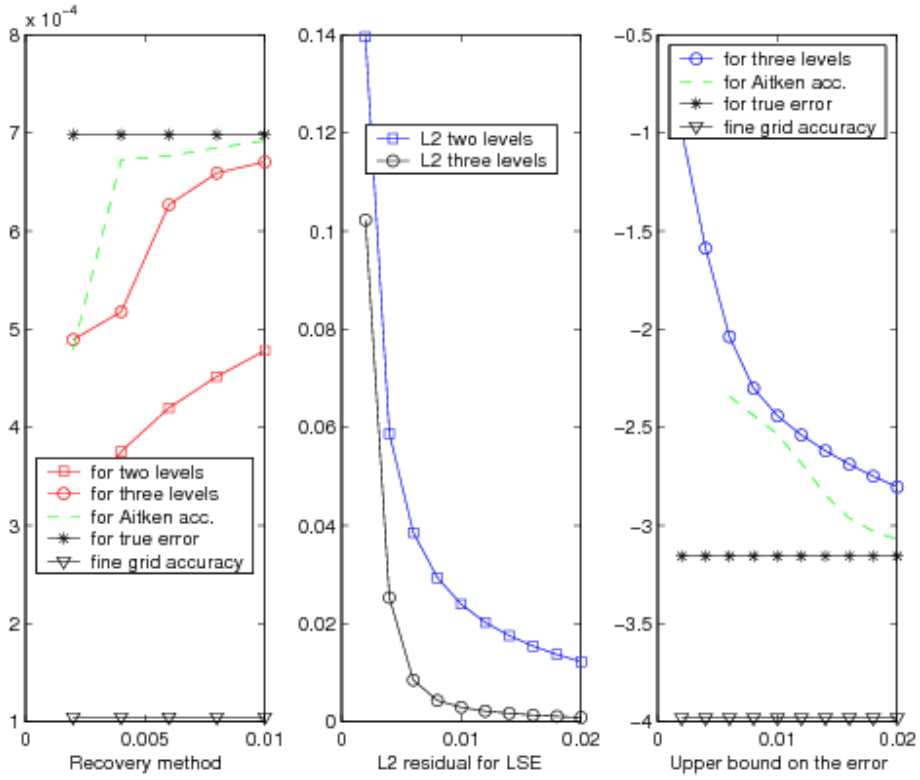

Figure 2.6: Error estimates with T_6. $\tau = 100$.

method. By construction, this Aitken acceleration improves the convergence of the numerical approximation *only* if the sequence of error prediction has a linear rate of convergence. Oscillations of the sequence of numbers generated by the Aitken process reflect either the lack of linear convergence, or the point at which the accelerated sequence is close to convergence within computer arithmetic accuracy.

We will now analyze the results that we obtained in our numerical experiments. Let us report first on the results with the LSE method for all the test cases.

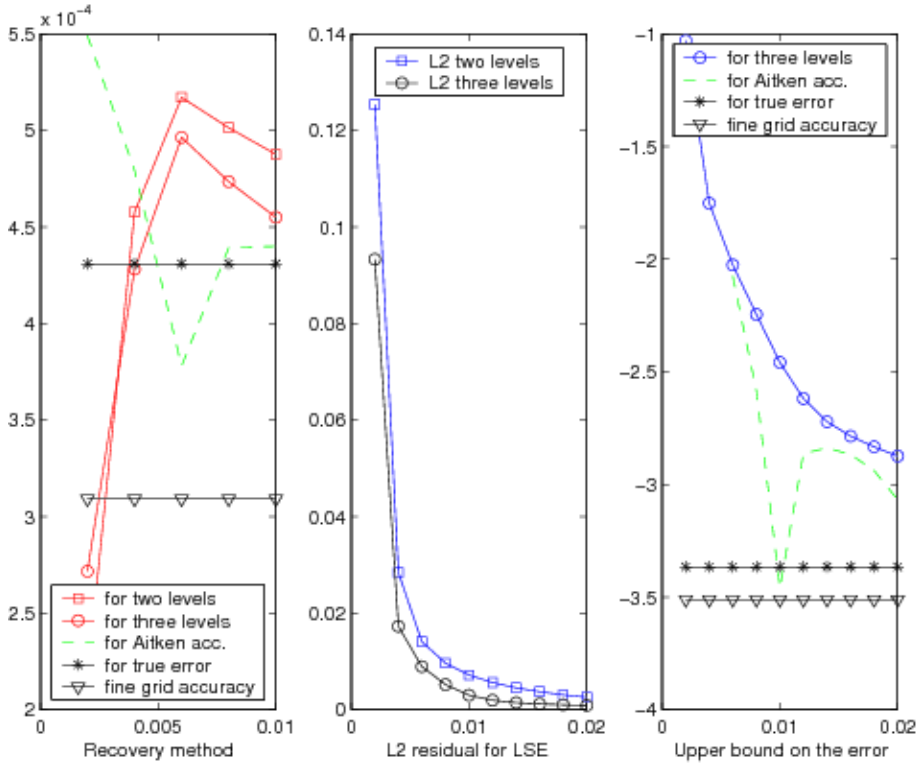

Figure 2.7: Error estimates with T_7, $\tau = 100$.

2.3.1 Energy Norm

All estimates here used the discrete L_2 norm and the trigonometric expansion Equation (2.17) of α and β. Let us discuss the influence of the relaxation procedure Equation (2.16).

2.3.1.1 Discussion on the postprocessing method

Both the recovery method and upper bound estimate provide accurate estimates for the Poisson problem in a square. We should expect this minimum from a new method,

since a basic second order Richardson Extrapolation already gives satisfactory results. One notices, however, the prediction step reaches good accuracy after four time steps, and that the spline interpolating function gives slightly better results than the bilinear interpolating function. Further, for this Poisson problem, there is no advantage to using a three-level extrapolation method versus a two-level method. For the Laplace operator, one can easily derive *a priori* how many time steps are required to damp the artificial frequency components of the interpolated solution that were not present in the coarse grid solution.

Let us write algebraically

$$(Id - \delta t\ A_0)\ \tilde{U}^{n+1} = \tilde{U}^n - \delta t\ F_h^0, \tag{2.25}$$

to be the time stepping Equation (2.16) applied to each interpolated coarse grid solution $\tilde{U}_j, j = 1, 2, 3$. A_0 is symmetric definite negative and we can order its eigenvalues λ_k as follows

$$0 > \lambda_1 > \lambda_2 > \ldots > \lambda_m. \tag{2.26}$$

$m = (N_0)^2$ is the number of grid points on the fine Cartesian grid M^0. Let us denote $v_k,\ k = 1 \ldots N$ as the corresponding orthonormal eigenvectors of A_0.

The matrix $B = (Id - \delta t\ A_0)^{-1}$ has for eigenvalues μ_k with $\mu_k = (1 - \delta t \lambda_k)^{-1}$. The eigenvalues of B are ordered as follows $0 < \mu_m < \mu_{m-1} < \ldots < \mu_1 < 1$.

Let us express $\tilde{U}_j^n$ in the orthonormal eigenvector basis v_k, with

$$\tilde{U}_j^n\ =\ \sum_{k=1..m} \tilde{U}_{j,k}^n\ v_k. \tag{2.27}$$

The following high frequency component of the solution

$$\sum_{k=m_j+1..m} \tilde{U}^n_{j,k}\, v_k \tag{2.28}$$

is an artifact of the interpolation procedure. It leads to the following error term that is part of the residual R^{n+1}:

$$\sum_{k=m_j+1..m} \lambda_k\, \tilde{U}^n_{j,k}\, v_k. \tag{2.29}$$

Let us assume that we expect *a priori* an error on $\tilde{U}_j,\ j = 1..3$, much larger than the tolerance number *tol* in the L_2 norm.

The high frequency component of the residual due to the interpolation process will decay as

$$\sum_{k=m_j+1..N} \mu^q_k \lambda_k\, \tilde{U}^n_{j,k}\, v_k \tag{2.30}$$

after q time steps. q should be chosen such that

$$\left(\sum_{k=m_j+1..N} (\mu^q_k \lambda_k)^2 \right)^{\frac{1}{2}} \ < \ tol. \tag{2.31}$$

A practical bound on q can be derived from Equation (2.31) and the analytical formula of the eigenvalue of the discrete Laplacian operator. The same analysis is not straightforward for stiff problems with an arbitrary ρ function in Equation (2.19). As a matter of fact, the operator A_o is still negative definite but no longer symmetric.

We use then the following heuristic argument. We know that for q large enough, the decay of the numerical error $E^n = \tilde{U}^n - U_o$ satisfies the asymptotic estimate

$$E^n \sim C^t \mu_1^q. \tag{2.32}$$

Similarly, the residual satisfies

$$R^n \sim C^t \lambda_1 \mu_1^q. \tag{2.33}$$

Once the spurious high frequency components of the error due to the interpolation process have been damped enough, the error *and* the residual decay at the same linear rate μ_1. Further, the larger the number of time steps, the more accurate is this convergence rate. This is the essence of the power method [58] to compute μ_1. We look then for a stop criterion that estimates how close the sequence is to its asymptotic rate of convergence.

Our criterion to stop the iteration is to compute the discrete second order derivative in time of $\log_{10} ||R^n||_2$; that is,

$$R_{tt}^n = \frac{\log_{10}(||R^{n+1}||_2) - 2\log_{10}(||R^n||_2) + \log_{10}(||R^{n-1}||_2)}{(\delta t)^2}, \tag{2.34}$$

and ensure that this number is below some *a priori* tolerance value.

We are going to show that our heuristic stop convergence criterion is also consistent with the use of the Aitken acceleration on our sequence of upper error bounds

$$\mu \, ||A_0 V_e^n - F||_2 + ||V_e - \tilde{U}_2||_2, \tag{2.35}$$

where V_e is the LES based on $\tilde{U}_j^n, \; j = 1 \ldots 3$.

Let us denote r_n the sequence of numbers

$$r_n = \mu \, ||A_0 V_e^n - F||_2. \tag{2.36}$$

The sequence r_n has also a linear rate of convergence at the speed μ_1.

The Aitken acceleration procedure that we apply to our upper-bound estimate Equation (2.35) writes

$$s_n = ||V_e - \tilde{U}_2||_2 + \frac{r_n \, r_{n+2} \; - \; r_{n+1}^2}{r_{n+2} \; - \; 2 \, r_{n+1} \; + \; r_n}. \tag{2.37}$$

Let us rewrite r_n as follows

$$r_{n+1} - r_\infty = (\mu_1 + \delta_n) \, (r_n - r_\infty). \tag{2.38}$$

We have $\delta_n \to 0$, as $n \to \infty$. We get

$$s_n - s_\infty = (r_n - r_\infty) \frac{(\mu_1 + \delta_{n+1}) \, (\mu_1 + \delta_n) \; - \; (\mu_1 + \delta_n)^2}{(\mu_1 + \delta_{n+1}) \, (\mu_1 + \delta_n) \; - \; 2 \, (\mu_1 + \delta_n)^2 \; + 1}. \tag{2.39}$$

We have then

$$s_n - s_\infty \sim (r_n - r_\infty) \frac{\mu_1}{(\mu_1 - 1)^2} (\delta_{n+1} - \delta_n). \tag{2.40}$$

Since

$$||R^n||_2 \approx r_n \tag{2.41}$$

therefore

$$R^{n+1} - R^\infty \approx (\mu_1 + \delta_n) \, (R^n - R^\infty). \tag{2.42}$$

Using this estimate in Equation (2.42) and in Equation (2.34), we have

$$R_{tt}^n \approx (\delta_{n+1} - \delta_n). \tag{2.43}$$

To ensure that R_{tt}^n is small is then a good stop criterion for the Aitken acceleration because of Equation (2.40). This Aitken acceleration will systematically be applied to enhance the upper bounds Equation (2.35).

We are now going to study the impact of the τ scale on the results obtained with the LSE method.

2.3.1.2 Impact of τ on the LSE result

In Figures 2.5, 2.6, and 2.7, we presented the results with spline interpolation using $m = 4$ in the trigonometric expansion of α, β. Let us consider the test cases T_2 to T_7. In all these test cases, we checked that the second order RE improves the solution accuracy, but not consistently. The corresponding error estimate based on the recovery method is thus unreliable. Because of the effect of the discrete operator on the interpolation, the upper bound error based on the computation of the residual is far too crude.

The LSE method always gives better results than RE with very few time steps to postprocess the coarse grid solution on the fine grid. For these stiff problems, the three-level method gives consistently better results than the two-level method.

In most cases, the recovery method gives an acceptable estimate from below of the error after very few time steps, as shown in Figures 2.5 and 2.6. The test case T_7,

in Figure 2.7, is an exception. It can be checked and seen in this test case that the code is not converging well, because $||U^0 - U^\infty||$ is not much smaller than $||U_3 - U^0||$.

The Aitken acceleration of the sequence of estimates based on the recovery method may sharpen this estimate, but not consistently. There is no obvious reason for which $||V_e - \tilde{U}_3||$ should converge linearly.

In all these test cases, the existence of oscillations might easily detect the failures of the Aitken acceleration.

In all test cases, the upper bound overestimates the error by a factor of five to ten at most, provided that we process the time integration of the solution with few time steps for an interval of time of size $2\ 10^{-2}$.

The Aitken acceleration of this time sequence of estimate provides a faster improvement of the upper bound for large τ than for small τ. This is consistent with the fact that the time stepping converges faster to the linear rate of convergence for large τ than for small τ. In the test case T_7 we have a strong oscillation of the Aitken acceleration of the upper bounds sequence. We conjecture that this is a good indicator of the bad convergence properties of the code due to the re-entry corner in this specific situation.

We found in all test cases the LSE method is robust if the relaxation scheme of (Equation 2.16) postprocesses the interpolated coarse grid solutions.

Let us now discuss the advantage of spline interpolation versus bilinear interpolation.

2.3.1.3 On the choice of the interpolating function

In principle, spline interpolation should preserve the smoothness of the solution and should give better results than linear interpolation with LSE. On the contrary, if the solution is very stiff, the spline interpolating functions smooth out the interpolated solution where it should exhibit a sharp front. The result should then be worse than linear interpolation with LSE. This is exactly what we have observed for the numerical error after very few time steps. However, thanks to the relaxation process the difference between both solutions after five time steps is marginal. In this case, we would prefer bilinear interpolation that is easier to implement in more geometry that is complicated. Let us discuss now the choice of the representation of the weight function.

2.3.1.4 On the choice of the basis function to represent the weight function

We have also tested the impact of the choice of the basis function to represent the unknown weight coefficients α and β. The general observation is that the accuracy of the LES prediction increases when one increases the value of m from one to a few units, typically four. The gain obtained in further increasing m becomes marginal in the Fourier case (Equation 2.17). On the contrary, our numerical simulation with m up to 12 shows slightly better convergence using Equation (2.18). However, the postprocessing procedure may make this improvement marginal after a few time steps. We speculate that the main contribution of the error with stiff problems is

so dominant in some local area that even the coarser grid solution is good enough outside the region of stiffness. In other words, the weight coefficient has very little influence on the quality of the LSE outside the region of stiffness. The least square extrapolation method can then capture the main component of the error with local or non-local basis functions as well. When the computation provides the coarse grid solutions on locally refined meshes, the hypothesis does not hold and needs revision. Let us further notice that the representation in Equation (2.18) gives fewer unknowns to compute for complex shape domains than Equation (2.17). We will now compare the LSE method with OES using the L_1 norm and the L_∞ norm.

2.3.2 Discussion on the choice of the norm

In Figure 2.8, we compare the error estimate based on the recovery method obtained with the L_1, (curved with dashed lines) and the L_2 norm (curved with continuous line), for the test case T_5. Curves labeled with $\Box$ and o correspond to the two-level methods and three-level methods. Except in the test case T_5 reported here, we did not find any significant advantages to using the L_1 norm instead of the L_2 norm for the objective function. The recovery method gives similar results for all other test cases with the L_1 and the L_2 norm.

In particular our conclusion on the impact of τ, the choice of the interpolating function and the basis function to represent the weight function is identical to our previous conclusion with LSE.

Further, the simplex search of the Nelder and Mead minimization procedure is

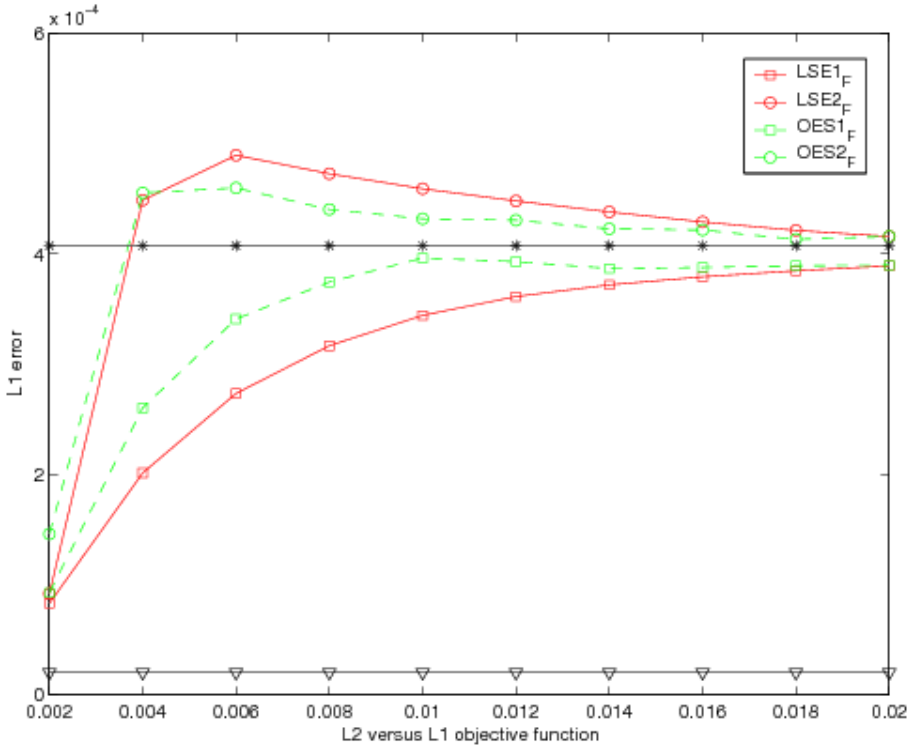

Figure 2.8: Error estimates in L_1 norm with T_5, i.e $\tau = 100$, based on linear interpolation and the recovery method.

obviously much more time-consuming than the least square method and is generally less accurate at convergence. The Aitken acceleration of the upper bound estimate is, therefore, not very effective. Figure 2.9 gives a representative example of our result. While a more efficient optimization procedure to construct OES might be used, we have also noticed that the upper bound obtained with the L_1 norm is much coarser than in the L_2 norm case for all test cases. This is a major drawback of the method.

There is, however, a significant interest in using the L_1 norm for solutions of

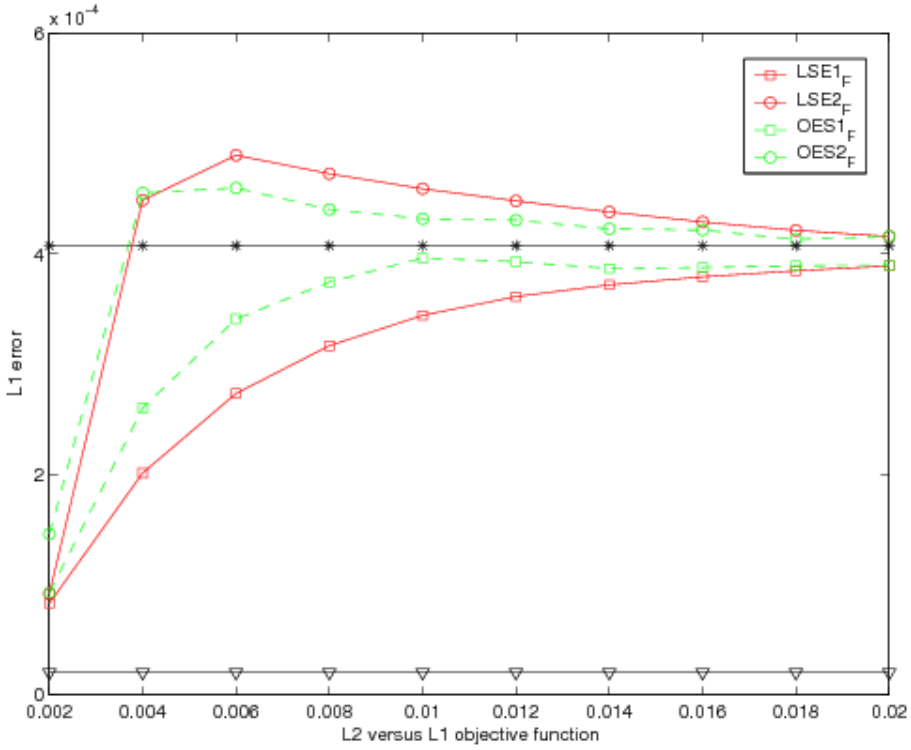

Figure 2.9: Upper error bounds in L_1 norm with T_4, i.e $\tau = 10$, based on linear interpolation.

PDE problems exhibiting discontinuities. Let us consider then the following test case, denoted T_8:

$$\Delta u = div(\vec{a}\ \delta^{\Gamma}(x,y),\ (x,y) \in (-1,1)^2,\ u_{|\partial\Omega} = 0, \tag{2.44}$$

where Γ is a circle of center 0 and radius $\frac{1}{2}$.

This test case is designed to represent the pressure equation in the Peskin method [45] when the membrane is the circle Γ. The force is distributed along the membrane with a set of discrete Dirac delta functions. These force terms have the direction of the radius of the circle. We choose then $\vec{a}$ to be the vector of components $(2\pi x, 2\pi y)$.

These force terms lead to a distribution of dipoles in the pressure equation. Let δ_h be the discrete approximation of the Dirac delta functions based on the piecewise cubic function given in [59] with a support of radius two space step h. The discrete representation of Γ uses M points. To ensure that the space steps between these grid points are of order h, we take $M = 6N$ with $N = 2/h$. The source term in Equation 2.44 is then

$$\delta^{\Gamma}(x,y) = \frac{1}{M}\sum_{i=1}^{i=M} \delta_h\left(x - 0.5\cos\left(\frac{2(i-1)\pi}{M}\right)\right)\ \delta_h\left(y - 0.5\sin\left(\frac{2(i-1)\pi}{M}\right)\right). \tag{2.45}$$

The solution of Equation 2.44 on the fine grid is given in Figure 2.4. One can notice the severe oscillation of the solution at the Γ location. Equation 2.44 shows our results using, successively, the L_2, L_1 and L_∞ norms and linear interpolation for the coarse grid projection on M^0.

Spline interpolation gives less accurate results than expected from the discontinuity of the solution.

One can notice, in Figure 2.10, that there is almost no advantage to using the three-level solution instead of the two-level solution. We conjecture that $\tilde{U}_2 - \tilde{U}_3$ and $\tilde{U}_1 - \tilde{U}_3$ have similar random behavior in the vicinity of Γ. There is no advantage to using $\tilde{U}_1$ in the three-level method.

The two upper bounds of the errors in L_2 and L_∞ norm are accurate after five time steps. The L_1 error estimate is more accurate than in the previous test cases. Although the error in the L_∞ norm is very large, the L_∞ error estimate gives a good prediction of the error that comes from the inaccuracy in the interface location.

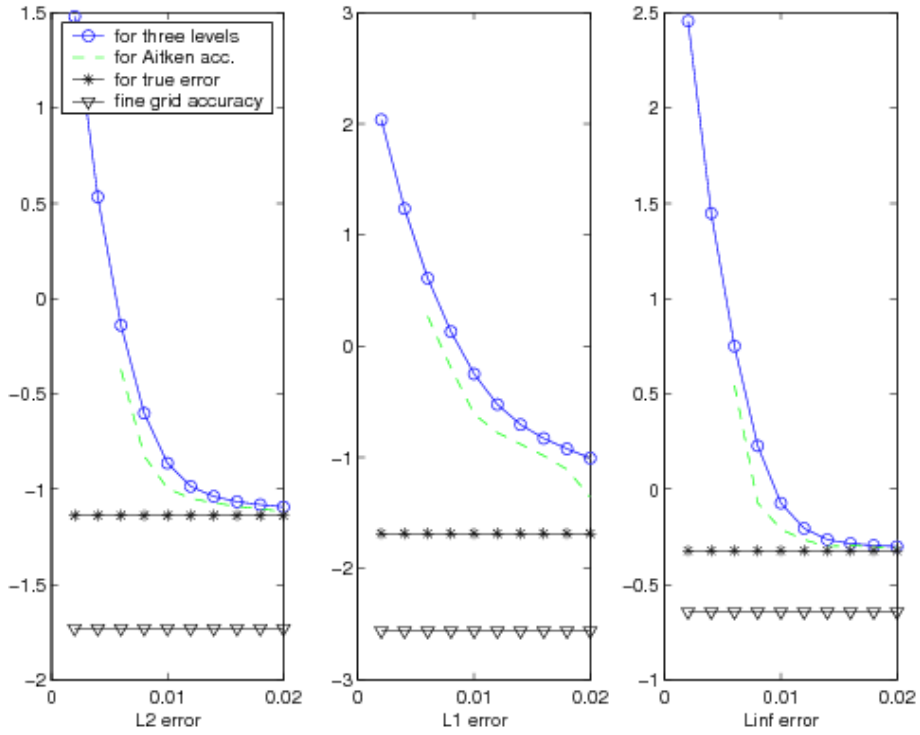

Figure 2.10: Upper bound on the error with L_2, L_1, and L_∞ norm with T_8 based on linear interpolation.

There is, therefore, no real advantage to using the L_1 norm versus the L_2 norm. Further, the L_∞ norm gives interesting complementary information to the L_2 norm estimate, on how inaccurate the solution can be near the circle of discontinuity. We present in the next section our general conclusion of this study.

In this chapter, we have shown how to extend the least square extrapolation method to a general optimization framework that allows one to use arbitrary norms and/or objective functions.

We concentrate our work on giving *a posteriori* estimates. We have presented

a rigorous upper bound error estimate technique to predict very fine grid solutions. We have used an acceleration technique to sharpen this error estimate and/or detect failures of convergence. We have applied the method to a multi-scale elliptic problem and a Poisson problem with a singular source term to demonstrate the robustness of the optimized extrapolated solution method.

In the next section, we study parabolic equations that are problems of a different mathematical nature, but we try to make use of the methodologies developed for elliptic problems.

Chapter 3

Optimized extrapolation method for parabolic problems

In this chapter, we focus on extending the Least-Square Etraxpolation method to unsteady problem by using coarse grid solutions that have different meshes in space and time.

3.1 Extrapolation method for parabolic problems

Following the same methodology as the one used in [46], we will consider problems of the following form

$$\frac{\partial u}{\partial t} = N[u], (x,t) \in \Omega \times (0,T) \tag{3.1}$$
$$u_{|\partial\Omega} = g(t), t \in (0,T)$$
$$u(x,0) = v(x), x \in \Omega$$

where N is an elliptic operator. We will use the notation L instead of N if the operator is linear. We assume the parabolic problem well-posed and that it has a unique solution. For the simplicity of the presentation, we will assume that the problem has one space dimension. However, our method is easily generalizable to a higher dimension problem, as shown in the last section of our numerical experiments. The main impact of a higher dimension problem will be the computational cost of the method with respect to the optimization procedure and also the potential complexity of the basis functions.

In the following sections, we first recall the principles of extrapolation for scalar functions in space and time. Then, we extend the concepts to grid functions. Finally, we introduce the optimized extrapolated solution method.

3.1.1 Extrapolation for continuous functions

Let $(E, ||.||)$ be a normed linear space. Let p_x, p_t, q_x, q_x be some positive integer. Finally, let (dx^0, dt^0) be a couple of positive real numbers. Let $u(x,t)$ be an element of E. $u(x,t)$ is the exact continuous solution in Ω of Equation 3.1. Let $u^{i,n}$, $i = 1 \dots 3$, $n = 1 \dots 3$ be elements of E that have the following asymptotic expansions:

$$u^{i,n} - u = C_1 \left(\frac{dx}{2^{i-1}} \right)^{p_x} + C_2 \left(\frac{dt}{2^{n-1}} \right)^{p_t} + o(dx^{q_x}, dt^{q_t}), \tag{3.2}$$

where C_1 and C_2 are constants independent of the discretization parameters dx and dt. Therefore, the asymptotic expansion $u^{i,n}$ satisfies $||u - u^{i,n}|| = O(dx^{p_x}, dt^{q_t})$.

If the orders of asymptotic expansions in space or time are known, i.e., p_x or p_t are known, then u^i_π defined by the Richardson Extrapolation formula in space is,

$$u^i_\pi = \frac{2^{p_x} u^{i+1} - u^i}{2^{p_x} - 1}, \quad i = 1, 2 \tag{3.3}$$

and satisfies $||u - u^i_\pi|| = o(dx^{p_x})$.

Similarly, u^n_ϕ defined by the Richardson Extrapolation formula in time is,

$$u^n_\phi = \frac{2^{p_t} u^{n+1} - u^n}{2^{p_t} - 1}, \quad n = 1, 2 \tag{3.4}$$

and satisfies $||u - u^n_\phi|| = o(dt^{p_t})$.

Combining Equations 3.3 and 3.4, we can build the Richardson Extrapolation in space and time, as follows:

$$u^{n,i}_r = \frac{2^{p_t} 2^{p_x} u^{i+1,n+1} - 2^{p_t} u^{i,n+1} - 2^{p_x} u^{i+1,n} + u^{i,n}}{(2^{p_t} - 1)(2^{p_x} - 1)}, \quad i, n = 1, 2. \tag{3.5}$$

The error between the solution u and u_r, defined by Richardson Extrapolation in space and time, satisfies $||u - u^{n,i}_r|| = o(dx^{p_x}, dt^{p_t})$. An *a posteriori* error estimate on $u^{i,n}$ will then be

$$||u^{i,n} - u^{n,i}_r||. \tag{3.6}$$

In general, the method might not provide a way to know or ascertain the asymptotic orders of convergence in space and/or time on the computational grids. However, we can compute estimates of these orders independently by using Equations 3.3 and 3.4:

$$p_x(t) = log_2 \frac{||u^i - u^{i+1}||}{||u^{i+1} - u^{i+2}||} \; and \tag{3.7}$$

$$p_t(x) = log_2 \frac{||u^n - u^{n+1}||}{||u^{n+1} - u^{n+2}||}. \tag{3.8}$$

Richardson Extrapolation is a particular case of an extrapolation method. We should now introduce a general definition for extrapolation methods.

Let u be in $E = L^2(\Omega)$. Let $(v^k_{dx,dt})_{k=1\ldots4}$ be four approximations of u in E:

$$v^k_{dx,dt} \to u \in E \text{ as } dx, dt \to 0, \; k = 1 \ldots 4. \tag{3.9}$$

A linear extrapolation of these four functions will write

$$\alpha_1 v^1_{dx,dt} + \alpha_2 v^2_{dx,dt} + \alpha_3 v^3_{dx,dt} + \alpha_4 v^4_{dx,dt}. \tag{3.10}$$

The consistency of the extrapolation requires having

$$\lim_{dx,dt \to 0} \alpha_1 v^1_{dx,dt} + \alpha_2 v^2_{dx,dt} + \alpha_3 v^3_{dx,dt} + \alpha_4 v^4_{dx,dt} = u. \tag{3.11}$$

Therefore, the coefficients of the extrapolation are constrained by the equation $\alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 = 1$.

So far, we have only recalled the principles behind Richardson Extrapolation in space and time for scalar functions. In the next section, we extend this study to grid functions.

3.1.2 Extrapolation for grid functions

In practice, we work with grid functions that approximate the continuous solutions of discretized PDE problems on regular grids.

Let χ be the discretization step in space and τ the discretization step in time. Let $E_{i,n}$ be a family of normed linear spaces associated with a mesh $(\mathcal{M}^j)_{j=1...n}$. For simplicity, we consider that the four meshes $(\mathcal{M}^j)_{j=1...n}$ are embedded space time meshes. Let $U^{(i,n)}$ be the numerical approximation of u on the mesh $\mathcal{M}^{2*(i-1)+n}$.

We suppose that the discrete grid functions have the following asymptotic expansion:

$$U^{(i,n)} = u + C_1(\chi)^{p_x} + C_2(\tau)^{p_t} + o(\chi^{q_x}, \tau^{q_t}), \tag{3.12}$$

where C_1 and C_2 are independent of the discretization parameters χ and τ.

For Richardson Extrapolation in space *and* time, we have then the following results.

Theorem 1: *There exists a unique linear combination of the coarse grid solutions* $U^{(i,n)}$ *with constant weights* $\alpha_1, \alpha_2, \alpha_3, \alpha_4$ *such that*

$$\alpha_1 U^{(1,1)} + \alpha_2 U^{(2,1)} + \alpha_3 U^{(1,2)} + \alpha_4 U^{(2,2)} - u = o(\chi^{p_x}) + o(\tau^{p_t}). \tag{3.13}$$

The $(\alpha_i)_{i=1...3}$ *are:*

$$\alpha_1 = \frac{1}{(2^{p_x}-1)(2^{p_t}-1)}, \tag{3.14}$$

$$\alpha_2 = -\frac{2^{p_t}}{(2^{p_x}-1)(2^{p_t}-1)}, \text{ and} \tag{3.15}$$

$$\alpha_3 = -\frac{2^{p_x}}{(2^{p_x}-1)(2^{p_t}-1)}. \tag{3.16}$$

Further, the consistency of the extrapolation formula implies

$$\alpha_4 = 1 - \alpha_1 - \alpha_2 - \alpha_3. \tag{3.17}$$

Proof. Using the asymptotic expansion of Equation 3.2 for $U^{(1,1)}$, $U^{(2,1)}$, $U^{(1,2)}$, and $U^{(2,2)}$, we have:

$$U^{(1,1)} - u = C_1\chi^{p_x} + C_2\tau^{p_t} + o(\chi^{q_x}) + o(\tau^{q_t}), \tag{3.18}$$

$$U^{(1,2)} - u = C_1\chi^{p_x} + C_2(\tau/2)^{p_t} + o(\chi^{q_x}) + o(\tau^{q_t}), \tag{3.19}$$

$$U^{(2,1)} - u = C_1(\chi/2)^{p_x} + C_2\tau^{p_t} + o(\chi^{q_x}) + o(\tau^{q_t}), \text{ and} \tag{3.20}$$

$$U^{(2,2)} - u = C_1(\chi/2)^{p_x} + C_2(\tau/2)^{p_t} + o(\chi^{q_x}) + o(\tau^{q_t}). \tag{3.21}$$

Canceling the time-dependent reminder between equations (a) and (b) (respectively (c) and (d)) allows us to derive the following two equations:

$$U^{(1,1)} - 2^{p_t}U^{(1,2)} = (1-2^{p_t})u + 2^{p_t}C_1\chi^{p_x} + o(\chi^{q_x}) + o(\tau^{q_t}); \tag{3.22}$$

$$U^{(2,1)} - 2^{p_t}U^{(2,2)} = (1-2^{p_t})u + 2^{p_t}C_1(\frac{\chi}{2})^{p_x} + o(\chi^{q_x}) + o(\tau^{q_t}). \tag{3.23}$$

Combining these two equations to cancel the space-dependent reminder, we have:

$$U^{(1,1)} - 2^{p_t}U^{(1,2)} - 2^{p_x}U^{(2,1)} - 2^{p_t}2^{p_x}U^{(2,2)} = (1-2^{p_t})(1-2^{p_x})u + o(\chi^{q_x}) + o(\tau^{q_t}). \tag{3.24}$$

Identifying the coefficients with Equation 3.13, we have

$$\alpha_1 = \frac{1}{(1-2^{p_t})(1-2^{p_x})}, \alpha_2 = -\frac{2^{p_t}}{(1-2^{p_t})(1-2^{p_x})},$$
$$\alpha_3 = -\frac{2^{p_x}}{(1-2^{p_t})(1-2^{p_x})}, \text{ and } \alpha_4 = \frac{2^{p_x+p_t}}{(1-2^{p_t})(1-2^{p_x})} \tag{3.25}$$

Taking the limit in Equation 3.13 when $\chi \to 0$ and $\tau \to 0$, we obtain

$$\alpha_1 u + \alpha_2 u + \alpha_3 u + \alpha_4 u - u = 0 \tag{3.26}$$

Therefore,

$$\alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 = 1, \tag{3.27}$$

giving the consistency result. □

One can use this result to approximate a fine grid solution $U_{dx,dt}$. We have

Corollary 1: *There is a unique linear combination of the coarse grid solutions $U^{(i,n)}$ with constant weights $\alpha_1, \alpha_2, \alpha_3, \alpha_4$ such that*

$$\alpha_1 U^{(1,1)} + \alpha_2 U^{(2,1)} + \alpha_3 U^{(1,2)} + \alpha_4 U^{(2,2)} + \tag{3.28}$$
$$C_1 dx^p + C_2 dt^q - U_{dx,dt} = O(\chi^p) + O(\tau^q) \tag{3.29}$$

The α_i are given by Theorem 1 and the constants C_1 and C_2 are the coefficient of the asymptotic expansion of $U_{dx,dt}$.

Proof. Using Equation 3.2 and replacing u in Equation 3.13 of Theorem 1, the result is straightforward. □

Remark 4: *One can notice that this asymptotic expansion is usable only if the term* $C_1 dx^p + C_2 dt^q$ *is not negligible with respect to the error term* $O(\chi^p) + O(\tau^q)$.

In order to make the stability analysis of this time-space Richardson Extrapolation, we use the following error model

$$U^{(i,n)} - u = C_i \chi^{p_x} + C_n \tau^{p_t} + \delta_{2(i-1)+n}, \tag{3.30}$$

where $C_i = c_1(1+\varepsilon_i)$ and $C_n = c_2(1+\xi_n)$. c_1 and c_2 are constants. The parameters $\varepsilon_i,\ i = 1, 2$ and $\xi_n,\ n = 1, 2$ are for the higher order term in the expansion of Equation 3.2, i.e., $\varepsilon = o(1)$ $\xi = o(1)$. Finally, δ_k stands for the numerical error coming, for example, from the imperfect convergence of the iterative scheme used in implicit time stepping. Using a similar approach as for the computation of Equation 3.5 in the case of continuous functions, we can derive a similar formula for grid functions :

$$U_r^{(i,n)} = \frac{2^{p_t} 2^{p_x} U^{i+1,n+1} - 2^{p_t} U^{i,n+1} - 2^{p_x} U^{i+1,n} + U^{i,n}}{(2^{p_t}-1)(2^{p_x}-1)}, \quad i, n = 1, 2. \tag{3.31}$$

Replacing the corresponding terms of Equation 3.31 by Equation 3.30, we obtain the following model for the error:

$$\begin{aligned} U_r^{(i,n)} - u = {} & \frac{\chi^{p_x}}{(2^{p_t}-1)(2^{p_x}-1)} c_1 (2^p(\varepsilon_1 - \varepsilon_3) + (\varepsilon_4 - \varepsilon_2)) \\ & \frac{\tau^{p_t}}{(2^{p_t}-1)(2^{p_x}-1)} c_2 (2^p(\xi_1 - \xi_2) + (\xi_3 - \xi_4)) \\ & \frac{1}{(2^{p_t}-1)(2^{p_x}-1)} (2^{p_t} 2^{p_x} \delta_4 - 2^{p_t} \delta_3 - 2^{p_x} \delta_2 + \delta_1) \end{aligned} \tag{3.32}$$

From Equation (3.32), we can see that the perturbation is amplified by the factor

$$\frac{2^{p_x+p_t} + 1}{(2^{p_t}-1)(2^{p_x}-1)}. \tag{3.33}$$

For small order of convergence in space and/or time, the amplification factor given by Equation 3.33 shows that Richardson Extrapolation in space and time error leads to deteriorated solutions.

Let us now describe the generalization of the least square extrapolation method for space-time problems.

3.1.3 Optimized extrapolation for parabolic equations

The goal of the least square extrapolation methods is to improve the Richardson Extrapolation method, addressing the following concerns:

- the lack of asymptotic expansion with a known order of convergence;
- the further uses of the PDE framework at the grid level; and
- the coefficient α in the Richardson Extrapolation is space-dependent.

For time-independent problems, it was shown in [39, 47, 46] that the method obtained was more robust than the Richardson Extrapolation because it may apply to problems where Richardson Extrapolation has not given convincing results.

Let us denote $U^n_{dx,dt}$ as the solution given at time t_n by a one step time integration scheme. Let us now describe the optimized extrapolation method for space-dependent problems.

The semi-discrete schema for the problem of Equation 3.1 is

$$U^{n+1}_{dx,dt} = G(U^n_{dx,dt}). \tag{3.34}$$

Let u_k^n, $k = 1 \dots 4$ be for grid functions approximating the solution $u_{dx,dt}^n$. As shown in the previous section, in Richardson Extrapolation, the coefficients α_k, $k = 1 \dots 4$ are constant. In this generalization of the optimized extrapolation, we formulate the following problem for the unknown α_k, $k = 1 \dots 4$ that is in general a non-constant function:

(P_α^t)**:** *Find the four weight functions* α_k, $k = 1 \dots 4$ *such that the residual*

$$\sum_{k=1}^{4} \alpha_k u_k^{n+1} - G(\sum_{k=1}^{4} \alpha_k u_k^n), \tag{3.35}$$

is minimum in $L^2(\Omega_x \times \Omega_t)$.

This formulation of the problem does not take into account the nature of the different meshes on which the grid functions u_k^{n+1} exist. The end of this section is dedicated to adjusting this loose notation, and to examining some properties of the OES.

Typically, α_k, $k = 1 \dots 4$ are chosen in a set of polynomial functions for the space $\Lambda(\Omega_x \times \Omega_t)$.

As a matter of fact, let us suppose that we do have the exact α_k, $k = 1 \dots 4$ such that $u = \sum_{j=1}^{4} \alpha_j v_{dx,dt}^j$. Any order ε approximation of α_j with $\varepsilon \ll 1$ can be used for our OES. More precisely:

Theorem 2: *Let* $v_{dx,dt}^j$, $j = 1 \dots 4$ *be four grid functions approximating the solution* u. *Let* α_j, $j = 1 \dots 4$ *be four functions in* $\Lambda(\Omega_x \times \Omega_t)$ *such that* $u = \sum_{j=1}^{4} \alpha_j v_{dx,dt}^j$. *If* α_j *are continuous, and* $v_{dx,dt}^j - v_{dx,dt}^4 = O(dx^{p_x}, dt^{pt})$, *then there exist* α_j^M, $j = 1 \dots 4$

approximations of α_j *to the order* M^{-1}, $M \gg 1$ *such that*

$$u = \sum_{j=1}^{4} \alpha_j^M v_{dx,dt}^j + O(dx^{p_x}, dt^{pt}) \times O(M^{-1}). \tag{3.36}$$

Proof. From Weierstrass's theorem on continuous functions, we can build a sequence of polynomials $(\alpha_j^M)_{M\geq 0}$, $j = 1 \ldots 3$ such that $\alpha_j^M - \alpha_j = O(M^{-1})$, $j = 1 \ldots 3$. A simple development of the previous expression gives

$$\sum_{j=1}^{4} \alpha_j^M v_{dx,dt}^j - u = \sum_{j=1}^{4} \alpha_j^M v_{dx,dt}^j - \sum_{j=1}^{4} \alpha_j v_{dx,dt}^j \tag{3.37}$$

$$= \sum_{j=1}^{3} (\alpha_j^M - \alpha_j) v_{dx,dt}^j - (\sum_{j=1}^{3} (\alpha_j^M - \alpha_j))(v_{dx,dt}^4) \tag{3.38}$$

$$= \sum_{j=1}^{3} (\alpha_j^M - \alpha_j)(v_{dx,dt}^j - v_{dx,dt}^4) \tag{3.39}$$

$$= O(M^{-1}) \times O(dx^{p_x}, dt^{p_t}). \tag{3.40}$$

□

From this theorem, it is not required in practice to compute the α_i exactly, but only an approximation to order $\varepsilon = M^{-1}$.

Now we will see how to build the OES from coarse grid solutions that use different meshes.

The coarse grid solution used in the numerical solution corresponds to the discretization $(\chi, \tau), (\chi/2, \tau), (\chi, \tau/2), (\chi/2, \tau/2)$. We will denote these coarse grids $(\chi/i, \tau/n)$ $\mathcal{M}^k, k = 1, \ldots 4$. The fine grid $\mathcal{M}^*$ used in OES corresponds to $(\chi/4, \tau/4)$. The projected coarse grid solutions are denoted now as $\tilde{U}^{(i,n)}, i \in \{1, 2\}, n \in \{1, 2\}$.

First, we introduce an interpolation operator in space I_χ^{dx}, that projects the coarse grid solution at each time step on the fine grid in space for the same time step:

$$\tilde{U}_{\chi,\tau}^n = I_\chi^{dx}[U_{\chi,\tau}^n]. \tag{3.41}$$

Second, we introduce an interpolation operator in time $\mathcal{I}_\tau^{dt}$ that interpolates the coarse grid solution in time on the fine grid in time at the same physical location:

$$\tilde{U}_{\chi,\tau} = \mathcal{I}_\tau^{dt}[U_{\chi,\tau}]. \tag{3.42}$$

For simplicity of the presentation, we use the generic notation $\tilde{U}$ for all types of projection of U.

Let us assume once and for all that $dt = \tau/4$. The goal is to have a construction of the interpolation of the four coarse grid solutions on the fine grid (dx, dt) using only the information in the time interval $(t^n, t^n + \tau)$.

OES in space-time will then combine ten vectors

$$\begin{aligned} &U_{\chi,\tau}(t^n), U_{\chi,\tau}(t^n+\tau),\\ &U_{\chi/2,\tau}(t^n), U_{\chi/2,\tau}(t^n+\tau),\\ &U_{\chi,\tau/2}(t^n), U_{\chi,\tau/2}(t^n+\tau/2),\\ &U_{\chi,\tau/2}(t^n+\tau), U_{\chi/2,\tau/2}(t^n),\\ &U_{\chi/2,\tau/2}(t^n+\tau/2),\ \textit{and}\ U_{\chi/2,\tau/2}(t^n+\tau). \end{aligned} \tag{3.43}$$

From $U_{\chi,\tau}^n$ and $U_{\chi,\tau}^{n+1}$, one can compute $\tilde{U}_{\chi,\tau}$ at time steps $t^n + jdt, j = 1 \ldots 3$ using linear interpolation. This method is second order in time.

Higher order time interpolation can be built if one uses Equation 3.1 in order to obtain the time derivative. Indeed, by providing the solution $(U^n_{\chi,\tau})$, the derivative in time of the solution $(N[U^n_{\chi,\tau}])$ one can use piecewise cubic Hermite interpolation that is of order three in time:

$$H_i(t) = a_i + b_i(t - t_i) + c_i(t - t_i)^2 + d_i(t - t_i)^2(t - t_{i+1}), \tag{3.44}$$

where a_i, b_i, c_i, d_i are solutions of

$$H_n(t_n) = U^n_{\chi,\tau}, \tag{3.45}$$

$$H'_n(t_n) = N[U^n_{\chi,\tau}], \tag{3.46}$$

$$H_n(t_{n+1}) = U^{n+1}_{\chi,\tau}, \tag{3.47}$$

$$H'_n(t_{n+1}) = N[U^{n+1}_{\chi,\tau}]. \tag{3.48}$$

With $U^n_{\chi,\tau/2}$ we have one more time set of coarse solutions to use.

Then, one can use either quadratic interpolation that is third order in time, or Hermite interpolation that is fifth order in time.

Applying interpolation in time, followed by interpolation in space

$$\tilde{U}_{\chi,\tau} = I^{dx}_{\chi}[\mathcal{I}^{dt}_{\tau}[U_{\chi,\tau}]], \tag{3.49}$$

we are able to finally build, for each coarse time step $(t^n, t^n + \tau)$, a projection of all four coarse grid solutions $U_{i,n}$ on the fine grid of space step dx and time step dt.

We recall that thanks to the consistency of the extrapolation method, we have $\alpha_4 = 1 - \sum_{j=1}^{3} \alpha_j$.

Because we compute the OES for each coarse time step separately, we will assume that the weight functions are space dependent only. For convenience, we will apply the same approximation in space as with the steady case.

Let us restrict ourselves to the case where the α_j are constant. The goal now is to show that OES is a general method that applied to different types of discretization.

One can show

Theorem 3: *If the following two assumptions are true,*

- *the asymptotic expansion of Equation 3.2 is valid in the discrete L^2 norm for the coarse grid solution used in OES;*
- *the consistency error for the one-step scheme (Equation 3.34) is asymptotically equivalent to the error on the solution;*

then the OES solution for the α_j coefficients is asymptotically equivalent to the RE solution within order 2.

Proof. For the simplicity of the demonstration, we will assume that the interpolation in space is performed using piecewise linear interpolation, and the interpolation in time is done using either Lagrange interpolation or Hermite interpolation.

First, considering Lagrange polynomial interpolation in time between coarse grid and fine grids (I_{τ}^{dt} for $\mathcal{M}_1$ and $I_{\tau/2}^{dt}$ for $\mathcal{M}_2$) between time t^n and $t^n + 4dt$ and space x_i and $x_i + 4dx$:

$$\mathcal{I}_{\tau}^{dt}(t) = \frac{U_n^1 - U_{n+1}^1}{2dt}(t^n - t) - U_n^1 \; and \tag{3.50}$$

$$\mathcal{I}_{\tau/2}^{dt}(t) = U_n^2 + \frac{3U_n^2 - 4U_{n+1/2}^2 + U_{n+1}^2}{2dt}(t^n - t) + \frac{U_n^2 - 2U_{n+1/2}^2 + U_{n+1}^2}{2dt^2}(t^n - t)^2 \tag{3.51}$$

These two polynomials are used to compute the values on the fine grid in time at time $(t^n, t^n + dt, t^n + 2dt, t^n + 3dt, t^n + dt)$. The values obtained are then interpolated on the fine grid in space using piecewise linear interpolation.

The next step is to construct the solution on the fine grid in space using this interpolated solution:

$$I_{\chi}^{dx}(x) = \frac{\tilde{U}_i^1 - \tilde{U}_{i+1}^1}{2dx}(x_i - x) - \tilde{U}_i^1 \; and \tag{3.52}$$

$$I_{\chi/2}^{dx}(x) = \begin{cases} \frac{\tilde{U}_i^2 - \tilde{U}_{i+1/2}^2}{2dx}(x_i - x) - \tilde{U}_i^2 \text{ if } x_i \leq x \leq x_{i+1/2} \\ \frac{\tilde{U}_{i+1/2}^2 - \tilde{U}_{i+1}^2}{2dx}(x_{i+1/2} - x) - \tilde{U}_{i+1/2}^2 \text{ if } x_{i+1/2} \leq x \leq x_i \end{cases}, \tag{3.53}$$

where $\tilde{U}$ is the interpolated solution on the fine time grid.

Replacing the value of τ, $\tau/2$,χ and $\chi/2$ in Equation 3.13, and combining the interpolated value obtained from Equation 3.52, one can evaluate the error on the fine grid in space and time. Three cases have to be distinguished:

- Case 1 - Interpolation at nodes matching the coarse grid level: interpolations via Equations 3.50 and 3.52 are exact at those nodes, and thus the error is given by Equation 3.13. The error given by the interpolation of those nodes is

therefore equal to zero:

$$\varepsilon_1 = 0. \tag{3.54}$$

- Case 2 - Interpolation at nodes matching the intermediate grid level: nodes of intermediate grids are matching, but the coarse grid interpolation introduces an error given by

$$\varepsilon_2 = O(\chi^2) + O(\tau^2), \tag{3.55}$$

since the interpolation is piecewise linear. The final error is given by accumulating this interpolation error and that given by Equation 3.13.

- Case 3: Interpolation at nodes present only at fine grid level: we introduce another error in the result that is the accumulation of the errors from interpolation from the coarse grids and the intermediate grids to the fine grid. Each coarse grid introduces an error of the order that is

$$\varepsilon_3 = O(\chi^2) + O(\tau^2). \tag{3.56}$$

In all three cases, the error is given by $\varepsilon = \varepsilon_1 + \varepsilon_2 + \varepsilon_3 = O(\chi^2) + O(\tau^2)$. Since $\chi = 4dx$ and $\tau = 4dt$, the asymptotic error for the OES solution given by the coefficients α_j is of the same order as the Richardson Extrapolation. □

Remark 5: *For second order elliptic operator N, access to second order derivatives allows us to use spline interpolation for I_{χ}^{dx}, and produced fewer perturbations on the interpolated solution than in polynomial interpolation.*

Remark 6: *In the two-dimensional case, the result can be easily generalized if the PDE is separable in space. Let N be an operator separable in space for which the two components are N_1 and N_2. The solution u of Equation 3.1 is $u = u^{(1)}(x,t)u^{(2)}(y,t)$. The two equations to solve are:*

$$\frac{\partial u^{(1)}}{\partial x} = N_1(u^{(1)}) \text{ and} \tag{3.57}$$

$$\frac{\partial u^{(2)}}{\partial y} = N_2(u^{(2)}), \tag{3.58}$$

with the corresponding boundary conditions.

Applying the optimized extrapolation method described previously to each equation leads to the following:

$$u^{(1)} - u^{(1)}_{\chi} = O(\chi^2) + O(\tau^2) \text{ and} \tag{3.59}$$

$$u^{(2)} - u^{(2)}_{\chi} = O(\chi^2) + O(\tau^2). \tag{3.60}$$

Hence,

$$\begin{aligned} u^{(1)}_{\chi} u^{(2)}_{\chi} = u^{(1)}u^{(2)} &- u^{(1)}(O(\chi^2) + O(\tau^2)) \\ &- u^{(2)}(O(\chi^2) + O(\tau^2)) + (O(\chi^2) + O(\tau^2))^2 \end{aligned} \tag{3.61}$$

$$u_{\chi} = u - O(\chi^2) + O(\tau^2). \tag{3.62}$$

We are now going to present our numerical experiments with the OES method.

3.2 Algorithm for least square extrapolation method

In this section, we present the algorithm used to perform the computation of the optimized extrapolated solution.

3.2.1 Algorithm

Inspired from the algorithm of LSE for the steady problem, the procedure for parabolic OES is as follows.

We define a meta-time step $dT = 2\tau$. It will be used as a synchronization point for coarse grid computations. For each meta-time step, we use the following algorithm.

1. Call Solver: solve the problem on $\mathcal{M}^1$, $\mathcal{M}^2$, $\mathcal{M}^3$ and $\mathcal{M}^4$, possibly in parallel.

2. Call fine mesh: generate a fine mesh $M(dx_\infty, dt_\infty)$ that is supposed to solve all scales of the problem. $M(dx, dt)$ is preferably a structured grid in space *and* time. We must have $dx \ll \chi$ and $dt_\infty \tau$.

3. Call Projection: project the coarse solutions $U^{1,1}$, $U^{1,2}$, $U^{2,1}$ and $U^{2,2}$ onto $M(dx_\infty, dt_\infty)$ and postprocess them to avoid spurious oscillations due to the interpolating function. There are several ways of performing this postprocessing of the interpolated solutions. First, one could filter the solution using a low-pass frequency filter that will cut off all the frequencies greater than some pre-defined values. A second method could be to use a relaxation technique as done in the Least-Square Extrapolation method [60, 47, 46]. Since we have a time-dependent problem, a better strategy could be to use the Symmetric Successive Over-Relaxation (SSOR).

4. solve minimization problem defined by Equation 3.1.3 using an off-the-shelf optimization package. The starting point of the optimization procedure can be set to be RE, but it is not a necessity.

After the completion of a meta-time step, the optimized extrapolated solution is found. It can be reused in a subsequent meta-time step as an initial condition, since its accuracy is better than that of the coarse grid solution.

This procedure applies to meta-time step, i.e., a sequence of small time steps for which an *a posteriori* estimate is expected. Hence, the role of time and space are not completely symmetric. Performing step 3 is expensive, but the operations involved, interpolation and smoothing, are mainstream, and yield possible parallelism.

The optimization involved in step 4 is critical and cumbersome, but offers good parallelism since only weak synchronization is required. The optimization procedure requires relaxing intermediate solutions multiple times. The results obtained for each computation are independent and are not required for subsequent relaxation steps. Therefore, in order to improve the efficiency of the process, one can imagine performing several relaxations in parallel.

In the next sections, we will examine different test cases taken from applications in biology and heat transfer.

3.2.2 1D Simulation

In this section, we focus on two distinct 1D problems. The first one is a heat transfer problem for which an analytical solution is known. The second one is the problem of reactive shock layer for reactive flows.

3.2.2.1 Thermal wave

In order to validate the methodology, a test problem should have an analytical solution and should have real variation in time. An exact solution allows the computation of exact errors, while time dependency supplies a way to try different features of the method. It is also good practice to choose a problem for which the literature provides enough data to compare the results. The first problem in the validation process of the optimized extrapolation method is the thermal wave problem,

$$\frac{\partial T}{\partial t} = \frac{\partial^2 T}{\partial x^2} + 8T^2(1 - T). \tag{3.63}$$

We refer to [61] for the notation and values of the parameters.

We studied the performance of OES for an individual time step. We also experimented numerically the stability of the time integration scheme using RE or OES to predict an improved numerical solution at the end of each time step. For time stepping, we used the implicit schemes, first order backward Euler method and Crank-Nicholson method.

We use the unconstrained minimization subroutine of Matlab to compare results with different choices of the norm, i.e., either the discrete L^2 norm or the maximum norm. There are three unknown coefficients, the consistency formula giving the fourth one. The starting points for the search are the set of Richardson Extrapolation coefficients. This differs from the Least Square Extrapolation method since the method does not rely on the minimization of a response surface, but on a direct optimization procedure.

We reached the following conclusions:

- RE does not work on the fine grid $\mathcal{M}^*$, but may work well on the coarse grid $\mathcal{M}^1$ at time steps $k\tau$, where τ is the coarse time step. The main reason is that RE is too sensitive to perturbations introduced by linear interpolation in time, and therefore becomes unstable.

- One requires few SSOR smoothing iterations of $\tilde{U}^{(i,n)}$ on $\mathcal{M}^*$ solutions to have OES performing better than on the fine grid solution $\tilde{U}^{(2,2)}$. SSOR does not solve the fine grid problem. Instead, SSOR nicely removes the high frequency components of the projected coarse grid solutions.

  The algorithm works as follows:

```
for k=1, max_ksor
  for kt = 1, $N_t$
    Initialize $u_0 = u^{(k_t-1)}$
    Compute RHS
    Perform one SSOR iterate
  end
end
```

  In this algorithm, *max_ksor* is the number of space-time SSOR iterations and N_t is the number of fine grid time steps. Each time step solution undergoes a solution procedure loop, which is the inner loop, and a SSOR relaxation loop, which is the outer loop. This "inversion" is responsible for the smoothing in space *and* time.

- Let us use for the RE formula the expression in Theorem 1 that approximates

the exact solution. OES is designed to approximate the fine grid solution on $\mathcal{M}^*$. One can have OES better than RE for $\mathcal{M}^*$ and at the same time OES worse than RE as an approximation of the exact solution. We will investigate the use of a fine grid solution with smaller time steps and space steps.

- The higher the order of the scheme, and/or the finer the discretization, the more iterations of SSOR we need. The number of SSOR iterations is an open problem.

- OES gives best results for under-resolved solutions with a low order scheme. Indeed, under-resolved solutions have large time steps and space steps, as well as Newton iterations that cannot reach complete convergence.

- Smaller residual on $\mathcal{M}^*$ does not lead to smaller errors. The lack of monotonicity in the stability estimate results in poor performance of OES. This problem is as in the steady case, the result of the spurious high frequency components of the projected fine grid solutions. The postprocessing step with SSOR is then essential to recover this monotonicity between residual and errors.

- Filtering the residual and/or the solution in space, as we did in the steady case, might be beneficial for large time step.

Let us illustrate these preliminary conclusions with a few numerical results. First, we address the performance of the method with the result for the first time step. Figure 3.1 shows the residual at a given intermediate time step for the projected solution.

Figure 3.1 shows results when using Lagrange interpolation between different time solutions and a Crank-Nicholson time stepping scheme. It demonstrates the need to filtrate the residual.

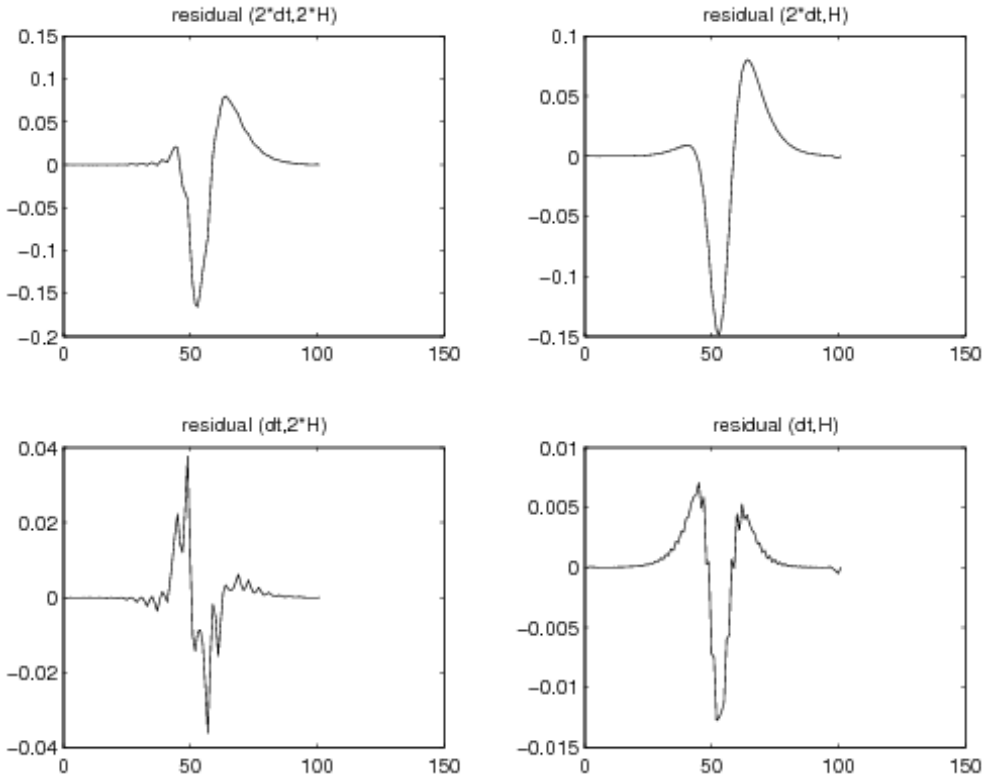

Figure 3.1: Non-filtered residual at a given time step.

Figure 3.2 reveals that with plain OES, minimization of the residual does not necessarily result in minimizing the error. The blue diamond is for the result with OES, and the red cross is for RE. Each of the points of the cloud is for a different combination of weight function. We have plotted roughly several hundred of them. We see a large discrepancy between the L^2 and the L^∞ results.

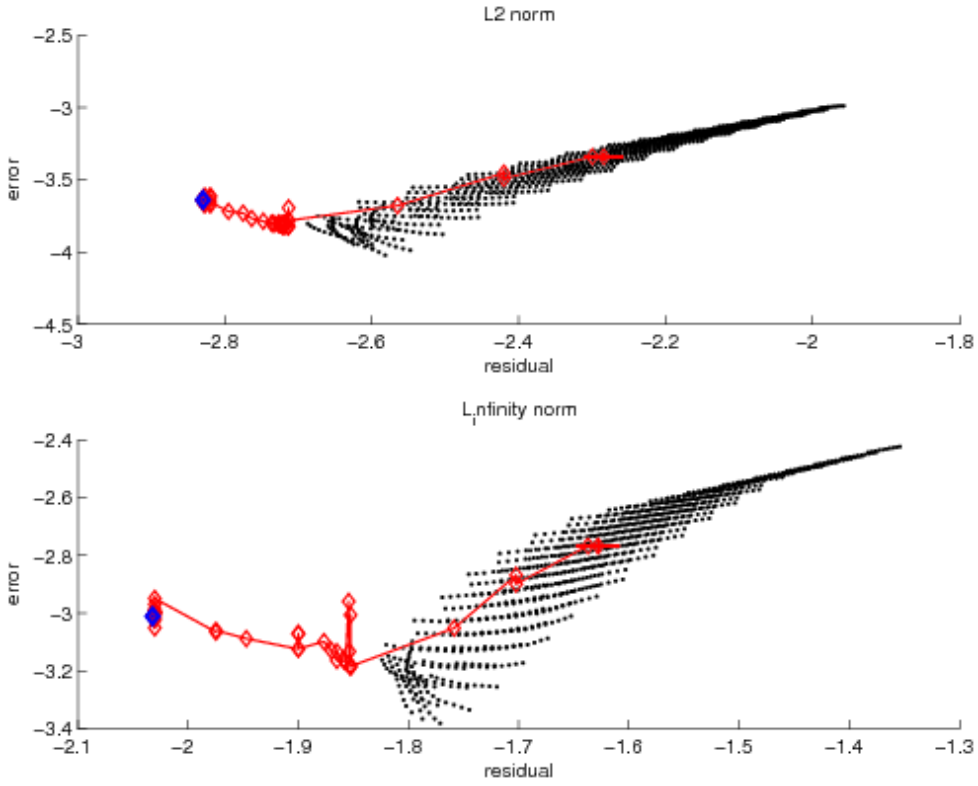

Figure 3.2: Error versus residual in a given norm: in red is the path followed by the optimization procedure, starting at RE (red cross) to convergence (blue dot).

In this optimization process, the goal of the time-space SSOR iterations is to remove high frequencies coming from the interpolation. Typically, the number of SSOR iterations is between 5 and 10; thus there are not enough to solve the PDE. Figure 3.3 displays the minimization path when applying SSOR relaxation. The minimization procedure produces better residual and error over the non-relaxed procedure.

Figure 3.4 displays results for Backward Euler. The vertical axis shows the L^2 error versus the exact analytical solution of Equation 3.63. The horizontal axis is for

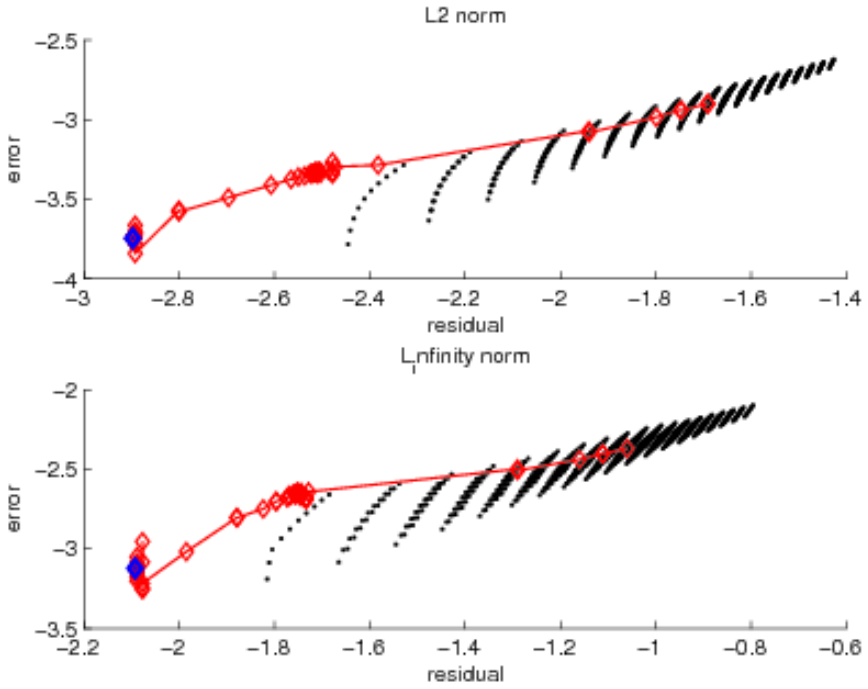

Figure 3.3: Error versus residual in a given norm: in red is the path followed by the optimization procedure, starting at RE (red cross) to convergence (blue dot) with SSOR relaxation.

time. The grid is very coarse in space, and RE is not accurate. OES predicts the fine grid solution fairly well .

Figure 3.5 shows a similar result for Crank-Nicholson method. In the last two figures, we have used few SSOR on the fine grid on each time interval $(t^n, t^n + dT)$.

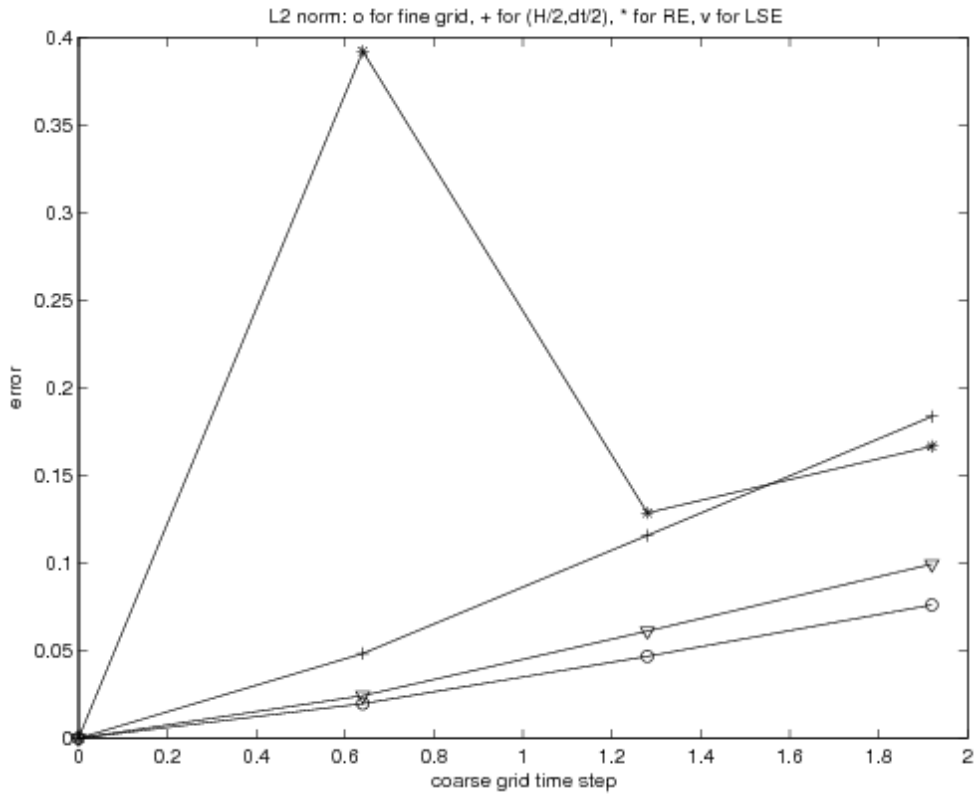

Figure 3.4: Backward Euler with coarse grids.

3.2.2.2 Reactive Shock Layer equations

In order to extend the optimized extrapolation to immersed boundary method in the future, we need to investigate its behavior for problems with stiff variations in space and time. The next models we are interested in describe the behavior of a reactive shock layer. The interest in these equations is the high non-linear behavior close to the shock position. In [62, 63, 64], one can see that the behavior of the numerical solution is link to different parameters. The model we are using is the one proposed

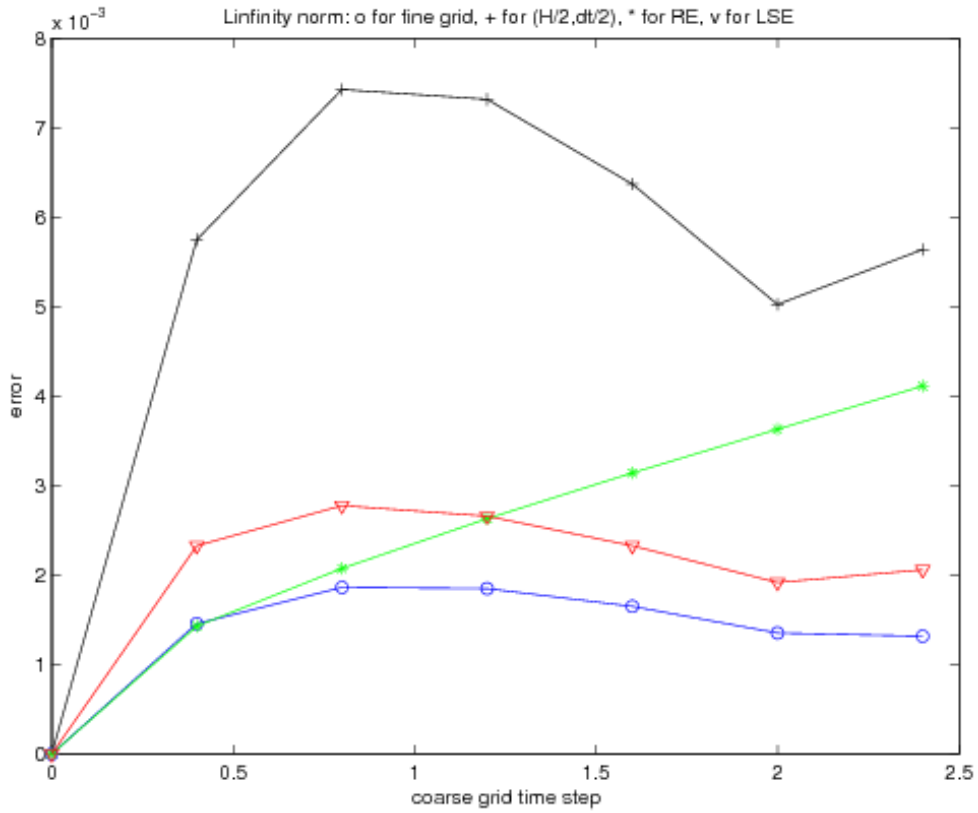

Figure 3.5: Crank-Nicholson with coarse grid as well.

by Majda [65]:

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}[F(u) - q_0 Z] = \varepsilon \frac{\partial^2 u}{\partial x^2} \tag{3.64}$$

$$Z_x = \varepsilon^{-1} \phi(u) Z, \tag{3.65}$$

where $\phi(u) = 1$ if $u > 0$ and 0 elsewhere.

In all the following experiments, we chose $u_l = 1.0$, $u_r = -1.5$, $x_0 = 0$ (position of the jump), $\varepsilon = 0.005$, $q_0 = 2.375$, $T = 0.5$ (simulation time). The coarse grids in space are, respectively, $N_x = 100$ and $N_x = 200$, and the fine grid in space is $N_x =$

400. The finest grid in space should satisfy the CFL condition, thus constraining the choice of the coarse grid in space.

A first numerical approach uses the traditional finite differences:

$$\frac{U_i^{n+1} - U_i^n}{\Delta t} = \varepsilon \frac{U_{i+1}^n - 2U_i^n - U_{i-1}^n}{\Delta x^2} - \frac{F(U_{i+1}^n) - F(U_{i-1}^n)}{2\Delta x} + q_0 \frac{Z_{i+1}^n - Z_{i-1}^n}{2\Delta x}, \tag{3.66}$$

$$\frac{Z_i^{n+1} - Z_{i-1}^{n+1}}{\Delta x} = K\phi(U_{i+1}^{n+1}) Z_{i+1}^{n+1}. \tag{3.67}$$

This discretization is far from being the optimal one, but our only interest is to test the behavior of our *a posteriori* error estimate. The solution is shown in Figure 3.6. The solution next to the jump in $x = 0$ shows one of the problems with the model.

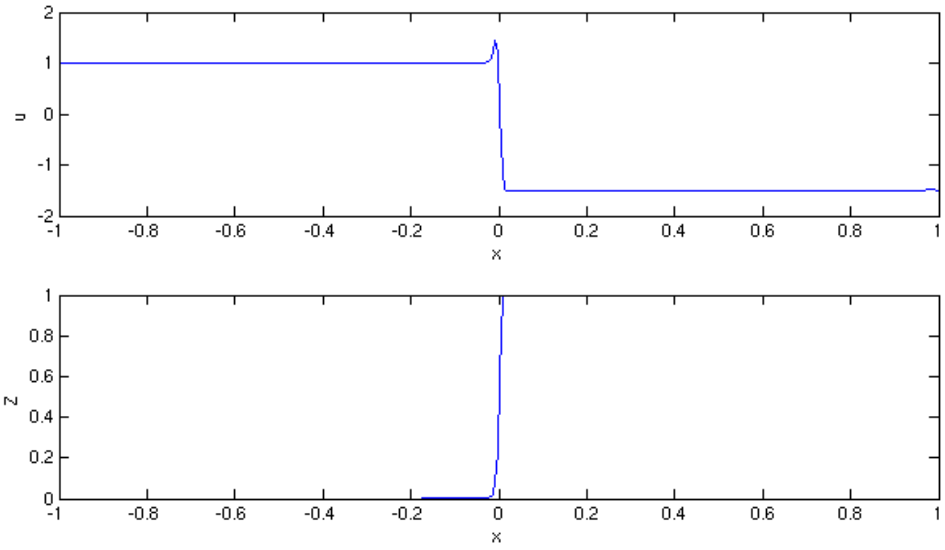

Figure 3.6: Solution for Reactive Shock Layer equation using finite differences.

Using the same methodology as for the 1D heat equation, we apply our *a posteriori* error estimate. The results are shown in Figure 3.7. This figure puts into light different aspects of the methodology. First, the browsing of the parameter space using the L_2 norm shows that simple linear combinations of computed solutions do not yield to improved solutions with respect to Richardson Extrapolation (red dot). Second, the method does not detect that the numerical model is not adapted to the problem and yields to inaccurate solution.

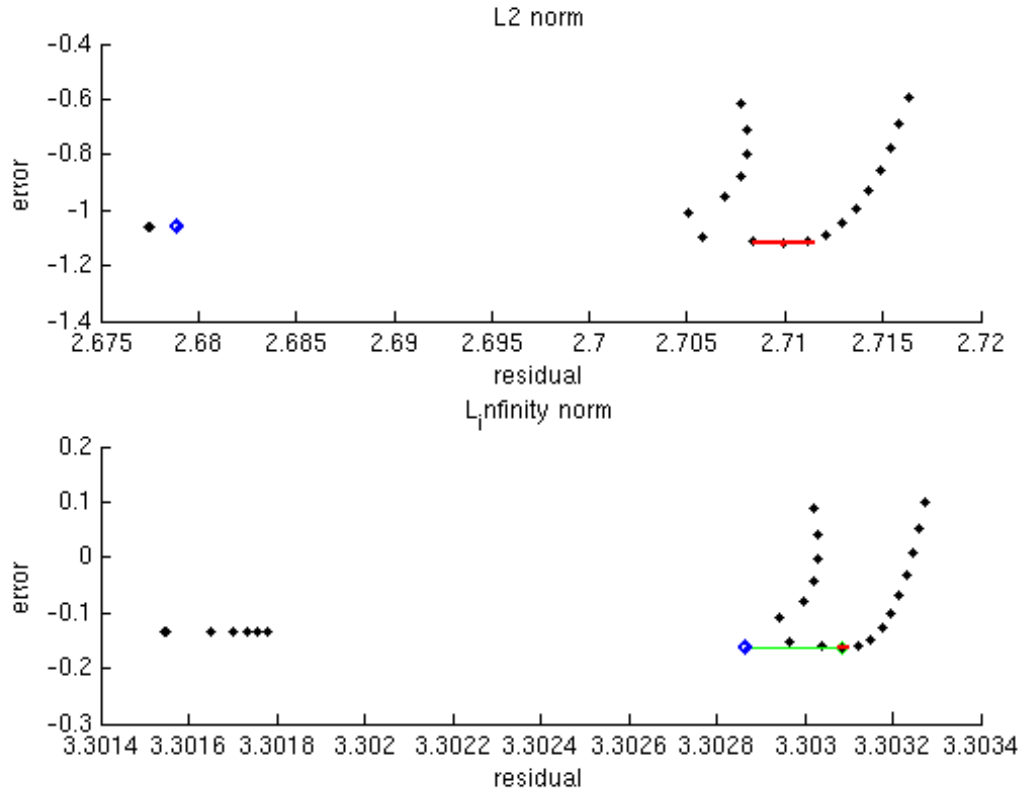

Figure 3.7: Optimization path for Reactive Shock Layer equation using finite differences.

The Piecewise Parabolic Method (PPM) described in [66], is an improved numerical method to perform the simulation of this equation. The solution for this model is shown in Figure 3.8. In this case, the solution next to the jump does not exhibit the same artifacts as those of Figure 3.7. The coefficients used in the model are those specified in [65].

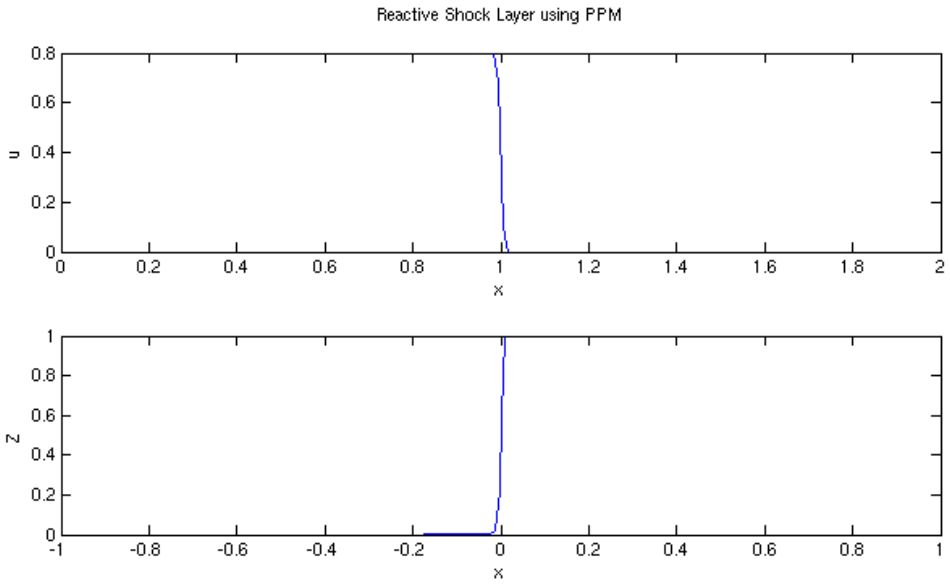

Figure 3.8: Solution for Reactive Shock Layer equation using PPM.

Once again, we use our *a posteriori* error estimate on this equation. Figure 3.9 shows the results for the L_2 norm and the L_∞ norm. For the L_2 norm, the procedure provides next to no improvement with respect to the Richardson Extrapolation, and the browsing of the parameter space indicates that the minimum of the residual does not necessarily map to the minimum of the error.

For the L_∞ norm, the behavior of the procedure is completely different. Our *a*

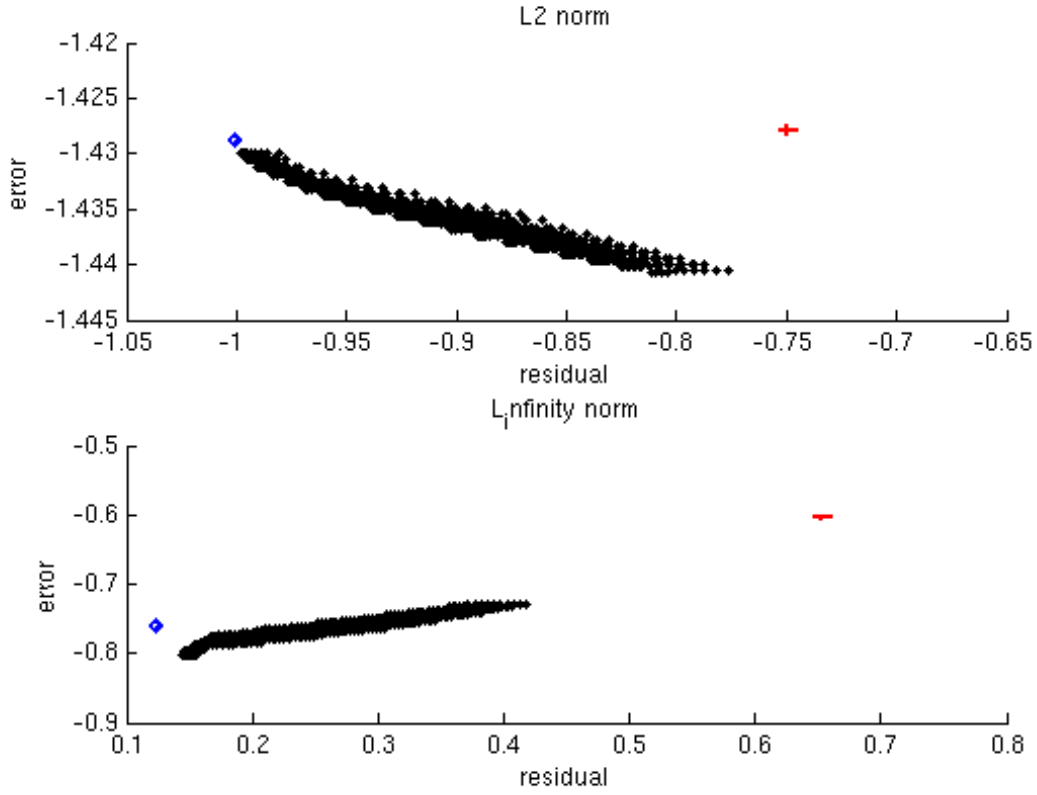

Figure 3.9: Optimization path for Reactive Shock Layer equation using PPM.

posteriori estimate provides a better evaluation than the Richardson Extrapolation, and we have a better mapping between the minimum of the residual and the minimum of the error, in spite of the fact that they do not match exactly. This difference of behavior between the L_2 norm method and the L_∞ norm method reflects the fact that the error in the discontinuous solution is better evaluated when the L_∞ norm is used.

The experiments on these two different numerical models show that while our a

posteriori method is sensitive to the quality of the model used for the stiff problem, the evaluation of the error in the case of an inaccurate model is not guaranteed. Moreover, different simulation parameters, initial conditions, or coefficients will lead to different interpretations of the results. A complete investigation of the range for which the *a posteriori* error estimate is functional for the Reactive Shock Wave equation is feasible, but such exhaustive analysis is of limited interest and is beyond the scope of this section.

3.2.3 2D Simulation

In this section, we extend the approach describe for one-dimensional problems to two-dimensional problems.

3.2.3.1 Fisher equation

For the 2D problem, we consider the Fisher equation for which some exact solutions can be computed, as was shown in [67]:

$$\frac{\partial T}{\partial t} = \frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} + T(1 - T), \tag{3.68}$$

with the boundary conditions given by

$$
\begin{aligned}
T(x_0, y, t) &= \frac{3}{2}\,\frac{1}{|cos(\frac{y}{2})| + exp(\frac{\gamma}{2}(x_0 - V_p t))}, \\
T(x_1, y, t) &= \frac{3}{2}\,\frac{1}{|cos(\frac{y}{2})| + exp(\frac{\gamma}{2}(x_1 - V_p t))}, \\
T(x, y_0, t) &= \frac{3}{2}\,\frac{1}{|cos(\frac{y_0}{2})| + exp(\frac{\gamma}{2}(x - V_p t))},\ \textit{and} \\
T(x, y_1, t) &= \frac{3}{2}\,\frac{1}{|cos(\frac{y_1}{2})| + exp(\frac{\gamma}{2}(x - V_p t))}.
\end{aligned}
\tag{3.69}
$$

and the exact solution is given by

$$
T(x, y, t) = \frac{3}{2}\frac{1}{|cos(\frac{y}{2})| + exp(\frac{\gamma}{2}(x - V_p t))}. \tag{3.70}
$$

Using the same notations as in the 1D case, the coarse grids used to build the numerical solutions were discretized according to (χ, χ, τ), $(\chi/2, \chi/2, \tau)$, $(\chi, \chi, \tau/2)$, $(\chi/2, \chi/2, \tau/2)$. The fine grid G^* used in the optimized extrapolated solution procedure corresponds to $(\chi/4, \chi/4, \tau/4)$. The projected coarse grid solutions are denoted now as $\tilde{U}_{i,j}, i, j \in \{1, 2\}$.

The solution on the fine grid is shown in Figure 3.10.

We are interested again in the behavior of the optimization procedure when the initial set of coefficients are the RE coefficients. First, Figure 3.11 shows the minimization path for OES in space and time without SSOR relaxation. It appears that the OES procedure again allows obtaining an error that is inferior to the one produced by RE. It also appears that minimizing the residual does not mean that the error is minimized. The background points in this figure correspond to the computation effected with regular meshing of the parameter space. While this systematic

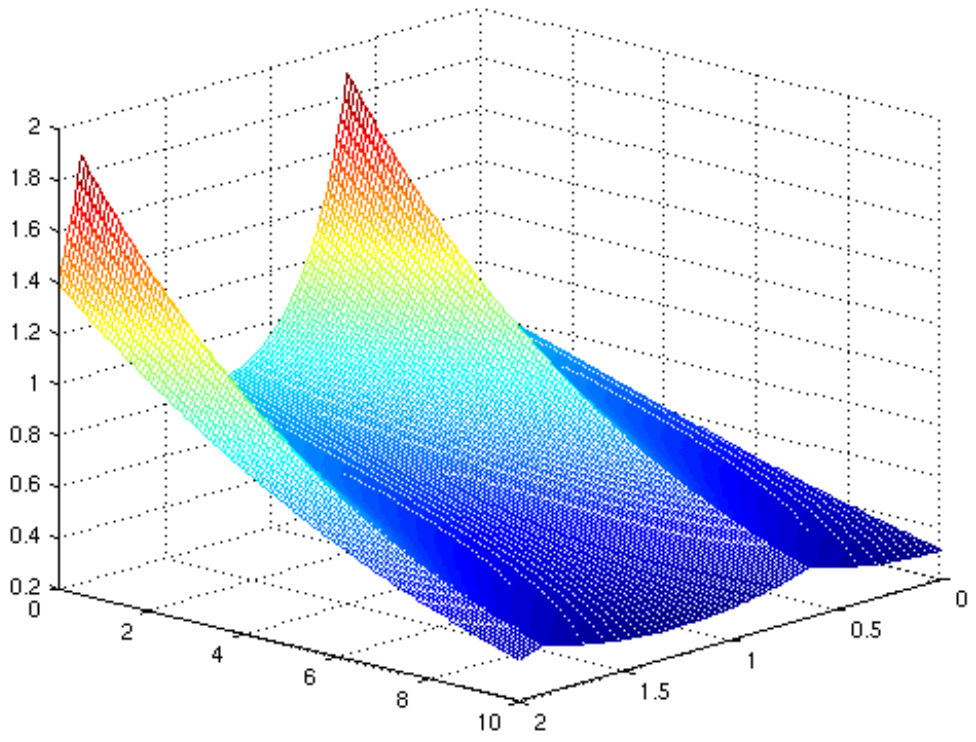

Figure 3.10: Solution of Fisher equation in 2D on the fine grid.

computation is expensive and incomplete, it shows that a minimum for the error exists outside the minimization path. One of the possible reasons is the gradient-based approach used in the minimization procedure. There is no guarantee that the minimum obtained for the residual is a global minimum of the functional.

In Figure 3.12, the computation uses SSOR smoothing. The first effect of this relaxation is a smaller number of minimization iterations. However, this modification of the algorithm does not necessarily provide the best minimized error.

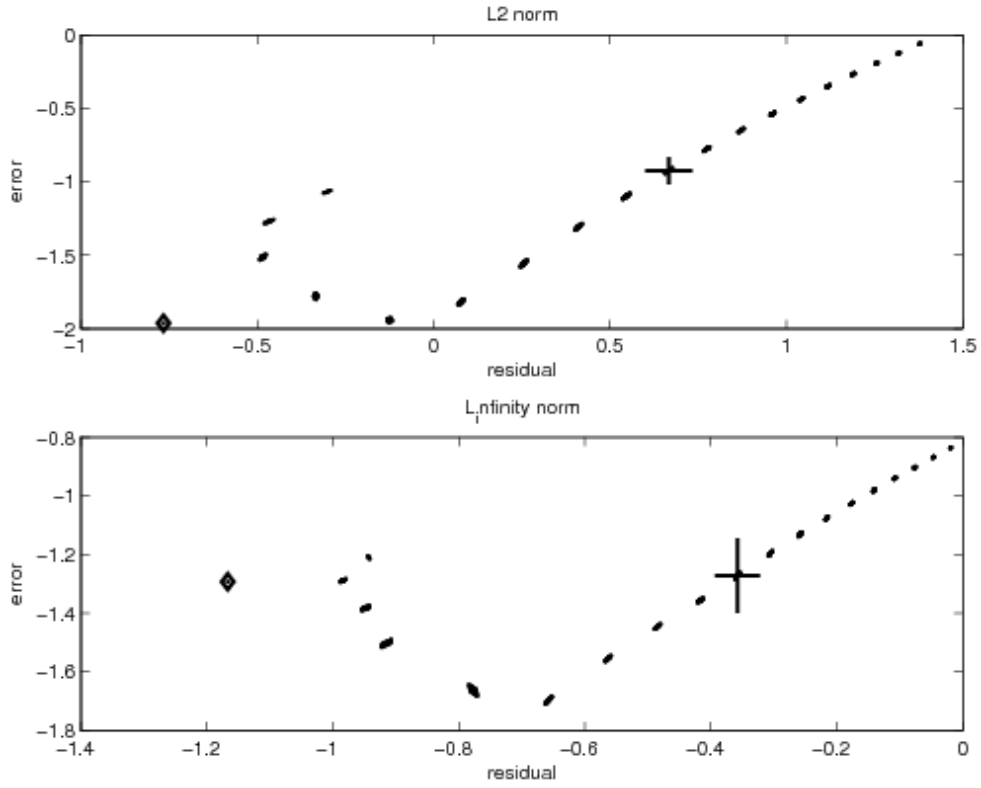

Figure 3.11: Error versus residual in a given norm for 2D Fisher equation without SSOR relaxation.

One of the drawbacks of the OES method in space and time that the higher dimension problem reveals is the computational cost. The next few points summarize these drawbacks and provide some indication for solutions:

1. Interpolation: each coarse grid solution interpolates to the fine grid in space and time. Mesh partitioning and look-up-table enable efficient space interpolation, and since time interpolation involves only a few points, it does not need a specific optimization.

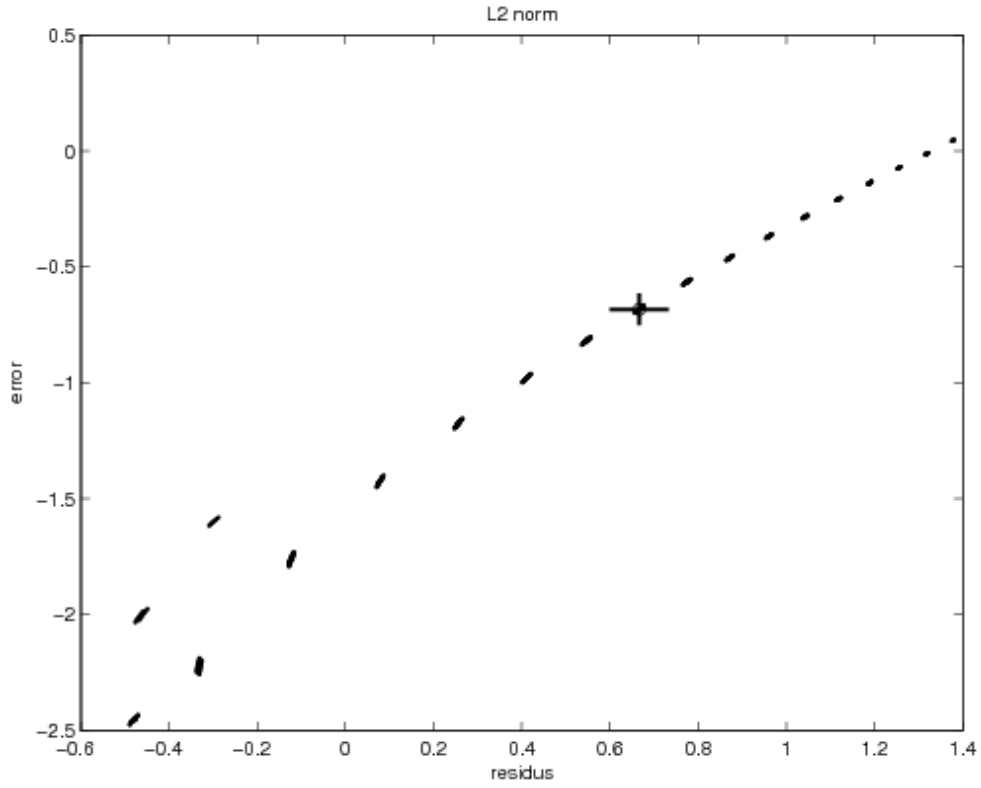

Figure 3.12: Error versus residual in a given norm for 2D Fisher equation with SSOR relaxation in L_2 norm.

2. SSOR: each computed solution requires smoothness on the fine grid. Parallel implementation could optimize the procedure.

3. Minimization: each step in the minimization procedure involved several space-time SSOR; therefore, the cumulative cost of all steps in the minimization can be prohibitive. As mentioned in the previous section, a potential approach is to perform minimization steps in parallel.

3.2.3.2 Cahn-Hilliard equation

The second equation we examined is the Cahn-Hilliard equation that describes phase separation, by which a binary fluid separates into different zones, each domain containing one specimen only. Given c the concentration of the fluid, with the extreme values $c = -1$ and $c = 1$ indicating the domain, the equation writes:

$$\frac{\partial c}{\partial t} = D\nabla^2(c^3 - c - \gamma c\nabla^2 c), \tag{3.71}$$

where D is the diffusion coefficient and $\sqrt{(}\gamma)$ is the length of the transition layer. In the simulation, homogeneous Neumann boundary conditions are used:

$$\frac{\partial c}{\partial n} = 0. \tag{3.72}$$

The initial condition is random, averaging to 0, on the coarsest OES grid (Equation 3.73), and interpolated on finer grids.

$$U = (rand - 0.5) * 0.01 \tag{3.73}$$

The interest in this equation rests in the higher order of the spatial derivatives as well as in the Neumann boundary conditions.

The solution of the equation converges toward a steady state in which the fluids are segregated.

Figure 3.13 shows the results of the simulation after convergence to the steady state. The simulation was done with 240 time steps with a time step size of $1.e-05$ for the fine grid, which is the referenced solution. The time step for the coarse grids, are respectively, $dT_1 = 4e-05$ and $dT_2 = 2e-05$.

Figure 3.14 represents one of the paths during the optimization process. Few smoothing steps of SSOR in space and time follow each interpolation. Without the few smoothing steps after the interpolation procedure, the method failed to converge to an optimum solution. This instability is due to the random initial condition.

We see again that the optimization procedure allows us to obtain a better error than RE in a long running simulation. The convergence toward a steady state of the PDE allows verifying the stability of OES for parabolic problem.

These results complement each other by giving consistent conclusions for the stability of the error estimation method for time-dependent problems.

3.3 Conclusion

In this chapter, we described a solution verification method for time-dependent problems that is more accurate than Richardson Extrapolation in the studied cases. Moreover, this method is stable for simulation, even when the solution has a sharp front. In order to improve the stability and convergence of the procedure, we added some smoothing such as the cost of computation time. While this is a drawback in terms of performance, it also offers an *a posteriori* estimation procedure for a wider set of time-dependent problems.

One of the strengths of this method is its simplicity. The procedure is built on a few existing solutions on different coarse grids in order to derive an *a posteriori* estimate. This part of the problem does not require having access to the internal

implementation of the solution procedure. The only possible limitation is the space-time SSOR smoothing that requires the operator of the PDE to be known in order to be implied.

As in the steady case, the minimum number of smoothing iterations is an open question. Further investigations should also be done for problems where the operator needed for the smoothing is not available. To make the procedure more computationally effective, optimization methods with embarrassing parallelism should also be investigated. Genetic algorithms seem to offer a good potential for later expansions of the method since the optimization is already fairly expensive.

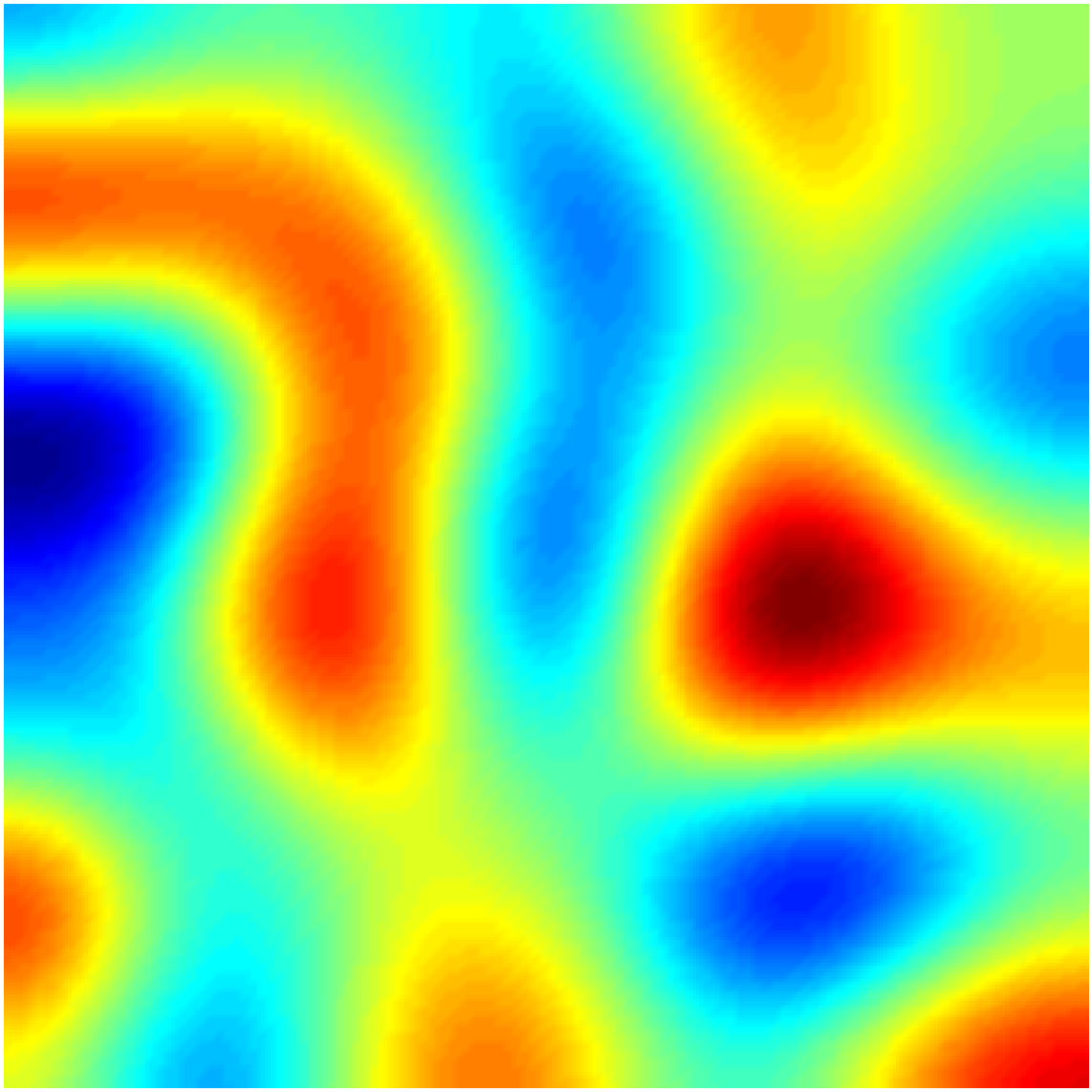

Figure 3.13: Solution of the Cahn-Hilliard equation.

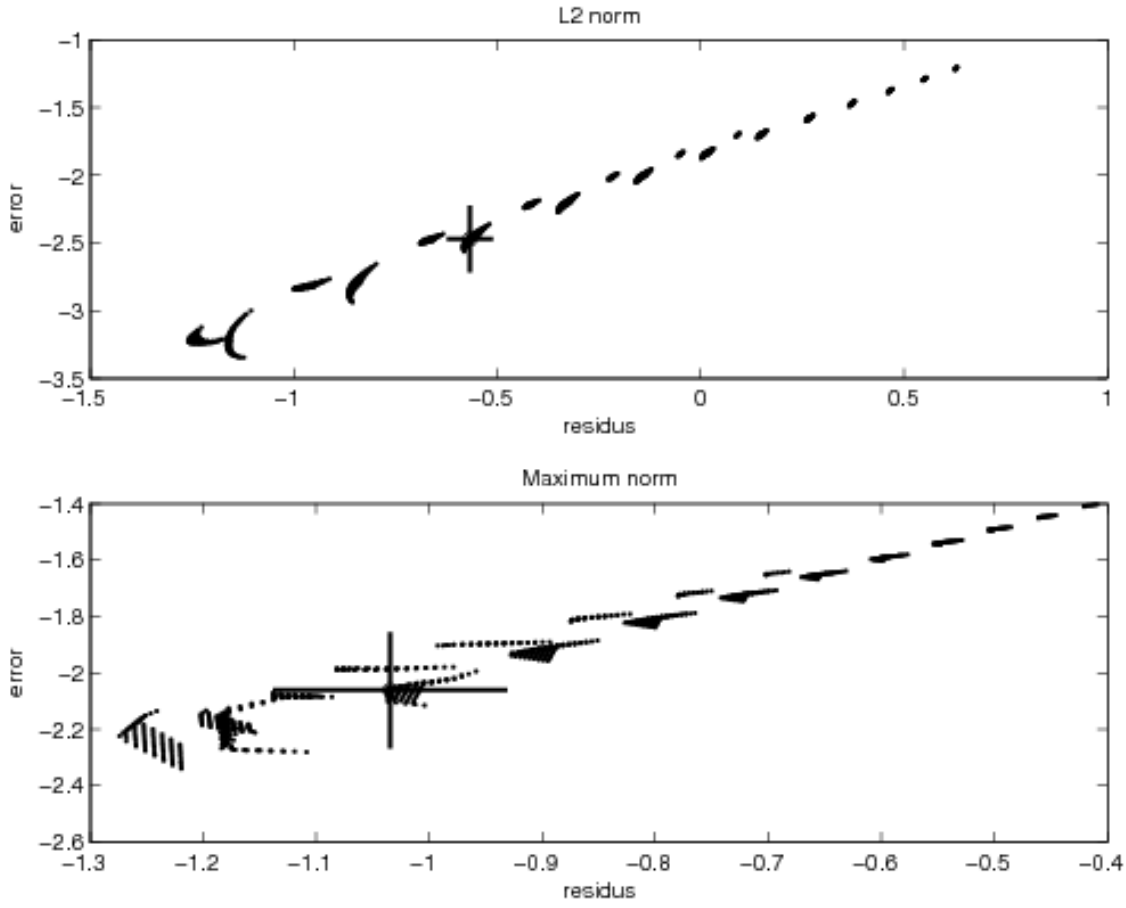

Figure 3.14: Optimization path for the Cahn-Hilliard equation with SSOR relaxation.

Chapter 4

Optimized extrapolation method for closed source applications

The goal of this chapter is to present a postprocessing software infrastructure that connects to any existing numerical simulation software, for example, to widely used commercial applications such as ADINA, Ansys, Fluent, Numeca, Star-CD, etc; in addition, we provide an *a posteriori* error estimate to their simulation.

The standard approach in applied mathematics to handle the problem of solution verification is to work on the approximation theory of the PDE. For each specific PDE problem, the right finite element (FE) approximation may provide the correct *a posteriori* error estimate [68]. Unfortunately, this approach may require a complete rewriting of an existing Computational Fluid Dynamic (CFD) application based on finite volume (FV), for example, and lack generality. Usually *a posteriori* estimates fail if the (non-linear) PDE solution is stiff or if the grid resolution is not adequate.

Since grid refinement itself requires an *a posteriori* estimate, this leads to an obvious problem. Large Reynolds number flows are common in many applications, including turbulence problems. For these applications, rigorous solution verification may not be achievable, given state-of-the-art numerical analysis. The difficulty of solution verification is even greater for complex multi-physic coupling. The general practice in scientific computing is to simulate PDEs, for which neither applied mathematics nor numerical analysis guarantee the result.

One of the difficulties in applied mathematics is the time lag between the development of rigorous mathematical tools and the common scientific computing practice. Therefore, our goal should be to improve existing solution verification tools such as the convergence index of Roache *et al.* [5, 69] , and the Richardson Extrapolation (RE) technique presented in Chapter 1, with something potentially more elaborate and reliable that can take advantage of existing *a posteriori* estimates when they are available. It could positively affect the daily use of the verification tools by practitioners.

Our method for *a posteriori* error estimate relies on four main ideas:

- *Reformulation of the problem :* a traditional approach for solving error estimation problems is to use a built-in procedure that utilizes information on the PDEs. The output of the *a posteriori* error estimate serves as a feedback parameter to improve the solution procedure, by using mesh refinement, for example. The method suggested in this chapter solves the error estimation problem by integrating the family of solutions generated by the code into an

optimum design framework completely independent of the code itself. Our *a posteriori* procedure constructs a new "optimal" solution from a set of two to three numerical solutions computed by the existing code.

- *Independence from the approximation theory framework of the PDE:* in most of the cases with commercial codes, detailed knowledge of the approximation theory framework used to represent the PDE's and compute the numerical solutions is at best limited and in the worst case completely inaccessible. Our new technique allows the processing of the underlying set of discrete (non)-linear equations without using *a priori* information on the approximation theory framework that is applied to solve the PDE.

- *Integration of existing a posteriori error estimates :* industrial and research laboratories' simulation tools might already make use of some form of error estimates. If the practice proves that they are efficient on a given class of PDE problems, our optimization framework should enable the use of these estimates.

The following two sections first describe the general method and then give some examples with two benchmark problems.

4.1 Optimized Extrapolation Method

We describe in the following the main ideas without seeking an exact formal mathematical description of a given specific PDE problem.

Let us consider a boundary value problem (Ω is a polygonal domain and $n =$

2 or 3) :

$$N[U(x)] = S(x),\ x \in \Omega \subset I\!R^n. \tag{4.1}$$

We assume that this Equation 4.1 defines a well-posed problem and has a unique smooth solution. We have restricted ourselves in this section to a steady problem, i.e., there is no time dependency. Let $(E, ||.||_E)$ and $(F, ||.||_F)$ be two normed linear spaces, and $N_h : E \rightarrow F$ be the operator corresponding to the problem solved by the code. It can be a finite volume approximation of Equation 4.1 on a family of meshes $M(h)$ parametrized by $h > 0$ a small parameter. In practice we look for an approximation of the accuracy of the solution U_h on the mesh $M(h)$ produced by the code $\mathcal{C}$ that operates on the data S_h:

$$\mathcal{C} :\ S_h\ \rightarrow U_h. \tag{4.2}$$

The smaller h, the finer should be the discretization.

Let p_h denote the projection of the continuous solution U onto the mesh $M(h)$. We assume *a priori*:

$$||U_h - p_h(U)||_E \rightarrow 0,\ \text{as } h \rightarrow 0. \tag{4.3}$$

We assume, therefore, that the code satisfies the criteria of software verification, and that the convergence to the exact continuous solution is satisfied.

The objective is to verify the solution produced by the code, not the code itself. We will get an error estimate versus a very fine grid solution U_∞ that is never computed, because the cost is prohibitive. We will skip the index h when it is not

essential. The spaces $E,\ F$ have in practice (very large) finite dimensions when they are for the discrete solutions on $M(h_\infty)$, and discrete data S_{h_∞}.

We assume that the code $\mathcal{C}$ has a procedure that provides the residual, i.e., $V \to \rho = N(U_h) - N(V)$, where $V \in E,\ \rho \in F$. We note that most commercial code offers this feature or provides a (first order explicit) time stepping procedure:

$$\frac{U_h^{n+1} - U_h^n}{dt} = N(U_h^n) - S. \tag{4.4}$$

The residual is then $\rho = \frac{U_h^1 - U_h}{dt}$. We assume that the following problem

$$N(u) \ = \ s,\ \forall s \in B(S,d) \tag{4.5}$$

is well-posed for $s \in B(S,r)$, where B is a ball of center S and radius r in $(F, ||.||_F)$. There should exist a unique solution for all data in $B(S,r)$ and the dependency of the solution on these data is supposed to be smooth enough to use a second order Taylor expansion.

Let us suppose that $N(U_h) \in B(S,r)$, that is

$$||\rho||_F = ||N(U_h) - S||_F < d. \tag{4.6}$$

We would like to get an error estimate on $e = U_h - U_\infty = \mathcal{C}(S+\rho) - \mathcal{C}(S)$. A Taylor expansion writes

$$\begin{aligned} \mathcal{C}(S) = \mathcal{C}(S+\rho) &- (\rho \cdot \nabla_s)\mathcal{C}(S+\rho) \\ &+ \frac{1}{2}\rho \cdot [\rho \cdot R(S)], \end{aligned} \tag{4.7}$$

$$\text{where } ||R(S)||_E \leq K = \sup_{s \in B(S,d)} ||\nabla_s^2 \mathcal{C}(s)||_E.$$

Therefore,

$$||e||_E \ \leq \ ||\rho||_F \ (||\nabla_s \mathcal{C}(S+\rho)||_E \ + \ \frac{K}{2}||\rho||_F). \tag{4.8}$$

This completely general error estimate points out two different tasks:

• Task 1: compute an accurate upper bound on $||\nabla_s \mathcal{C}(S+\rho)||$; and

• Task 2: obtain a solution $U_\infty + e$ that gives a residual $||\rho||$ small enough to make the estimate useful, i.e., compatible with Equation 4.6.

Task 2 is the purpose of the Optimized Extrapolation Solution (OES) method, while Task 1 can be achieved by a sensitivity analysis of $\mathcal{C}$.

4.1.1 Task 1: Stability Estimate

Let $\{b_i^E, \ i = 1..N\}$ be a basis of E_h, $\{b_i^F, \ i = 1..N\}$ be a basis of F_h and $\varepsilon \in I\!R$ such that $\varepsilon = o(1)$. Let $(V_i^{\mp})_{i=1..N}$, be the family of solutions of the following problems:

$$N(U_h \mp \varepsilon V_i) = S + \rho \mp \varepsilon b_i. \tag{4.9}$$

We get from finite differences the approximation

$$\begin{aligned} C_{h_\infty} &= ||\nabla_s \mathcal{C}(S+\rho)|| \\ &\approx ||(\frac{1}{2}(V_j^+ - V_j^-))_{j=1..N}|| + O(\varepsilon^2). \end{aligned} \tag{4.10}$$

We can get in a similar manner an approximation of the norm of the Hessian $\nabla_s^2 \mathcal{C}(S+\rho)$. For ρ small enough, we can verify that the upper bound is given at first approximation by:

$$||e||_E \ \leq \ C_{h_\infty}||\rho||_F. \tag{4.11}$$

The column vectors $V_j^{\mp}$ can be computed with embarrassing parallelism. It is, however, unrealistic to compute these solutions that lie on the fine grid $M(h_\infty)$.

To make this task manageable, we have to reduce the dimension of the problem. We use the following two observations: while the solution of a CFD problem can be very much grid-dependent, the conditioning number of the problem is in general much less sensitive to the grid. The idea is then to compute an approximation of C_{h_∞} by extrapolation from an estimate of two or three coarse grid computations of C_{h_j}. Further, let us assume that the fine grid $M(h_\infty)$ is a regular Cartesian grid. The number of terms to accurately represent the projected solution $\tilde{U}_j,\ j = 1..3$ with a spectral expansion or a wavelet approximation at a given accuracy is much smaller than the dimension of the coarse grid used in a finite element/finite volume computation. We propose to use preferably a grid $M_{h\infty}$ that has enough regularity to allow a representation of the solution U_∞ with some form of reduced representation, using either trigonometric expansion or wavelets.

The grid M_{h_∞} may have many more grid points than necessary. Therefore, it might not be computationally efficient to perform a true fine grid computation, but we do not have to do this computation anyway.

Further, regular grids are far more easy to construct. If for some reasons $M_{h\infty}$ has to be unstructured, we can also use spectral elements. An ideal method might be to use a proper orthogonal decomposition that captures the main feature of the solution [70].

Let us denote $\hat{E}$ and $\hat{F}$ to be the spaces corresponding to one of these compact

representations of the solution and residual. Let $(\hat{b}_j^{E/F}, j = 1 \dots \hat{N})$ be the corresponding base with $\hat{N} \ll N$. Let $\hat{q}_{E/F}$ be a mapping $E/F \to \hat{E}/\hat{F}$, and $q_{\hat{E}/\hat{F}}$ be a mapping $\hat{E}/\hat{F} \to E/F$, and let $\hat{\mathcal{C}} : \hat{S}_h \to \hat{U}_h$ be the code that uses this postprocessing of the residual and solution.

Figure 4.1 illustrates the relation between the different discrete spaces, along with the corresponding mappings. The mapping $\hat{q}_E$ can be a least square approximation of the solution u into $\hat{E}$, acting as a filter on the solution, while $q_{\hat{E}}$ is a projection onto E.

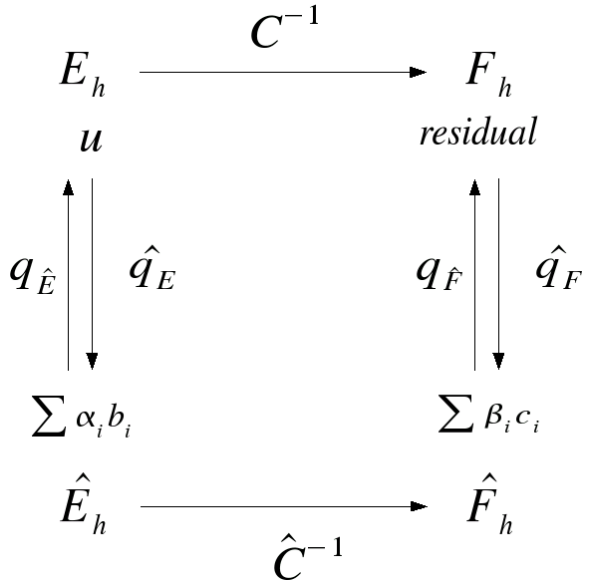

Figure 4.1: Mappings and vector spaces.

The construction of q_E and $\hat{q}_E$ on one side, and the construction of q_F and $\hat{q}_F$ on the other side do not consider the nature of the true approximation space used in the code $\mathcal{C}$ since the implementation details are most of the time unavailable: the

mappings involve only the discrete representations of the functions.

To summarize the procedure for Task 1, the estimate on $\mathcal{C}_{h_\infty}$ will be applied to verify the code $\hat{\mathcal{C}}$ based on the computation of $(\hat{V}_j^{\mp}, j = 1..\hat{N})$ vectors on the coarse grids $M(h_j),\ j = 1..3$ done by the code $\hat{\mathcal{C}}$. We notice that the computation of the vector $\hat{V}_j^{\mp}$ can be done with embarrassing parallelism. Further, because ε is small, the code $\hat{\mathcal{C}}$ can use, as an initial guess in its iterative process, the solution U_h, which is hopefully very close to the unknown $\hat{U}_h \pm \epsilon \hat{V}_j^{\mp}$.

Several issues need to be carefully investigated when applying this procedure. We will mention two of them. First, our method assumes that the spectral properties of the operator follow some fairly regular asymptotic properties for the coarse meshes under consideration, as $h \to 0$. This hypothesis should be verified and/or may not be true. Second, ε must be chosen carefully as a function of the mesh size, in order to avoid a dramatic inaccuracy on the stability estimate of C_{h_∞}. We recall, however, that we are only looking for an order of magnitude of C_{h_∞} and not its true value.

Let us discuss our second task, which is to compute a solution on the fine grid that is good enough to recover an error estimate.

4.1.2 Task 2: Optimized Extrapolation

Let $M(h_1)$ and $M(h_2)$ be two different meshes used to build two approximations U_1 and U_2 of the PDE problem expressed in Equation 4.1.

A linear extrapolation formula that combines these two solutions has the form

$$\alpha U_1 + \beta U_2; \tag{4.12}$$

since $\lim_{h_1 \to 0} U_1 = U^\infty$ and $\lim_{h_2 \to 0} U_2 = U^\infty$, a consistent linear extrapolation requires to have $\lim_{h \to 0} \alpha U_1 + \beta U_2 = U^\infty$. This adds a constraint on α and β which is: $\alpha + \beta = 1$.

Therefore, a consistent linear extrapolation formula should have the form

$$\alpha U_1 + (1 - \alpha) U_2, \tag{4.13}$$

where α is a weight function. We define our optimized extrapolation problem as follows:

(P_α)**:** *Find* $\alpha \in \Lambda(\Omega) \subset L_\infty$ *such that* $G(\alpha U_1 + (1 - \alpha) U_2)$ *is minimum.*

The OES, if it exists, is denoted as $V_e = \alpha U_1 + (1 - \alpha) U_2$.

For computational efficiency, $\Lambda(\Omega)$ should be a finite vector space of very small dimension. The objective function G might be derived from any existing *a posteriori* error estimates, depending on their availability. Our ambition is to provide a numerical estimate on $||U_j - U_\infty||,\ j = 1, 2$, without computing U_∞ explicitly. The solution U_j can then be verified, assuming Equation 4.3.

The fine mesh $M(h_\infty)$ should be set such that it captures all the scales of the continuous solution with the level of accuracy required by the application. We have *a priori* $h_\infty << h_1,\ h_2$. Both coarse grid solutions U_1 and U_2 must be projected onto $M(h_\infty)$. We will denote $\tilde{U}_1$ and $\tilde{U}_2$ as the corresponding functions. In this section,

we choose to minimize the consistency error for the numerical approximation of Equation 4.1 on a fine mesh $M(h_\infty)$. According to the stability estimate Equation 4.11 established in the previous section, this is a rational choice for the general case, where no mathematically rigorous *a posteriori* estimate is known. The objective function is then

$$G(U^\alpha) = ||N_{h_\infty}\, U^\alpha - F_{h_\infty}||, \tag{4.14}$$

where $U^\alpha = \alpha\, \tilde{U}_1 \; + \; (1-\alpha)\, \tilde{U}_2$.

Whenever $U_1 - U_2 << U - U_2$ in some set of non-zero measure, the weight function in the extrapolation formula should be ill-defined. For example, if at some spatial point, $U_1 = U_2$, then it is not possible to evaluate α locally from the extrapolation formula. We consider these local values to be outliers of the optimization problem. A potentially more robust approximation procedure makes use of a third approximation U_3. We formulate the problem with three grids as follows:

$(P_{\alpha,\beta})$**:** *Find* $\alpha, \beta \in \Lambda(\Omega) \subset L_\infty$ *such that* $G(\alpha U_1 + \beta U_2 + (1-\alpha-\beta)U_3)$ *is minimum.*

If the three approximations $U_j, j = 1 \ldots 3$ coincide at the same spatial point, the local accuracy cannot be improved using an extrapolation method only. It might be that all three solutions have fully converged or are all erroneous. We will still get a *global* upper bound on the error with our method, providing that the residual is small enough for Equation 4.6 to hold true.

To reduce the dimension of this problem we search for the unknown weight functions in a small space for which basis functions are trigonometric expansion, wavelet expansion, or possibly spectral elements. If Ω is the physical domain for the CFD solution, the unknown weight function can be extended to a square domain $(a,b)^n$, such that $\Omega \subset (a,b)^n$. As a matter of fact, no boundary conditions are required for the unknown weight functions. Let $\{\theta_j,\ j = 1 \ldots m\}$ be the set of basis functions of $\Lambda(\Omega)$. We look for the solution of the optimization problem as follows

(Π_{α}^{m})**:** *Find* $(\alpha_j)_j \in I\!R^m$*, such that*

$$||G([\sum_{j=1..m} \alpha_j \Theta_j]\tilde{U_1} + [1 - \sum_{j=1..m} \alpha_j \Theta_j]\tilde{U_2})||_F \tag{4.15}$$

is minimum.

We have a similar formulation for the three-level OES described in [46, 47], which is:

$(\Pi_{\alpha,\beta}^{m})$**:** *Find* $(\alpha_j)_j(\beta_j)_j \in I\!R^m$*, such that*

$$||G([\sum_{j=1..m} \alpha_j \Theta_j]\tilde{U_1} + \sum_{j=1..m} \beta_j \Theta_j \tilde{U_2} + \sum_{j=1..m} [1 - \alpha_j - \beta_j]\Theta_j \tilde{U_3})||_F \tag{4.16}$$

is minimum .

Further, we need a filtering process of the solution to have this minimization procedure be numerically efficient. In practice the interpolation of the coarse grid solution to the fine grid M_{h_∞} introduces spurious high order oscillations that may make the optimization process given by Equation 4.15 unreliable. The postprocessing $\hat{q}_F$ regularized the problem. We can easily obtain the result when the weight function

is a scalar function. To make this computation robust we use a surface response methodology [71] that is rather trivial in the scalar case. This procedure consists of computing a lower order polynomial, best fit of the functional $||G(\alpha\tilde{U}_1 + (1-\alpha)\tilde{U}_2)||$, by sampling α according to the expected convergence order range of the code. The minimization of α is then done on this polynomial approximation by a standard method. The sampling process is a cumbersome embarrassing parallel process that can take advantage of a computational grid [72].

It is, however, impractical when the dimension of the problem for the α search is more than a few units. One may use a combination of a genetic algorithm and a local optimization search to solve the non-linear optimization problem (Equation 4.15). There is extensive knowledge and a set of optimization software [73] available that we can indeed reuse. We have observed in our experiments that if the solution provided by the CFD code is very coarse, the use of space-dependent weight functions might not be required. Similarly, if the solution is very accurate and the code has uniform convergence, we do not need space-dependent weight function. However, in the case of stiff problems we would like to get some adaptivity into the construction of the weight function. This is an open problem that we are currently working on [39].

4.2 Algorithm for optimized extrapolation method

4.2.1 Algorithm

The algorithm of our method written in its simplest version is as follows:

1. *Call coarse Mesh*: generate the (coarse) meshes $M(h_1)$ and $M(h_2)$. If h_i is the average space step for the grid $M(h_i)$ we should have $h_2 < h_1$ but this is not necessary.

2. *Call fine Mesh*: generate a fine mesh $M(h_\infty)$ that is supposed to solve all the scales of the problem. $M(h_\infty)$ is preferably a structured mesh. We must have $h_\infty << h_1, h_2$.

3. *Call Solver*: solve the problem on $M(h_1)$ and $M(h_2)$, possibly in parallel.

4. *Call Projection*: project the coarse solutions U_1 and U_2 onto $M(h_\infty)$ and post-process them to avoid spurious oscillations due to the interpolating function.

5. *Solve minimization problem*: we can create, for example, sample solutions $U_\alpha = [\alpha\tilde{U}_1 + (1-\alpha)\tilde{U}_2]$ and/or use an off-the-shelf optimization package.

6. *Get Stability Constant*: compute in parallel an increasing set of perturbed solutions $U_h \pm \varepsilon V_i^{\pm}$ until eventual convergence of the stability estimate.

4.2.2 Applications

We have selected two test cases, one in fluid dynamics, and the other in heat transfer. Further details on these numerical simulations can be found in [74].

We will carry through our method starting from two coarse grid calculations only. The minimization problem is the following:

(P_α)**:** *Find* $\alpha \in I\!R$ *such that* $G(\alpha U_1 + (1-\alpha)U_2)$ *is minimum.*

For the sake of simplicity, we restrict ourselves to α being a constant function on the domain ($\alpha \in I\!R$). While there is no technical difficulty in considering $\alpha = \sum_{i=1}^{M} \alpha_1 h_i$, M being small with respect to the number of points on the fine grid, it did not appear necessary to have α space-dependent to obtain satisfactory results in this experimental study.

The finite element commercial package used to solve both test cases is ADINA. We could have used another commercial package for this demonstration. ADINA is a comprehensive finite element software that enables analysis of structures, fluid simulations, and fluid flow simulations with structural interactions. http://www.adina.com/ provides more information on this software.

4.2.2.1 Backward facing step flow

We are going to consider first a steady incompressible viscous flow in the backward facing step configuration.

Equation 4.17 models the incompressible flow:

$$\begin{cases} \frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}.\nabla\mathbf{u} + \nabla\mathbf{p} &= \nu\Delta\mathbf{u} \\ \nabla\mathbf{u} &= 0 \end{cases} \tag{4.17}$$

where ν is the viscosity of the fluid.

Figures 4.2 and 4.3 show an example of an unstructured coarse mesh used for the calculation of the backward-facing step flow at Reynolds number 500 and the contour of the solution field.

The length of the cavity is $L_x = 10$, the width is $L_y = 2$, and the size of the step

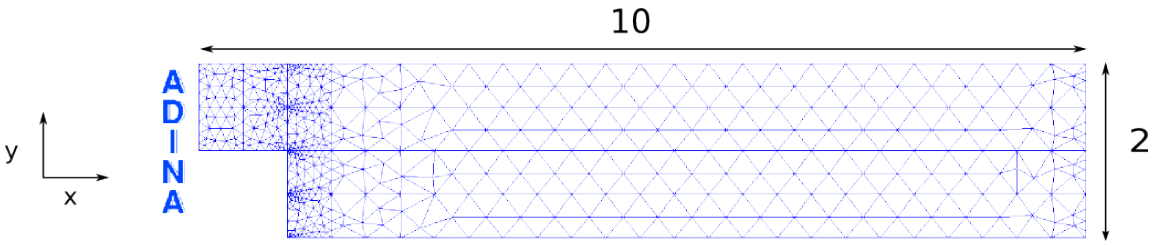

Figure 4.2: Coarse unstructured mesh for the backward-facing step test case generated by Adina.

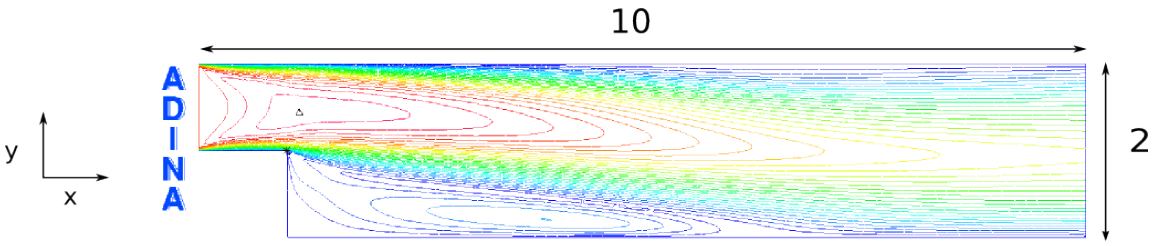

Figure 4.3: Contour of the velocity field for the backward facing step produced by Adina.

is 1. The inflow is set to be a Poiseuille flow with maximum velocity 1 *U.I.*. The outflow boundary condition is free. We assume no slip boundary conditions on the walls. In this simulation, the number of quad elements are, respectively, 10347 on the fine grid G^{∞}, 1260 on the coarse grid G_1, and 2630 on the coarse grid G_2.

To provide a rigorous measurement of the error we construct a nearby manufactured solution [75] that is within 1/1000 of a very fine grid solution. This procedure forces the right-hand side and boundary conditions of the backstep problem in such a way that we get an exact solution to Navier-Stokes that is very close to the solution of the backstep problem. We obtain then an exact measurement of the numerical

error for this manufactured solution that is $\mathbf{F}_s$. Let (u_{ms}, p_{ms}) be the exact analytical solution of the following problem:

$$\begin{cases} \frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}.\nabla\mathbf{u} + \nabla p &= \nu\Delta\mathbf{u} + \mathbf{F_s} \\ \nabla\mathbf{u} &= 0 \\ u &= \Phi \text{ on } \partial\Omega. \end{cases} \tag{4.18}$$

Assume that the approximation of the numerical solution has the following form:

$$\begin{cases} u^{ms} &= \sum_{i=0}^{N_1} \alpha_i \mathbf{u}_i \\ p^{ms} &= \sum_{i=0}^{N_2} \beta_i p_i \quad , \\ \nabla\mathbf{u}_i &= 0 \end{cases} \tag{4.19}$$

where $\alpha_i, \beta_i \in I\!R$.

Then the set of basis functions

$$\mathbf{u_{k_1}} = \begin{cases} cos(k_1\pi\frac{x}{L_x})sin(k_1\pi\frac{y}{L_y}) \\ -sin(k_1\pi\frac{x}{L_x})cos(k_1\pi\frac{y}{L_y}) \end{cases} \tag{4.20}$$

allows the construction of a manufactured solution that satisfies the divergence free condition and

$$p_{k_2} = cos(k_2\pi\frac{x}{L_x})cos(k_2\pi\frac{y}{L_y}). \tag{4.21}$$

$\hat{E}$ is spanned by the basis functions Equation 4.20 while we use standard trigonometric expansions for $\hat{F}$. The pressure is a Lagrange multiplier in the finite element formulation of ADINA. The different figures and results show an error estimate for the velocity field only. The error is computed as the maximum of the L_2 norm for each component.

First, the flow field variables are interpolated from the fine mesh finite elements solution onto a regular grid of Ω that is globally about the same mesh size to ease the fitting with the trigonometric functions of Equation 4.20. The discrete functions obtained on this grid are noted $(\tilde{U}_{FE}, \tilde{p}_{FE})$. Then, the weight coefficients α_i and β_i are computed using least square interpolation on the regular grid functions $(\tilde{U}_{FE}, \tilde{p}_{FE})$. Ω is a subset of the rectangle $(0, L_x) \times (0, L_y)$ and the numerical convergence is not granted: we need to verify *a priori* if this interpolated solution is giving an accurate approximation of the numerical solution.

The discrete error for the backward-facing step flow between the numerical solution and its trigonometric approximation is shown in Figure 4.4. Let $M(h_j), j = 1 \ldots N$ be the nodes of the mesh. We use the standard discrete vector norm $||.||_1$, $||.||_2$,$||.||_\infty$ to measure the numerical error on the trace of the numerical solution at these points. To be more specific

$$||(f_j)_{j=1..M}||_1 = \frac{1}{M} \sum_j |f_j|, \tag{4.22}$$

$$||(f_j)_{j=1..M}||_2 = \sqrt{\frac{1}{M} \sum_j (f_j)^2}, \ and \tag{4.23}$$

$$||(f_j)_{j=1..M}||_\infty = \max_j |f_j|. \tag{4.24}$$

These norms are unrelated to the finite element space used in the ADINA calculation. Our verification procedure follows the spirit of an experimental measurement where one may get information at specific locations.

Figure 4.5 shows the evolution of the error in L_2 norm as a function of the number N_1 and N_2 of basis functions. We choose $N_1 = 38$ for the velocity and $N_2 = 33$ for

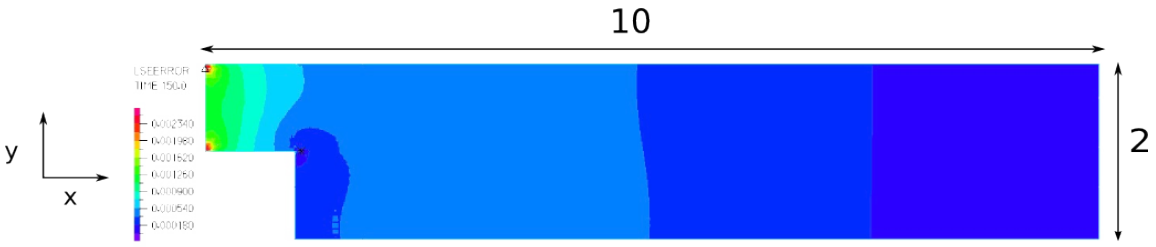

Figure 4.4: Difference between the NS FE solution and the nearby manufactured solution.

the pressure, giving some small differences with $(\tilde{U}_{FE}, \tilde{p}_{FE})$. Specifically we have

$$||\tilde{U}_{FE} - u^{ms}||_2 = 1.76e^{-04},\ ||\tilde{U}_{FE} - U^{ms}||_\infty = 2.12e^{-03}\ and \tag{4.25}$$

$$\frac{||\tilde{p}_{FE} - p^{ms}||_2}{||\tilde{p}_{FE}||_2} = 2.25e^{-03}. \tag{4.26}$$

The expression of F_s is computed symbolically using Maple for each element of the basis and its expression is injected into ADINA as a forcing term in the momentum equation. We have now a manufactured solution for the backstep that is near by the numerical solution obtained with the finite elements code on a very fine mesh. Let us compute the coarse grid solution of the corresponding new Navier-Stokes problem with the right-hand side F_s, and construct the *a posteriori* error estimate. The only motivation to work with this new NS problem is to provide exact numerical accuracy.

The next step is to proceed with the *a posteriori* estimate calculation using the algorithm from Section 4.2.1. Two coarse mesh solutions of problem given by the Equation 4.18 are computed with, respectively, 1506 and 2630 elements. ADINA generates both meshes automatically and they are not coincident. Both solutions

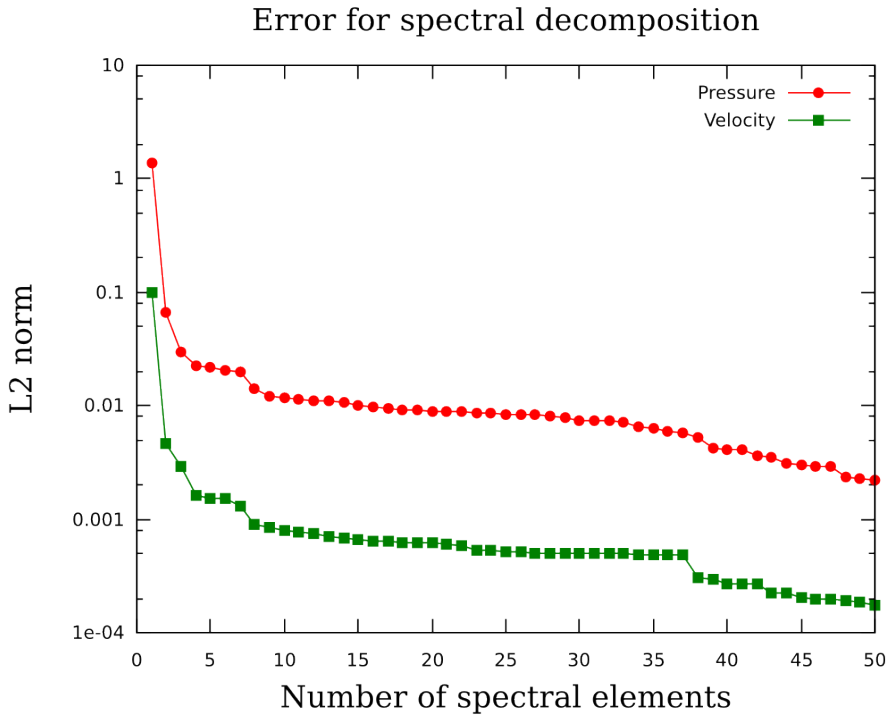

Figure 4.5: Numerical error in L_2 norm as a function of the number of trigonometric basis functions of the nearby manufactured solution.

are interpolated to the fine mesh $M(h_\infty)$ with a P1 method. Each solution is post-processed on the fine grid with few explicit time steps reusing ADINA on the fine grid. The implementation converts this explicit time stepping into a relaxation procedure to damp all the spurious high frequencies of the solution introduced by the P1 interpolation. Its efficiency decreases as the Reynolds number increases. Practically, the residual decreases extremely fast during the first relaxations, which get rid of the perturbation introduced by the P1 interpolation. After this initial stage, the rate of convergence slows down significantly. The turning point in the convergence behavior setup our stop criterion. A small number of explicit time steps by no means provide

a convergence to the fine mesh solution. In our test case we have 10 to 30 time steps. To avoid this time stepping, one might use a high order interpolation procedure on the coarse meshes solutions, such as BSpline, to impose the smoothness of the solution on the fine mesh. This is in practice far more expensive than the relaxation steps and increases the complexity of the code.

Based on these two coarse grid solutions postprocessed on $M(h_\infty)$, the optimized extrapolated solution $\alpha\tilde{U}_1+(1-\alpha)\tilde{U}_2$ is built. The weight α are scalar functions. The expected order of convergence is between 0 and 3, hence the search of α is limited to $\Lambda = (-0.2\ ,\ 2.2)$.

Figure 4.6 shows the evolution of the true error as a function of the residual in the L_2 norm for α varying in Λ. One can observe that there is no need to postprocess every linear combination $\alpha\tilde{U}_1 + (1-\alpha)\tilde{U}_2$ with further explicit time stepping. Figure 4.7 shows that minimizing the error is reasonably achieved by minimizing the residual for both the L_1 and L_2 norms. This correspondence is less well satisfied in the L_∞ norm, as shown by Figure 4.8. We speculate that this discrepancy comes from the fact that the NS solution is not smooth along the backstep wall at the corners.

It also appears that the optimal combination α is weakly dependent on the choice of the discrete norm used.

The computation of the stability estimate described in Section 4.1.1 uses the

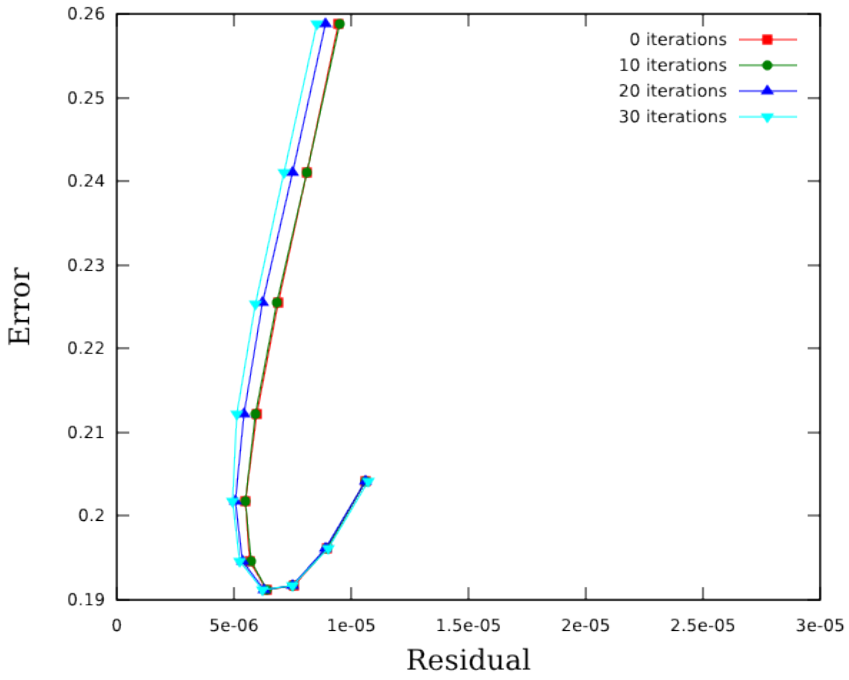

Figure 4.6: Dependence of the Error and Residual in L_2 norm on the number of relaxations used in postprocessing $\alpha\tilde{U}_1 + (1-\alpha)\tilde{U}_2$.

following set of basis functions $b_i(x/L_x)b_j(y/L_y)$, where

$$b_0(z) = 1, \tag{4.27}$$

$$b_1(z) = cos(\pi z),$$

$$b_2(z) = sin(\pi z),$$

$$b_3(z) = sin(2\pi z), \ldots$$

$$b_j(z) = sin((j-1)\pi z), \ldots$$

The computation of the stability estimates and condition numbers employs both coarse grids. The amplitude of the perturbation ϵ is set to 0.01. We observe that

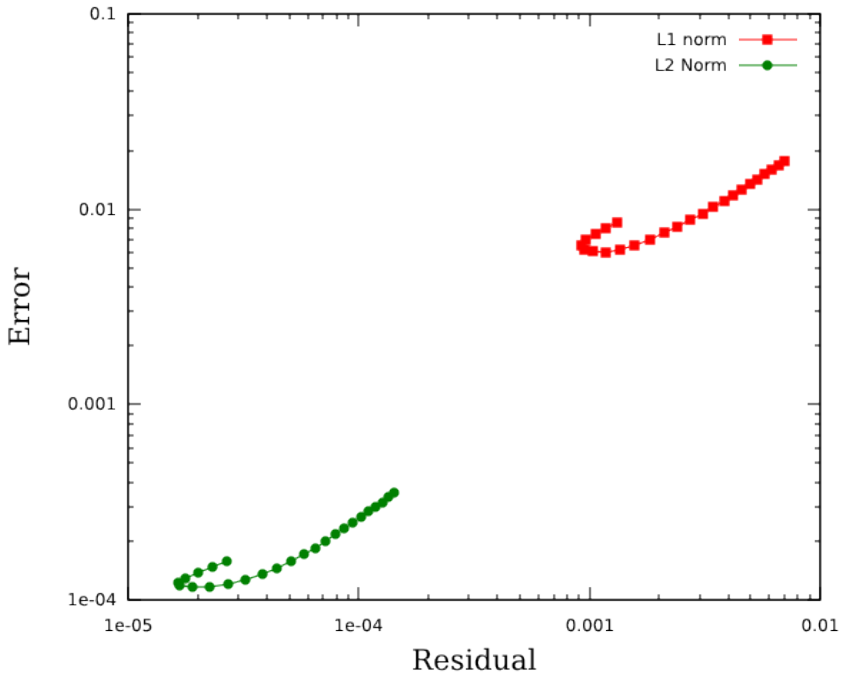

Figure 4.7: Comparison of OES solutions for different discrete norms.

larger values such as $\epsilon = 0.1$ lead to oscillation in the computation of the stability constant. Smaller values of ϵ than 0.01 do not provide a sufficient amplitude for the perturbation to have a significant impact on the computed solution.

Because the initial two coarse grid calculations are the only component of the sensitivity analysis, starting from the unperturbed coarse grid calculation requires very few time steps. From the two condition numbers corresponding to each coarse grid solution it is possible to extrapolate the stability constant for the Navier-Stokes calculations on the fine grid mesh M_∞.

Figure 4.9 shows the evolution of the stability constant as the number of basis

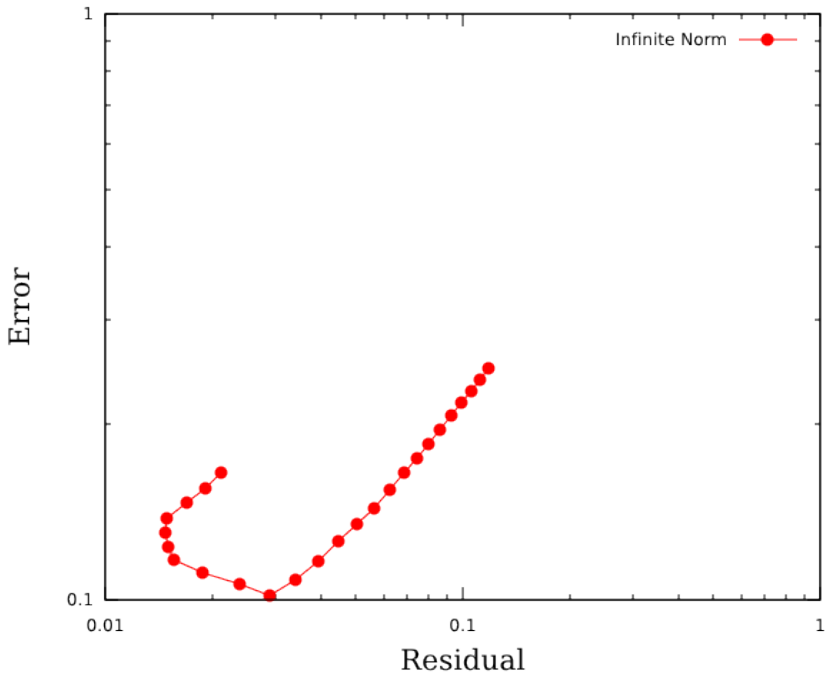

Figure 4.8: OES solution with the L_∞ norm.

functions used increases. These results seem to show that only a limited number of basis functions are needed to represent the numerical solution via the transformation $\hat{q}_F$ defined in Figure 4.1.

Figure 4.10 gives the error for the OES versus the fine grid solution in the L_2 norm computed in a reduced space generated by trigonometric functions specified by Equation 4.20. The low horizontal line is the true error in the L_2 norm for the OES obtained previously. One can observe that the extrapolation of the stability constant, which combines the calculation of the estimate on both coarse grids with the best α obtained in the OES, process does improve the accuracy of the error estimate.

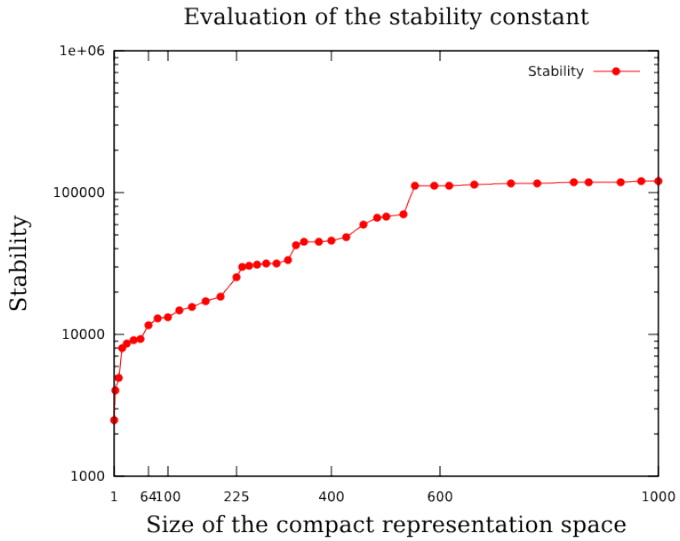

Figure 4.9: Evaluation of the stability constant for the modified NS problem in the L_2 norm.

Similar results are obtained for other norms, as shown in Figures 4.11 and 4.12. As observed previously, the quality of the estimate is less satisfying for the L_∞ norm, but it still provides an error estimate for the coarse grid calculation within 10% of the 'true' error. These results show that our procedure is independent of the norm used to do the calculations, and more specifically, that it is not required to know the norm of the finite element used by ADINA in that case.

We observe also that the error estimate in any of these discrete norms requires the same calculation. Once we have generated the surface response for α and the sensitivity calculation for Navier-Stokes, one can provide the *a posteriori* error estimate for all three discrete norms, $L_1,\ L_2,\ L_\infty$, at once.

Based on the OES we can proceed by giving error estimates on each coarse grid

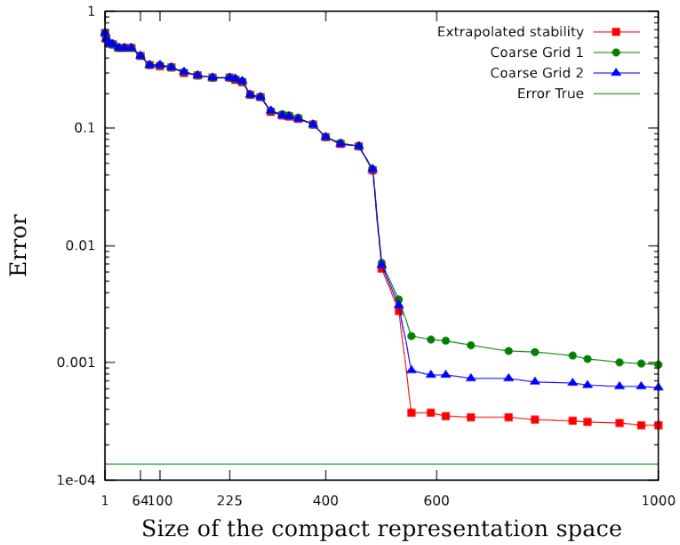

Figure 4.10: Evolution of the error versus the manufactured solution with the L_2 norm for the modified NS problem.

solution. This would not have been possible with the Richardson Extrapolation, which has inaccuracies. We observe also that the OES can serve as an initial guess for a new Navier-Stokes calculation on a refined grid if the *a posteriori* error estimate does not fit the goal of the simulation.

The fundamental result of Figure 4.10 is that we can provide an upper bound on the numerical error of the simulation for each coarse grid calculation within 10% of the 'true' error. The prediction of the error for the extrapolated solution is, however, within 50% of the 'true' error.

4.2.2.2 Battery

The energy equation 4.28 that governs the model of the second example is:

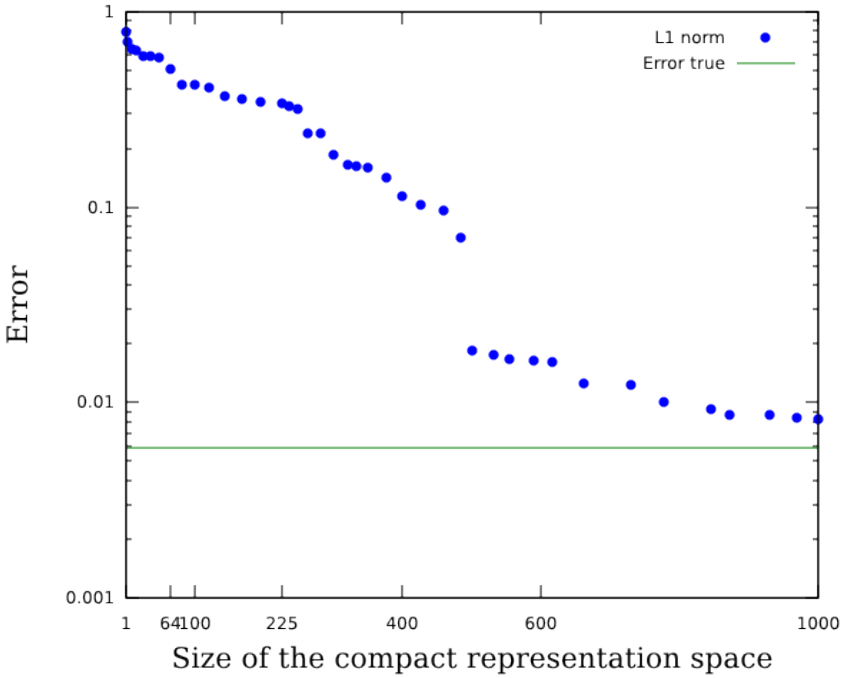

Figure 4.11: Evolution of the error versus the manufactured solution for the L_1 norm.

$$\frac{\partial}{\partial x_i}\left(k_{ij}(T)\frac{\partial T}{\partial x_j}\right) + Q(T) = \rho c_p(T)\frac{\partial T}{\partial t}, \qquad \text{on } \Omega \times (0,t) \tag{4.28}$$

with i, j running from 1 to 2 for this model. This two-dimensional problem is solved in a square domain $\Omega = (0, L_x) \times (0, L_y)$. T is the temperature, t is time, ρ is the material density, c_p is the specific heat as a function of T, Q is the volumetric heat source as a function of t, and k_{ij} is the thermal conductivity tensor as a function of T.

The boundary conditions are:

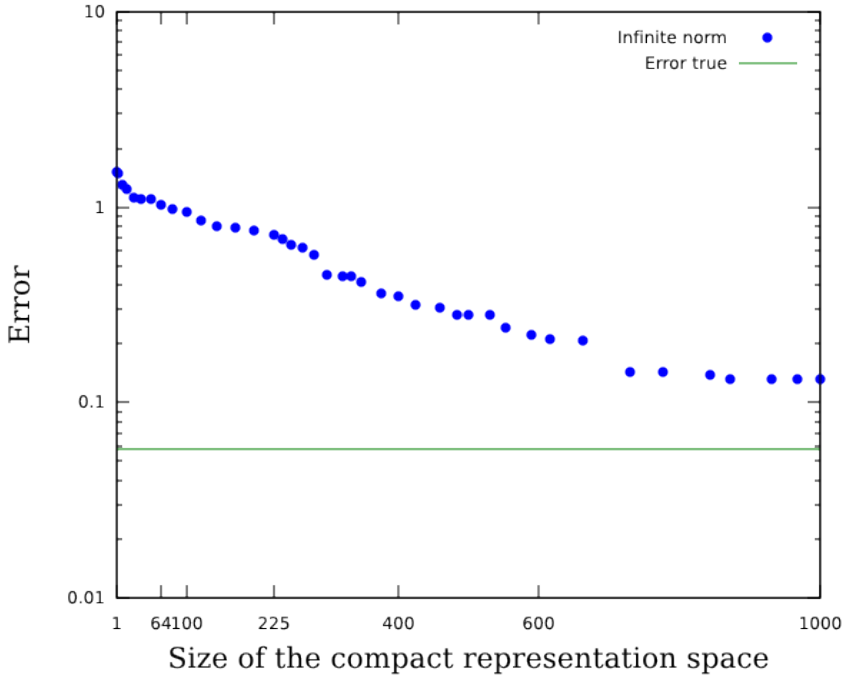

Figure 4.12: Evolution of the error versus the manufactured solution for the L_∞ norm.

$$\bullet - \left(k_{ij}\frac{\partial T}{\partial x_j}\right)\cdot n = h_1(T)(T - T_\infty) \tag{4.29}$$

$$+ \sigma\varepsilon_1\left(T^4 - T_\infty^4\right) \text{ on } \Gamma_{N_1} \tag{4.30}$$

$$\text{(radiation, convection);} \tag{4.31}$$

$$\bullet - \left(k_{ij}\frac{\partial T}{\partial x_j}\right)\cdot n = h_2(T)(T - T_\infty) \tag{4.32}$$

$$+ \sigma\varepsilon_2\left(T^4 - T_\infty^4\right) \text{ on } \Gamma_{N_2} \tag{4.33}$$

$$\text{(radiation, convection);} \tag{4.34}$$

$$\bullet - \left(k_{ij}\frac{\partial T}{\partial x_j}\right)\cdot n = h_3(T)(T - T_\infty) \text{on } \Gamma_{N_3} \tag{4.35}$$

$$\text{(convection); and} \tag{4.36}$$

$$\bullet \text{Symmetric boundary condition on } \Gamma_{N_4}, \tag{4.37}$$

where $T_\infty = 313.0K$, $\varepsilon_1 = \varepsilon_2 = 0.25$, $\sigma = 5.670 \times 10^{-8}$ is the Stefan-Boltzmann constant ($W \cdot m^{-2} \cdot K^{-1}$), and $h_3 = 1.0$ ($W \cdot m^{-2} \cdot K^{-1}$). The functions $h_1(T)$, $h_2(T)$, and c_p are given by tables.

Temperature is initialized to $T_0 = 313.0K$ in the structure. The difficulty of this study is due to the composition of the structure and the possibility of phase change in two regions. The material composing the structure might have coefficients depending on space and temperature. The problem is therefore very stiff and the solution is almost discontinuous near the wall, as shown in Figure 4.13. We are only interested in the final solution in time that should reach equilibrium. The problem becomes then unsteady. To simulate the heat transfer in this structure, we have used quad elements in each physical subdomain. The total numbers of elements the coarse grids G_1 and G_2 have are 8767 and 21'072 respectively . As opposed to the numerical experiment done before, we will assume that the fine grid solution obtained with 57'258 elements *is* the true solution.

The basis $\hat{b}^{E/F}$ used to perform the *a posteriori* error estimation for the battery problem are the sine and cosine trigonometric functions.

Figure 4.14 gives the numerical error in the L_2 norm as the function of the number of basis functions $b_i^{\hat{E}/F}$ used. We need on the order of 400 basis functions to reach a plateau in the error estimate. Once again we obtain an *a posteriori* error estimate on the coarse grid solution within 10% of the 'true' error versus the reference solution on $M(h_\infty)$. To be more specific, we are able to estimate a solution with an absolute error of 0.2 Kelvin, using coarse grid solutions with absolute errors of 10 Kelvin.

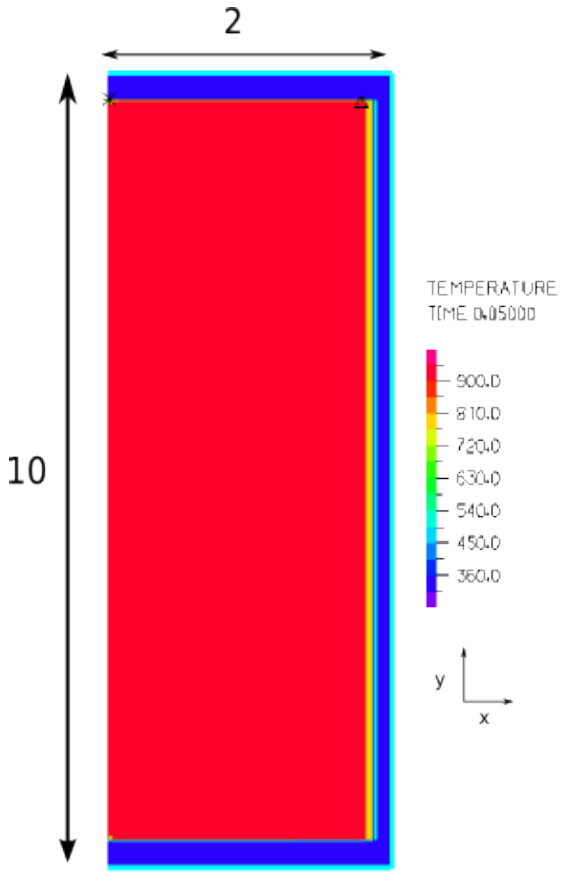

Figure 4.13: Steady state solution of the heat transfer problem.

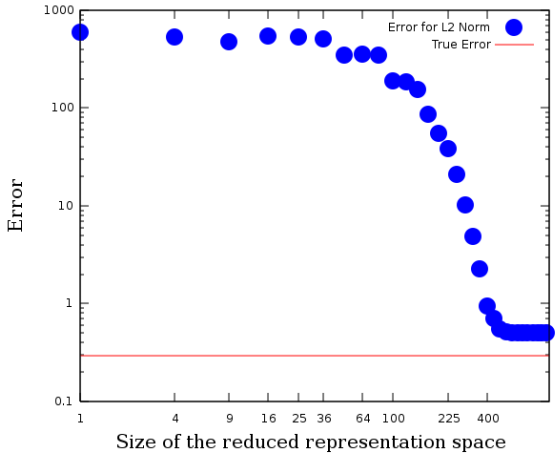

Figure 4.14: Evolution of the error versus the fine grid solution with the L_2 norm for the heat transfer problem.

4.3 Conclusion

In this chapter, we have explored a new framework to provide *a posteriori* estimation for CFD simulations produced with a commercial package. The challenge comes from the fact we do not have access to the source code, and we may not know precisely what finite element approximation has been used. In such a context, the only methods that practitioners use using are mesh refinement and eventually a Richardson Extrapolation procedure. This process is time-consuming and may not even be possible for complex PDE problems. We have proposed an alternative solution that still uses a fine mesh as a reference, but does not require the calculation of the CFD solution on this fine mesh.

Our method starts from the calculation of two or three solutions and searches for an optimized extrapolation on a fine mesh. This fine mesh supposedly resolves

all scales of interest. With this process, we can retrieve the best information from coarse grid solutions. Because we solve the problem as a (non)linear set of discrete equations and ignore the finite element framework, which may not be at our disposal for our specific application, our method is extremely general and can use all kinds of discrete norms to measure error. A sensitivity analysis is necessary to complete the work by providing the error estimate for the best consistent extrapolation solution we found.

The drawback of our method is that it is computationally intensive and requires hundreds of evaluation of the residual. On the other hand, this process has embarrassing parallelism and constitutes an effective way of using a grid of computers.

In the next section, we are going to describe our software implementation in more details .

Chapter 5

Distributed computation of optimized extrapolation method

5.1 Performance issues

Step 5 and Step 6 of our algorithm presented in Chapter 4 are sources of a large set of cumbersome computations for the backstep facing step flow test case. In Figures 4.9 and 4.10, one could observe that about 10^3 computations are required for a proper evaluation. Each of these computations can take several minutes depending on the size of the problem. This computation time takes into account the few explicit time steps needed to smooth the high frequency components of the projected approximation on the fine grid. Thus, a simple sequential implementation might require several days. The key feature that makes our solution verification procedure effective is its natural parallelism. Indeed, since the nodes are not sharing any information,

the local executions are all independent. In a modified implementation, threads are scheduling computational tasks for parallel execution. In order to perform this scheduling efficiently, part of the system should keep track of the completion of each task, and distribute the remaining tasks. A master node can centralize this information and thus ensure a natural load balancing of the tasks among the computational nodes. This is an important element for distributed computation, because it dynamically adjusts the workload of the computers depending on their availability and the calculations to be performed. Moreover, the master attributes a task to a slave node only if the current task is finished without an error message.

The last level to take into consideration is the user control interface. Even though the procedure might be parallel, the execution time might not always be negligible. Therefore, the user should be able to check and remotely control the execution of the verification procedure. For this reason, we developed a software interface that does not need to run on the master node or the computational node.

Figure 5.1 reflects the interaction between the components of our system. The idea is to take advantage of a network of computers by limiting the communications to a few megabytes of data only, and only when the computation on each node is finished. The slave nodes perform these computations because they only have the knowledge of the existence of the master node. A computational nexus distributes the task dynamically, self-adjusting to the deficiencies of the network or the availability of the computational nodes. The limiting factor of the process is the number of slave nodes we can access simultaneously since the computation is virtually scalable. Moreover, each task in Figure 5.1 makes use of a different programming language,

as follows:

- C# is used for the user interface because the goal was to have a good integration of the user interface with its environment;
- Java is used for the communications protocol on the master and computational nodes for portability reasons;
- C++ is employed for the preprocessing interfaces that are the core of our method; and
- the computational code, which is the target of our verification procedure, can be in any language.

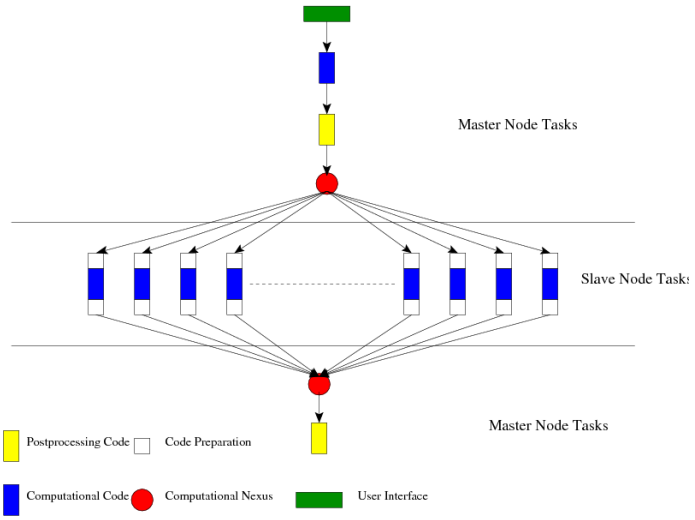

Figure 5.1: Task distribution for effective distributed computation in OES context.

We use a standard three-tier client/server/slave architecture. The *a posteriori* estimate takes a very small fraction of the time that it would take to compute the

fine grid solution. More details on this parallel implementation can be found in [72].

Figure 5.2 shows the performance improvement for three setups using a three-tier architecture when running the optimization procedure for the backward-facing step flow problem. The first column is for a sequential execution, the second column is for a heterogeneous network of computers, and the third column is the expected execution time with a parallelized interpolation. The computational grid used to perform the simulation contains Windows XP and Linux systems. Parts of the system run over a gigabit network, but a simple fast Ethernet network is connecting most of the systems.

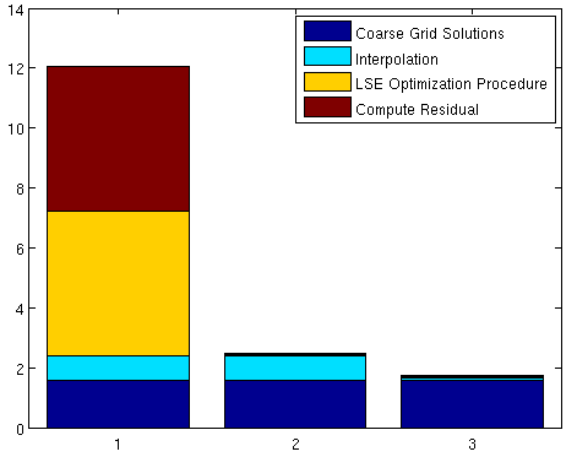

Figure 5.2: Overview of performance for a simple parallel implementation.

The hardware is also diversified: 3 Intel Core Duos E6300, 5 AMD Athlon 2800+, 1 Apple G5, 1 8-Way AMD Opteron 846. The speedup factor obtained for this simulation is approximately 9.5 times faster than on one of the AMD dual core PCs.

The computational grid cannot achieve a perfect speedup for several reasons:

- communication issues between computers located on different networks;
- differences in computing power between the different slave nodes; and
- the interpolation procedure was not parallelized, and was fairly expensive for this experiment with 6 10^4 mesh points.

This implementation of the solution verification procedure is independent of the operating system or architecture use. The Java communication interfaces only require the Java Runtime Environment to be present on the target machine.

While the architecture of the system is straightforward, other aspects should raise concerns, such as security and error checking. The data to be exchanged between the control interface, the master node, and the computational nodes can have a sensitive purpose. The spectrum of application ranges from medical data relative to a patient's condition, to physical data for a DOD project. It is, therefore, necessary to pay special attention to how transfer the data.

5.2 Error control

We have designed our system to be cost-effective by reusing any standard computer equipment that can support the Java technology. The three-tier architecture relies on computers and network elements that can be completely different by nature. Also, because of the mix of computational and communication processes, it is important

to access what the problems are and which components are involved in each entity.

A naive implementation of the system is shown in Figure 5.3. The main defect of this implementation is the lack of feedback in case of an error. There is no way to verify if a computation is finished, if the network is down, or if the master still has some tasks to process.

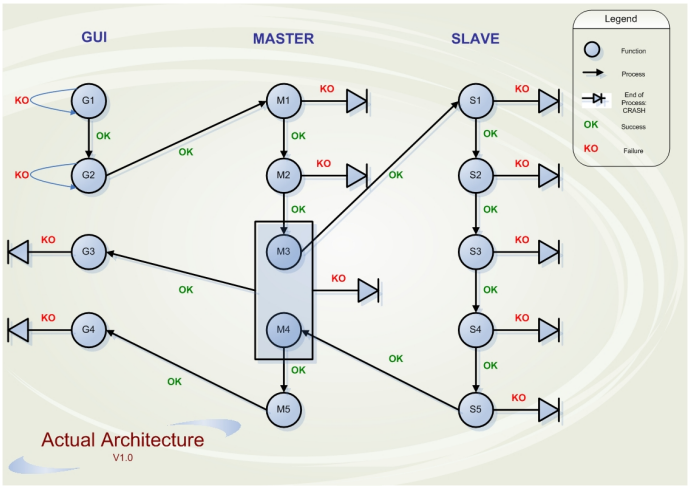

Figure 5.3: Naive implementation of the distributed computation.

The failure of this type of implementation leads us to include a checkpoint at each step of the process. This might appear to be a costly solution, but it is a necessity to ensure the robustness of the verification procedure. Figure 5.4 outlines the control points of our software. Figure 5.5 gives a short description of each element.

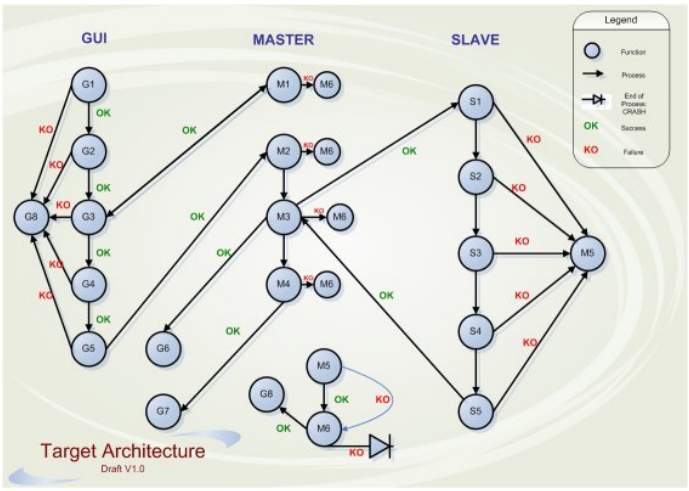

Figure 5.4: Overview of error control and secure data transfer.

5.3 Security

The last aspect of this implementation is the security. Again, if one follows the easy approach, as shown on Figure 5.6, some security problems might occur. Because the different entities can communicate over the Internet, communication must be secure between the entities. Authentication is important because the slaves of the grid can belong to research centers all over the world that have some kind of partnerships. If a partnership ends, the master of a grid should be able to forbid communication to the slave(s) of that specific research center.

Instead, one should use the procedure illustrated in Figure 5.7, which shows the authentication procedure between the master and a slave.

The implementation uses SSL Sockets instead of simple sockets. The communications between the different components need encryption. It is the function of

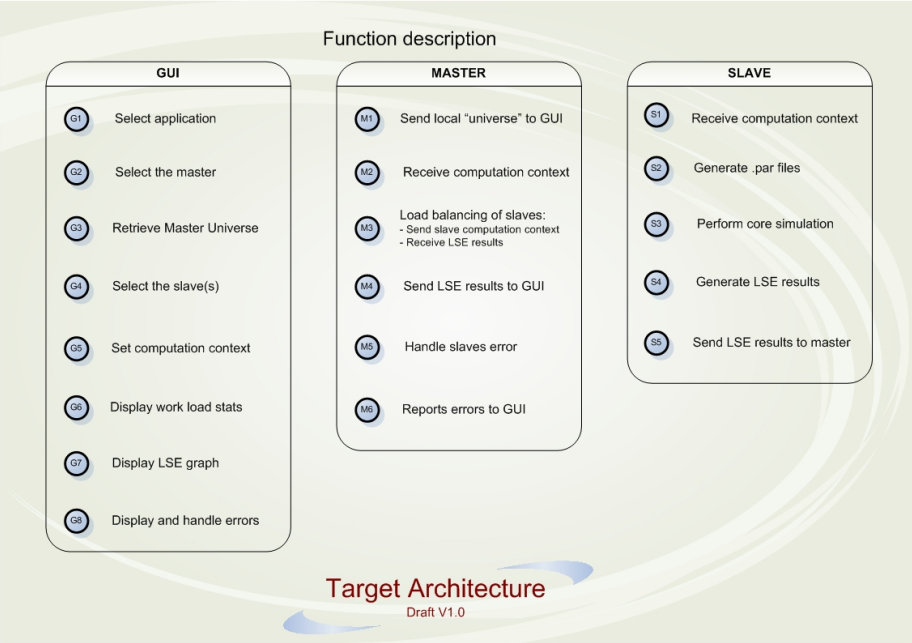

Figure 5.5: Description of controls elements for error control and secure data transfer.

SSL public/private keys to generate the 'session key'. The control interface owns the public key, the master node owns both the private and the public key and the computational nodes own the public key. Security issues dictate this distribution of the keys. The question remains of where the keys should be stored.

In the SSL scheme, the communication is secure but any computational node can obtain the public key to exchange information with the master. It is the confidentiality, integrity, and authentication concept for data security. SSL sockets realize both confidentiality and integrity, but there is no control on the authentication. This justifies the implementation of an authentication method between the master node and the slave nodes to ensure that only 'allowed slaves' can communicate with the

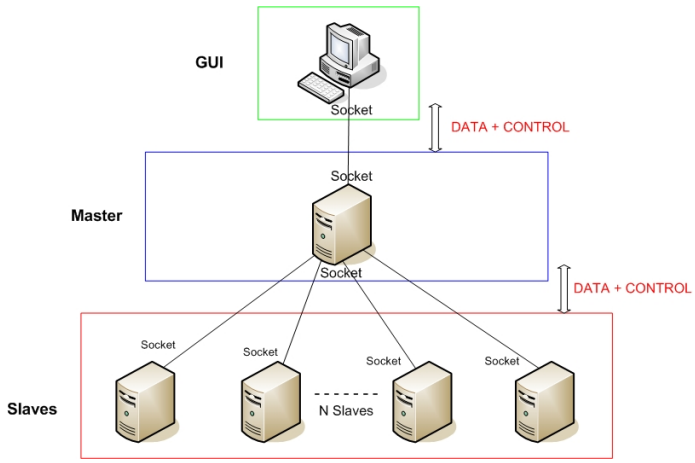

Figure 5.6: Naive communication between nodes

master node.

In our architecture, a file stores an encrypted login and password, thus providing the authentication source for the master. When a computational node wants to communicate with the master, it initiates the SSL, and then the master asks the computational node for a login and a password. The computational nodes find their authentication data in some other encrypted file. This file is created either the first time the computational node socket initiates, or when the socket loads and does not find the file in its root directory. The interface will prompt the administrator that launches the socket for the login and the password. In the authentication process, if a computational node has a wrong login or a wrong password, some data sent on the SSL socket will notify the control interface. The SSL tunnel is closed, and a pop-up notifies the user of the console that one or more slaves did not authenticate properly. Figure 5.8 shows the communication process between the master and the

slaves regarding the establishment of the SSL tunnel and the authentication process.

The overhead generated by this control statement and by the SSL sockets encryption is negligible with respect to the time needed to execute a task on the computational node. When computational nodes are in a cluster with private IP addresses, the mechanisms required are more specific. One can, for example, create a new level in the hierarchy by giving some scheduling control to the node of the cluster that owns a public IP address.

5.4 Conclusion

This section has focused on the software side of the implementation and the different constraints we faced when developing the software. The most interesting seems to be that we managed to obtain good performance without neglecting security and error checking. The results show that if we had a homogeneous grid of computers, i.e., only identical nodes in our network, the scalability would be quasi-perfect. The efficiency is a direct consequence of the structure of the algorithm. Indeed, each node needs to compute only its solution without knowing anything about the other client nodes, and by only exchanging a really small amount of data with the master node.

The next section presents our conclusion on the optimized extrapolation method and its implementation using distributed computing.

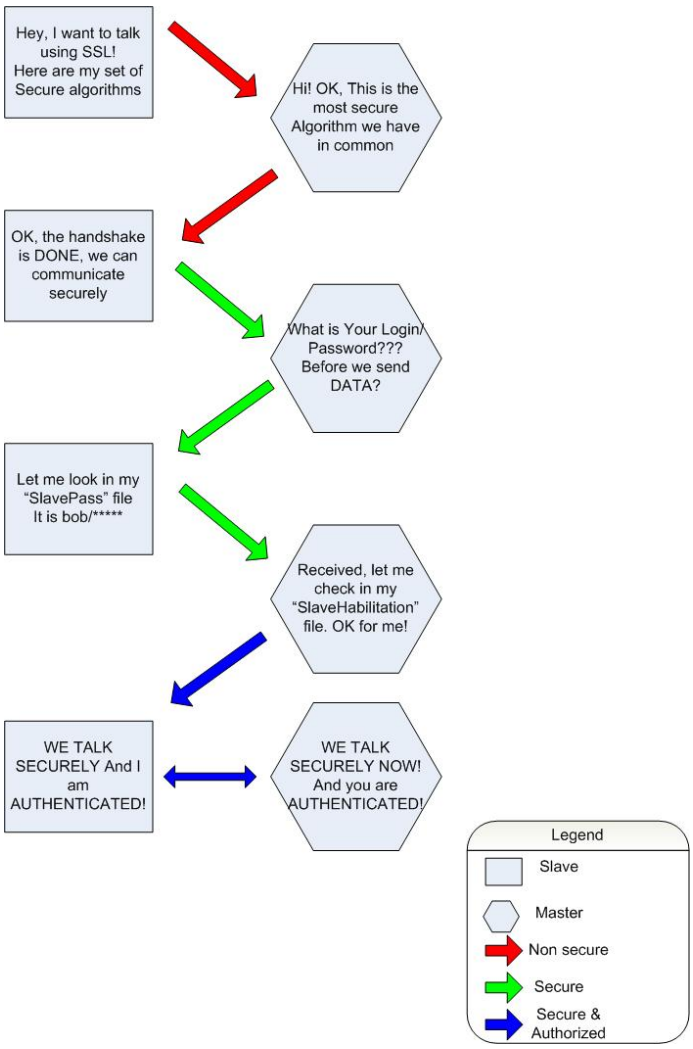

Figure 5.7: Handshake process.

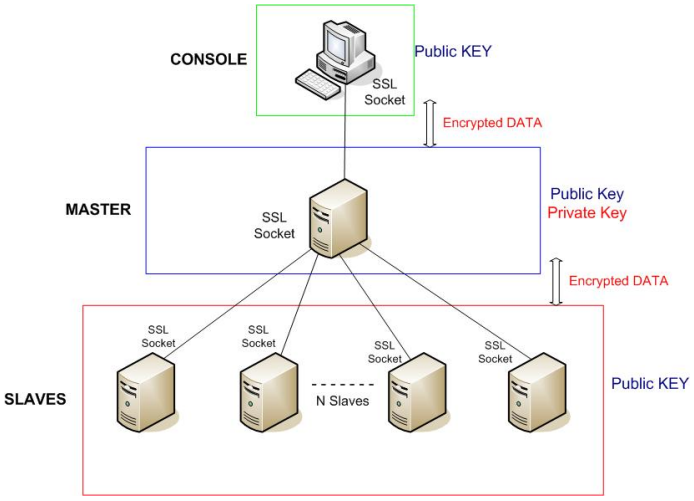

Figure 5.8: SSL communication scheme.

Chapter 6

Conclusion

6.1 Summary

In this dissertation, we have presented three original results. First, we extended the least square extrapolation method to parabolic equations. The optimized extrapolated solutions computed are more accurate than those of the Richardson Extrapolation method. Moreover, our method has proved to be effective on various equation sets, ranging from phase separation equations to shock layer models.

The second innovative contribution was the *a posteriori* error estimate framework. This framework provides a better tool for solution verification than Richardson Extrapolation when the convergence order is space-dependent or far from the asymptotic rate of convergence. It also provides an easy way to evaluate numerically the stability condition for complex and different PDEs. The strength of the method

is that no specific implementation knowledge or discretization details are necessary to be able to obtain an *a posteriori* error estimate. However, this *a posteriori* error estimate refers to a very fine grid solution (never computed) that the same approximation framework is able to provide. We do have to assume then that the code does converge, and that the code is correct in the sense of software verification, for example, manufactured solutions. The combination of these two aspects leads to a postprocessing method that is fairly versatile, general, and friendly to the user.

Finally, we produced a software that can perform the optimized extrapolation and the error estimation on a grid of computers. Despite its computational cost, we show that, because of its intrinsic parallel nature, the computation benefits from an efficient parallel implementation. This implementation is also neutral with respect to the operating system, architecture, or network. We successfully managed to achieve that goal for a simple application. Moreover, the solution we developed is secure and by nature fault-tolerant on the client side.

6.2 Future Work

First, more applications could benefit from the framework presented in this dissertation. For instance, we have presented results essentially for fluid flow computations and heat equations. An extension to structure equations, with the goal of verifying fluid-structure applications, should be one of the main pivots of such additional research.

Second, there is no *a posteriori* error estimate for parabolic equations in this

dissertation. By extending the work done in Chapter 3, one could develop an *a posteriori* error estimate for time dependent equations.

Further, the software library could benefit from some abstraction. It may trigger interest from software developers for inclusion and testing with more simulation applications. The simplicity of the framework makes it an attractive solution for software engineers who do not want to modify their code to include error estimation tools.

Glossary

Code validation

It is the process of determining the degree to which a model is an accurate representation of the real world from the perspective of the intended uses of the model.

Code verification

It aims at finding and removing mistakes in the source code and in the numerical algorithms, and improving software using software quality assurance practices.

A posteriori **error estimates**

Technique for estimating the error in a numerical solution to a PDE that make use of the numerical algorithm that approximates the partial differential operators, the initial and boundary conditions and previous numerical solutions.

A priori **error estimates**

Technique for estimating the error in a numerical solution to a PDE that make

use of the numerical algorithm that approximates the partial differential operators, and the initial and boundary conditions.

Solution verification

It aims at assuring the accuracy of input data for the problem of interest, estimating the numerical solution error and assuring the accuracy of output data for the problem of interest.

Bibliography

[1] D. Keyes. *A science-based case for large-scale simulation*, volume 1. DOE, 2003.

[2] D. Keyes. *A science-based case for large-scale simulation*, volume 2. DOE, 2003.

[3] W.L. Oberkampf and T.G. Trucano. Verification and validation in computational fluid dynamics. Technical report, Sandia National Laboratory, 2002.

[4] I. Babuska and H-S. Oh. Pollution problem of the p- and h-p versions of the finite element method. *Communications in Applied Numerical Methods*, 3(6):553–561, 1987.

[5] P.J. Roache. *Verification and Validation in Computational Science and Engineering*. Hermosa Publishers, Albuquerque, New Mexico, 1998.

[6] M. Ainsworth. Identification and a posteriori estimation of transported errors in finite element analysis. *Computer Methods in Applied Mechanics and Engineering*, 176(1-4):3–18, 1999.

[7] M. Ainsworth and I. Babuska. Reliable and robust a posteriori error estimation for singularly perturbed reaction-diffusion problems. *SIAM Journal on Numerical Analysis*, 36(2):331 – 353, 1999.

[8] D. Kay and D. Silvester. A posteriori error estimation for stabilized mixed approximations of the stokes equations. *SIAM Journal on Scientific Computing*, 21(4):1321–1336, 2000.

[9] P. Coorevits, P. Hild, and J.P. Pelle. A posteriori error estimation for unilateral contact with matching and non-matching meshes. *Computer Methods in Applied Mechanics and Engineering*, 186(1):65–83, 2000.

[10] S. Adjerid. A posteriori error estimates for fourth order elliptic problem. *Computer Methods in Applied Mechanics and Engineering*, 191(23-24):2539–2559, 2002.

[11] S. Adjerid. A posteriori finite element error estimation for second order hyperbolic problems. *Computer Methods in Applied Mechanics and Engineering*, 191(41-42):4699–4719, 2002.

[12] S. Adjerid and T.C. Massey. A posteriori discontinuous finite element error estimation for two-dimensional hyperbolic problems. *Computer Methods in Applied Mechanics and Engineering*, 191(51-52):5877–5897, 2002.

[13] P. Castillo, B. Cockburn, D. Schotzau, and C. Schwab. An optimal a priori error estimate for the hp-version of the local discontinuous galerkin method for convection-diffusion problems. *Mathematics of Computation*, 71:455–478, 2002.

[14] P. Castillo, B. Cockburn, I. Perugia, and D. Schötzau. An a priori error analysis of the local discontinuous Galerkin method for elliptic problems. *SIAM Journal on Numerical Analysis*, 38(5):1676–1706, 2000.

[15] S. Albert, B. Cockburn, D. A. French, and T. E. Peterson. A posteriori error estimates for general numerical methods for Hamilton-Jacobi equations. Part I: The steady state case. *Mathematics of Computation*, 71:49–76, 2002.

[16] D.J. Estep, S.M. V. Lunel, and R.D. Williams. Analysis of shear layers in a fluid with temperature-dependent viscosity. *Journal of Computational Physics*, 173:17–60, 2001.

[17] O. Kruger, M. Picasso, and J.F. Scheid. A posteriori error estimates and adaptive finite elements for a nonlinear parabolic problem related to solidification. *Computer Methods in Applied Mechanics and Engineering*, 192(5-6):535–558, 2003.

[18] R. Rannacher and F.T. Suttmeier. A posteriori error estimation and mesh adaptation for finite element models in elasto-plasticity. *Computer Methods in Applied Mechanics and Engineering*, 176:333–361, 1999.

[19] L. Machiels, J. Peraire, and A. T. Patera. A posteriori finite element output bounds for incompressible Navier-Stokes equations; application to a natural convection problem. *Journal of Computational Physics*, 172(2):401–425, 2000.

[20] M. Paraschivoiu, J. Peraire, and A. T. Patera. A posteriori finite element bounds for linear–functional outputs of elliptic partial differential equations. *Computer Methods in Applied Mechanics and Engineering*, 150:289–312, 1997.

[21] A.T. Patera, J. Peraire, L. Machiels, and Y. Maday. A general framework for finite element A posteriori error control: Application to linear and nonlinear convection-dominated problems. In *ICFD Conference on Numerical Methods for Fluid Dynamics, Oxford, England*, volume 328, pages 823–828, 1998.

[22] J. Sarrate, J. Peraire, and A. Patera. Posteriori finite element error bounds for nonlinear outputs of the helmholtz equation. *International Journal of Numerical Methods in Fluids*, 31(1):17–36, 1999.

[23] D.A. Venditti and D.L. Darmofal. A grid adaptive methodology for functional outputs of compressible flow simulations. In *15th AIAA Computational Fluid Dynamics Conference*, number AIAA 2001-2659, 2001.

[24] D.A. Venditti and D.L. Darmofal. Grid adaptation for functional outputs: application to two-dimensional inviscid flows. *Journal of Computational Physics*, 176(1):40–69, 2002.

[25] O.C. Zienkiewicz and J.Z. Zhu. The superconvergent patch recovery and a posteriori error estimates.Part 1: The recovery technique. *International Journal for Numerical Methods in Engineering*, 33:1311–1364, 1992.

[26] O.C. Zienkiewicz and J.Z. Zhu. The superconvergent patch recovery and a posteriori error estimates.Part 2: Error estimates and adaptivity. *International Journal for Numerical Methods in Engineering*, 33:1365–1382, 1992.

[27] C. Carstensen and S. Funken. Averaging technique for FE – a posteriori error control in elasticity. *Computer Methods in Applied Mechanics and Engineering*, 190(35-36):4663–4675, 2001.

[28] C. Carstensen and B. Faermann. Mathematical foundation of a posteriori estimates and adaptive mesh refining algorithm for boundary integral equations of the first kind. *Engineering Analysis with Boundary Elements*, 25(7):497–509, 2001.

[29] J. Glimm, H. Kim, D. Sharp, and T. Wallstrom. A stochastic analysis of the scale up problem for flow in porous media. *Computational and Applied Mathematics*, 17:67–79, 1998.

[30] J. Glimm and D. Sharp. Stochastic partial differential equations: Selected applications in continuum physics. *Mathematical Surveys and Monographs*, 64:3–44, 1998.

[31] J. Glimm and D. H. Sharp. Prediction and the quantification of uncertainty. *Physica D*, 133(1-4):152–170, 1999.

[32] J. Glimm and D. Sharp. Stochastic methods for the prediction of complex multiscale phenomena. *Quarterly of Applied Mathematics*, LVI(4):741–765, 1998.

[33] J. Glimm, S. Hou, H. Kim, D. Sharp, and K. Ye. A probability model for errors in the numerical solutions of a partial differential equation. *Computational Fluid Dynamic Journal*, 9(1):485–493, 2001.

[34] H. Kim. *Computational methods for statistical solutions of inverse problems for flow in porous media*. PhD thesis, Stony Brook University, 2000.

[35] J. Xu and S. B. Pope. Assessment of numerical accuracy of pdf/monte carlo methods for turbulent reacting flows. *Journal of Computational Physics*, 152(1):192–230, 1999.

[36] J. Gustafson. Computational verifiability and feasibility of the asci program. *IEEE Computational Science and Engineering*, 05(1):36–45, 1998.

[37] American Institute of Aeronautics and Astronautics Staff. *AIAA Guide for the Verification and Validation of Computational Fluid Dynamics Simulations*. American Institute of Aeronautics & Astronautics, 1998.

[38] Council on Codes, Board on Performance Test Codes: Committee on Verification Standards, and Validation in Computational Solid Mehanics. *V&V 10 - Guide for Verification and Validation in Computational Solid Mechanics*. American Society of Mechanical Engineers, 2006.

[39] M. Garbey and W. Shyy. On optimized extrapolation method for elliptic problems with coefficients having large variation. Technical report, preprint UH-CS-05-17 submitted., Dept. of Computer Science, University of Houston, 2005.

[40] W. Shyy, M. Garbey, A. Appukuttan, and J. Wu. Evaluation of richardson extrapolation in computational fluid dynamics. *Numerical Heat Transfer: Part B: Fundamentals*, 41(2):139 – 164, 2002.

[41] I. Celik, J. Li, G. Hu, and C. Shaffer. Limitations of richardson extrapolation and some possible remedies. *Journal of Fluids Engineering*, 127:795–805, 2005.

[42] P. J. Roache and P. M. Knupp. Completed richardson extrapolation. *Communications in Numerical Methods in Engineering*, 9:365–374, 1993.

[43] M. Francois. Computations of drop dynamics with the immersed boundary method, part 1: Numerical algorithm and buoyancy-induced effect. *Numerical Heat Transfer Part B: Fundamentals*, 44:101–118(18), August 2003.

[44] M. Francois. Computations of drop dynamics with the immersed boundary method, part 2: Drop impact and heat transfer. *Numerical Heat Transfer Part B: Fundamentals*, 44:119–143(25), August 2003.

[45] C.S. Pekin. The immersed boundary method. *Acta Numerica*, 11:479–517, 2002.

[46] M. Garbey and W. Shyy. A least square extrapolation method for improving solution accuracy of pde computations. *Journal of Computational Physics*, 186(1):1–23, 2003.

[47] M. Garbey and W. Shyy. A least square extrapolation method for the a posteriori error estimate of the incompressible navier stokes problem. *International Journal of Numerical Methods in Fluids*, 48:43–59, 2005.

[48] A.A. Björck. *Numerical Methods for Least Squares Problems*. SIAM, Philadelphia, PA, 1996.

[49] J. A. Nelder and R. Mead. A simplex method for function minimization. *Computer Journal*, 7:308–313, 1965.

[50] N.J. Higham. Fortran codes for estimating the one-norm of a real or complex matrix, with applications to condition estimation. *ACM Transactions on Mathematical Software*, 14(4):381–396, 1988.

[51] N.J. Higham. Experience with a matrix norm estimator. *SIAM Journal on Scientific and Statistical Computing*, 11(4):804–809, 1990.

[52] M. Ainsworth and T.J. Oden. *A Posteriori Error Estimation in Finite Element Analysis*. Wiley, 2000.

[53] R. Verfurth. *A review of a posteriori error estimation techniques Adaptive Mesh-Refinement Techniques*. Wiley-Teubner, 1996.

[54] W. Eckhaus. *Asymptotic Analysis of Singular Perturbations*. North–Holland, Amsterdam, 1979.

[55] M. Garbey and H.G. Kaper. *Numerical Methods for PDEs with Critical Parameters*. Kluwer, Dordrecht, 1992.

[56] H.G. Kaper and M. Garbey. *Asymptotic-induced Numerical Methods,in: Asymptotic Analysis and the Numerical Method of PDEs*, volume 130. Lect. Notes in Pure and Applied Math., Dekker, 1991.

[57] D. Gottlieb and C.W. Shu. On the gibbs phenomenon and its resolution. *SIAM Review*, 39(4):644–668, 1997.

[58] G. H. Golub and C. F. Van Loan. *Matrix Computations*. Baltimore, MD, USA, third edition, 1996.

[59] A-K. Tornberg and B. Engquist. Numerical approximations of singular source terms in differential equations. *J. Comput. Phys.*, 200(2):462–488, 2004.

[60] M. Garbey. Some remarks on multilevel method, extrapolation and code verification. In N. Debit, M. Garbey, R. Hoppe, D. Keyes, Y. Kuznetsov, and J. Periaux, editors, *13th Int. Conf. on Domain Decomposition DD13, Domain Decomposition Methods in Science and Engineering*, pages 379–386. CIMNE, Barcelona, 2002.

[61] D.L. Ropp, J.N. Shadid, and C.C. Ober. Studies of the accuracy of time integration methods for reaction-diffusion equations. *Journal of Computational Physics*, 194(2):544–574, 2004.

[62] Anne Bourlioux, Andrew J. Majda, and Victor Roytburd. Theoretical and numerical structure for unstable one-dimensional detonations. *SIAM Journal on Applied Mathematics*, 51(2):303–343, 1991.

[63] M. Garbey. Domain decompositin to solve transition layers and asymptotics. *SIAM Journal on Scientific Computing*, 15(4):866–891, 1994.

[64] Anne Bourlioux, Anita T. Layton, and Michael L. Minion. High-order multi-implicit spectral deferred correction methods for problems of reactive flow. *Journal of Computational Physics*, 189(2):651–675, 2003.

[65] P. Colella, A. Majda, and V. Roytburd. Theoretical and numerical structure for reacting shock waves. *SIAM Journal on Scientific and Statistical Computing*, 7(4):1059–1080, 1986.

[66] P. Colella and P.R. Woodward. The piecewise parabolic method (ppm) for gas-dynamical simulations. *Journal of Computational Physics*, 54:174–201, 1984.

[67] P.K. Brazhnik and J. J. Tyson. On traveling wave solutions of fisher's equation in two spatial dimensions. *SIAM Journal on Applied Mathematics*, 60(2):371–391, 2000.

[68] J.T. Oden. A posteriori error estimation. In *Verification and Validation in Computational Solid Mechanics*. Hans Maier, ASME/USACM Standards, 2002.

[69] P.J. Roache. Code verification by the method of manufactured solutions. *Journal of Fluids Engineering*, 124:4–10, 2002.

[70] G. Berkooz, P. Holmes, and J.L. Humley. The proper orthogonal decomposition in the analysis of turbulent flows. *Annual Review of Fluid Mechanics*, 25:539–575, 1993.

[71] R.H. Myers and D.C. Montgomery. *Response Surface Methodology: Process and Product in Optimization Using Designed Experiments*. John Wiley & Sons, Inc., 1995.

[72] C. Picard, M. Garbey, and V. Subramanian. Mapping lse on a grid: Software architecture and performance gains. In *International Conference on Parallel Computational Fluid Dynamics*, 2005.

[73] J.J. More and S.J. Wright. *Optimization Software Guide*. Society for Industrial and Applied Mathematics, 1993.

[74] M. Garbey and C. Picard. Toward a general solution verification method for complex pde problem with hands off coding. Technical report, preprint UH-CS-07-2 submitted., Dept. of Computer Science, University of Houston, 2007.

[75] C.J. Roy and M. M. Hopkins. Discretization error estimates using exact solutions to nearby problems. In *41st AIAA Aerospace Sciences Meeting & Exhibit*, 2003.

[76] A.T. Patera and J. Peraire. A general lagrangian formulation for the computation of a-posteriori finite element bounds. In *Error Estimation and Adaptive Discretization Methods in CFD*. T. Barth & H. Deconinck, 2002.

[77] M. Garbey and C. Picard. A least square extrapolation method for the a posteriori error estimate of cfd and heat transfer problem. In *Sixth European Conference on Structural Dynamics, Paris*, 2005.

[78] M. Garbey and D. Tromeur-Dervout. A new parallel solver for the nonperiodic incompressible navier–stokes equations with a fourier method: Application to

frontal polymerization. *Journal of Computational Physics*, 145(1):316–331, 1998.

[79] M. Garbey, Yu. A. Kuznetsov, and Yu. V. Vassilevski. A parallel schwarz method for a convection-diffusion problem. *SIAM Journal on Scientific Computing*, 22(3):891–916, 2000.

[80] T. Barth and M.G. Larson. A posteriori error estimates for higher order godunov finite volume methods on unstructured meshes. Technical report, NASA Ames Research Center Code IN, 2002.

[81] C. Ilinca, X.D. Zhang, J.Y. Trepanier, and R. Camarero. A comparison of three error estimation techniques for finite-volume solutions of compressible flows. *Computer Methods in Applied Mechanics and Engineering*, 18:1277–1294, 2000.

[82] M. Garbey and Yu. V. Vassilevski. A parallel solver for unsteady incompressible 3d navier-stokes equations. *Parallel Computing*, 27(4):363–389, 2001.

[83] M. Ainsworth and J. T. Oden. A posteriori error estimates for stokes and oseen equations. *SIAM Journal on Numerical Analysis*, 34:228–245, 1997.

[84] M. Ainsworth and D. W. Kelly. A posteriori error estimators and adaptivity for finite element approximation of the non-homogeneous dirichlet problem. *Advances in Computational Mathematics*, 15:3–23, 2001.

[85] A. Bergam, C. Bernardi, and Z. Mghazli. A posteriori analysis of the finite element discretization of some parabolic equations. *Mathematics of Computation*, 74(251):1117–1138, 2005.

[86] C. Bernardi and R. Verfürth. A posteriori error analysis of the fully discretized time-dependent stokes equations. *Mathematical Modelling and Numerical Analysis*, 38:437–455, 2003.

[87] R. Verfurth. A posteriori error estimation techniques for non-linear elliptic and parabolic pdes. *Revue europeenne des elements finis*, 9:377–402, 2000.

[88] J.R. Stewart and T.J.R. Hughes. A tutorial in elementary finite element error analysis: A systematic presentation of a priori and a posteriori error estimates. *Computer Methods in Applied Mechanics and Engineering*, 158:1–22, 1998.

[89] R. Rannacher. Adaptive galerkin finite element methods for partial differential equations. *Journal of Computational and Applied Mathematics*, 128(1-2):205–233, 2001.

[90] J. B. Burie and M. Marion. Adaptive multilevel methods in space and time for parabolic problems-the periodic case. *Mathematics of Computation*, 69:547 – 581, 2000.

[91] M. B. Giles and E. Suli. Adjoint methods for PDEs: a posteriori error analysis and postprocessing by duality. *Acta Numerica*, 11:145–236, 2002.

[92] J. Peraire and A.T. Patera. Asymptotic a posteriori finite element bounds for the outputs of noncoercive problems: the Helmholtz and Burgers equations. *Computer Methods in Applied Mechanics and Engineering*, 171:77–86, 1999.

[93] M. Garbey and H.G. Kaper. Asymptotic-numerical study of supersensitivity for generalized burgers' equations. *SIAM Journal on Scientific Computing*, 22(1):368–385, 2000.

[94] W.L. Oberkampf. Bibliogrpahy for verification and validation in computational simulation. Technical report, Sandia National Laboratory, 1998.

[95] R. Vaidyanathan, W. Shyy, M. Garbey, and R. Haftka. Cfd code verification using least square extrapolation method. In *42nd AIAA Aerospace Sciences Meeting and Exhibit*, 2004. AIAA 2004-0739.

[96] R. Peyret and T.D. Taylor. *Computational Methods for Fluid Flow*. Springer-Verlag, New York, 1985.

[97] W. Shyy, S.S. Thakur, H. Ouyang, L. Liu, and E. Blosch. *Computational Techniques for Complex Transport Phenomena*. Cambridge University Press, New York, 1997.

[98] J. Baranger and H. El-Amri. Estimateur a posteriori d'erreur pour le calcul adaptatif d'ecoulement quasi-newtoniens. *Mathematical Modelling and Numerical Analysis*, 25:31–48, 1991.

[99] D.J. Estep, M.G. Larson, and R.D. Williams. *Estimating the error of numerical solutions of systems of reaction-diffusion equations*. American Mathematical Society, 2000.

[100] P.K. Moore. Finite difference methods and spatial a posteriori error estimates for solving parabolic equations in three space dimensions on grids with irregular nodes. *SIAM Journal on Numerical Analysis*, 36(4):1044–1064, 1999.

[101] C.J. Roy. Grid convergence error analysis for mixed-order numerical schemes. *AIAA Journal*, 41(4):595–604, 2003.

[102] G.P. Warren, W.K. Anderson, J.T. Thomas, and S.L. Krist. Grid convergence for adaptive methods. In *10th AIAA Fluid Dynamics Conference*, number AIAA 91-1592, 1991.

[103] M. Pilch, T.G. Trucano, J.L. Moya, G. Froehlich, and A. Hodges. Guidelines for sandia asci verification and validation plans - content and format: Version 2.0. Technical report, Sandia National Laboratory, 2001.

[104] M. Garbey and H.G. Kaper. Heterogeneous domain decomposition for singularly perturbed elliptic boundary value problems. *SIAM Journal on Numerical Analysis*, 34(4):1513–1544, 1997.

[105] B.I. Wohlmuth. Hierarchical a posteriori error estimators for mortar finite element methods with lagrange multipliers. *SIAM Journal on Numerical Analysis*, 36:1636–1658, 1999.

[106] W.L. Oberkampf, F.G. Blottner, and D.P. Aeshliman. Methodology for computational fluid dynamics code verification/validation. In *26th AIAA Fluid Dynamics Conference*, number AIAA 95-2226, 1995.

[107] A. Ecer, M. Garbey, and M. Hervin. On the design of robust and efficient algorithms that combine schwarz method and multilevel grids. In C.B.Jenssen et al., editors, *12th International Conference on Parallel Computational Fluid Dynamics*, pages 165–172, 2000.

[108] T.G. Trucano, M. Pilch, and W.L. Oberkampf. On the role of code comparisons in verification and the validation. Technical report, Sandia National Laboratory, 2003.

[109] A.G. Hutton and M.V. Casey. Quality and trust in industrial cfd - a european initiative. In *39th AIAA Aerospace Sciences Meeting*, 2001. AIAA 2001-0656.

[110] R.J. O'Malley. *Singular Perturbation Methods for Ordinary Differential Equations*, volume 89. Springer Verlag Applied Mathematical Sciences, 1991.

[111] R.E. Bank and A. Weiser. Some a posteriori error estimators for elliptic partial differential equations. *Mathematics of Computation*, 44(170):283–301, 1985.

[112] N. A. Baker, D. Sept, M. J. Holst, and J.A. McCammon. The adaptive multilevel finite element solution of the Poisson–Boltzmann equation on massively parallel computers. *IBM Journal of Research and Development*, 40(4):427–438, 2001.

[113] A. Chorin, A. Kast, and R. Kupferman. Unresolved computation and optimal prediction. *Communications on Pure and Applied Mathematics*, 52:1231–1254, 1999.

[114] A. El Hamidi and M. Garbey. Using maple for the analysis of bifurcation phenomena in gas combustion. *Theoretical Computer Science*, 187(1-2):249–262, 1997.

[115] C.J. Roy, M.A. McWherter-Payne, and W.L. Oberkampf. Verification and validation for laminar hypersonic flowfields. In *AIAA Fluids 2000 Conference*, number AIAA 95-2226, 2000.

[116] C.J. Roy, T.M. Smith, and C.C. Ober. Verification of a compressible cfd code using the method of manufactured solutions. In *32nd AIAA Fluid Dynamics Conference*, number AIAA 2002-3110, 2002.

[117] W.L. Oberkampf, T.G. Trucano, and H. Charles. Verification, validation and predictive capability in computaional engineering and physics. Technical report, Sandia National Laboratory, 2003.

[118] W.P. Burgess. *Extrapolation techniques applied to parabolic partial differential equations.* PhD thesis, Princeton University, 1971.