Thirteenth International Conference on Domain Decomposition Methods
Editors: N. Debit, M.Garbey, R. Hoppe, J. Périaux, D. Keyes, Y. Kuznetsov ©2001 DDM.org

38 Some Remarks on Multilevel Method, Extrapolation and Code Verification

M. Garbey [1] [2]

1 Motivation

Large scale CFD is becoming an important tool in the industry, and its success is very much connected to the reliability of the output and the cost to produce the data. It is rare that the reliability of the answer can be based exclusively on a firm mathematical basis, because the models of interest for industry are often too complex for that. Code validation and verification are therefore becoming essential.

Code Validation (1) and Verification (2) decompose roughly into a search for: Physical Modeling Errors (1), Discretization Errors (2), Programming Errors (2) (i.e mistakes), and Computer Roundoff errors (2). We refer to [OBA95] for a proper taxonomy of errors. According to [R98], discretization error can be evaluated by grid refinement verification studies. The so-called grid convergence index [R98] can be computed to assess accuracy. Richardson extrapolation plays a central role, in this type of code verification but is limited to approximation method with a known order of convergence. This information for complex flow simulation is not available in general, either because the hypothesis of the approximation theory are not fully satisfied or simply because error estimates are essentially asymptotic relations and practical calculation have not meshes fine enough to be in the range of the asymptotic limit description. A better solution from the stand point of applied mathematics is to use an a posteriori estimator, and lead the grid refinement with this tool. However, we will stay away from this solution in order to deal with a CFD code as a black box, as it should be done in principle for code verification.

We would like further to point out that the verification of large scale computation code is difficult with modern parallel computing environment. Discretized errors might be affected because the type of grid that is distributed on parallel computers is not necessary the classical one that used on a sequential machine ; classical examples are overlapping grids , non matching grids, etc ... Programming errors on parallel systems with distributed memory as Beowulf systems are of main concern because parallel codes are more complex than sequential code. Also there is no shared resources on Multiple Instruction Multiple Data Architecture (MIMD), communication might be asynchronous, and result can depend slightly on the run, with modern iterative solvers [1]. Round off errors are critical, because the grids have very large number of nodes and therefore the condition numbers might be very high.

In this paper, we will discuss some methodology that tries to combine multilevel method, such as cascade of computation on refined grids, with some aspect of code verification such as order of accuracy verification. One of the key constraints that we would like to keep in the design of this methodology is the fact that one reuses an existing CFD code on different grids without rewriting the solvers, and one basically accumulates experience from the coarse grid

[1]COCS, University of Houston
[2]CDCSP/ISTIL - University Lyon 1, 69622 Villeurbanne, France
{garbey}@cdcsp.univ-lyon1.fr, http://cdcsp.univ-lyon1.fr

to the finest grid computation in order to save eventually cpu time and improve the solution quality. Our ultimate goal, in the design of the algorithm, is therefore to separate completely possible improvements of the CFD code done by the user from post processing/preprocessing methods that should increase the efficiency and trust in the overall numerical process.

2 Verification of Code and Cascade of Grids

We are going to present an attempt [EGH00] to make the additive Schwarz procedure more efficient while keeping as much as possible the simplicity of the original method and using cascade algorithm with three level of grids for acceleration purpose as well as verification. Most of the concepts describe thereafter can be applied in a straightforward way to FE or FV discretization with unstructured grid. But for simplicity, we will report on second-order FD solution of the well-known Bratu problem in a square. The problem is written :

$$N[u] = \Delta u + \lambda e^{u} = O \ in \ \Omega = (0,1)^2, \qquad u_{|\partial\Omega} = 0 \tag{1}$$

The discretized problem has the form,

$$N^h[U] = 0 \equiv \begin{cases} \frac{U_{i+1,j}-2U_{i,j}+U_{i-1,j}}{h_x^2} + \frac{U_{i,j+1}-2U_{i,j}+U_{i,j-1}}{h_y^2} + \lambda\, e^{U_{i,j}} = 0, \\ U_{1,j} = U_{N_x,j} = U_{i,1} = U_{i,N_y} = 0, \\ i = 1, \cdots, N_x - 1, \ j = 1, \cdots, N_y - 1, \end{cases}$$

where h_x (resp. h_y) denotes the space step in x variable (resp y variable). We consider a basic decomposition of the domain into nd-overlapping strips, $(x_A(k), x_B(k)) \times (0,1),\ k = 1 \cdots nd$ with arbitrary overlap of q intervals (meshes) that is : $x_B(k) - x_A(k+1) = q \cdot h_x, \quad k = 1 \cdots nd - 1$, q being a positive integer. At continuous level, the algorithm appears, for $k = 1, \cdots, nd$:

$$N[U^{k,n+1}] = 0$$

$$U^{k,n+1}(x_A(k), \bullet) = U^{k-1,n}(x_A(k), \bullet)$$

$$U^{k,n+1}(x_B(k), \bullet) = U^{k+1,n}(x_B(k), \bullet)$$

with, in our notation, formally $U^{0,n} \equiv U^{nd+1,n} \equiv 0$.
This process constructs a multi-valued piecewise solution because of the overlap and we define the global solution as follows. For $x \in (x_A(k), x_B(k))$:

$$u^{n+1}(x,y) = \sum_{k=1}^{nd} \aleph^k u^{k,n+1}(x,\bullet) + (1 - \aleph^k) \begin{cases} u^{k-1,n+1}(x,\bullet) & \text{if } x > \frac{1}{2}(x_A + x_B)(k) \\ u^{k+1,n+1}(x,\bullet) & \text{if } x \le \frac{1}{2}(x_A + x_B)(k) \end{cases}$$

with $\aleph^k$ a smooth partition of unity that is 1 in $(x_B(k-1), x_A(k+1))$ and 0 outside $(x_A(k), x_B(k))$.

We know that the convergence of this algorithm is linear and very slow. However, there is obviously no need to solve exactly each nonlinear problem in each sub-block, since the domain decomposition is an iterative process. One can optimize then the stopping criterion of the subdomain iterative solver by using a prediction of some norm of the jump at artificial interfaces at the end of each Schwarz iterate [EGH00].

The block solver is actually a nonlinear solver based on the Newton algorithm and a Preconditioned Conjugate Gradient algorithm with Incomplete LU factorization (PCGILU) for the linear system. In our basic strategy to enhance the additive Schwarz algorithm, we introduce three levels of grids, G^m, $m = 1, 2, 3$. For simplicity, we restrict ourselves to embedded grids with discretization ratio 2, but as it will be seen later on, this simplification is not necessary. The classical idea of cascade algorithm is to provide as an initial guess for the iterative solution process on the grid level m an initial guess that is obtained from the discretized solution on the coarser grid $m - 1$. This very old idea is called nested loop when Successive Over Relaxation (SOR) is the block solver. It is known that in terms of arithmetic efficiency the nested loop method should be limited to two levels of grids. The implementation is very simple, since one always go from the coarse grid to the next finer grid level, as long as grid level can be defined properly. The main purpose of using three levels of grids instead of two will be that it will provide enough information in order to proceed to some code verification. We will denote in the following $U^{(m,h)}$ the discrete solution obtained by our iterative method on grid level m. So the algorithm that we propose is to solve, first, with additive Schwarz and iterative block solver, the discretized problem on the grid G^1. Then, we project the solution on grid G^2 and use an interpolation procedure to define the initial guess every where. Second-order linear interpolation seems at first sight a natural tool. The same solution procedure is then reproduced on the grid G^2. If we basically project the solution obtained on G^2 into G^3, we miss a very important property of our approximation method. The discretized solution of Bratu problem should converge to the exact solution with order two accuracy in space as $h = (h_x, h_y)$ goes to zero, that is $U - U^h = O(h_x^2 + h_y^2)$. Using the classical Richardson extrapolation principle reported as in Roache [R98] for code verification, we may therefore produce an initial guess for the iterative solution procedure on grid G^3that is much better than a basic projection of U^2 onto G^3. We define the initial guess for the iterative solution on G^3 to be

$$U_1^{(3)} = 4/3U^{(2,h)} - 1/3U^{(1,h)}. \tag{2}$$

We encounter two difficulties. First of all, $U_1^{(3)}$ is defined only on grid $G^{(1)}$. We do need therefore to interpolate $U_1^{(3)}$ with an interpolation procedure that keeps the Richardson extrapolation procedure effective on $G^{(3)}$. If $U^{(2,h)}$ and $U^{(1,h)}$ are known at second-order on $G^{(1)}$, we can expect $U_1^{(3)}$ to be at least a third-order approximation of $U^{(3,h)}$ on $G^{(1)}$. Actually with regular grids of constant space steps, one obtains a fourth-order approximation. In order to preserve as much as possible the quality of this information, we should use a third-order interpolation method to extend our grid solution $U_1^{(3)}$ from $G^{(1)}$ to $G^{(3)}$. In our case, we have applied cubic bilinear interpolation as well as spline interpolation. A second difficulty is that $U^{(1,h)}$ and $U^{(2,h)}$ are computed with an iterative procedure. The Richardson extrapolation principle applies to exact discrete solution. The error is then given by a truncation error formula based on Taylor expansion of the discrete operator with respect to discrete parameter h for which the leading coefficients are a priori independent of h. As opposed to the iterative solution procedure on a single grid, one therefore needs to compute an approximation of $U^{(j,h)}$, $j = 1, 2$ on the grids $G^{(j)}$, $j = 1, 2$, with a residual that is of the order the expected rate of convergence of Richardson extrapolation method, that is at least h^3. This procedure combining Richardson extrapolation and high order interpolation allows then to produce a good initial guess for the iterative solution on grid $G^{(3)}$, and the code can be verified by simply proceeding with the computation of $U^{(3,h)}$. In principle, the number of Schwarz iterates should be small if the construction of $U^{(3)}$ and the resolution of $U^{(j,h)}$, $j = 1, 2$ are correct. The solution process is however robust because $U^{(3)}$ is only used as an initial guess. In the mean time the order $B(x)$ of the method can be checked with formula:

$$B = log_{10}|U^{(1)} - U^{(2)}|/|U^{(2)} - U^{(3)}|/log_{10}(2), \tag{3}$$

at each point x.

This formula can be used in combination to the computation of the residual in order to verify the stopping criterion of the iterative computation of the solution on the finest grid $G^{(3)}$that in practice dominates the overall cost of the method.

In order to illustrate the method, we report thereafter on the numerical solution of the Bratu problem with $\lambda = 6$. Our computation, realized with matlab, is rather modest. The fine grid $G^{(3)}$ has 65×61 grid points. The main thrust of this method is not the arithmetic efficiency but its simplicity. However, we compare the number of floating point operations realized with our algorithm to the flops performance of PCGILU, no domain decomposition, one single fine grid and the trivial initial guess on $G^{(3)}$. Parallel efficiency is analogous to [PARCFD] and it is known that the additive Schwarz scales well even on MIMD parallel systems with slow network as long as the load per processors is high enough .

The Schwarz algorithm is applied with minimum overlap on grid $G^{(j)}$, $j = 1, 2$. The size of the overlap is chosen on $G^{(3)}$ in order to minimize the overall flops performance. The order of convergence of the method $B(x)$ is computed on the coarse grid; as expected, we observe that $B(x)$ is 2 within 5%. Table 1 shows the global efficiency of the iterative solver by listing the total number of Mflops used on cascade algorithm, and the number of Schwarz iterates for each grid level. It should be noticed that monitoring the order of the method with formula (3) may avoid premature iteration stops of the Schwarz iteration process. It is also interesting to notice that the overall number of flops to reach the correct solution is relatively insensitive to the number of subdomains, but that the number of Schwarz iterates grows as expected with the number of subdomains. This has an impact on the parallel efficiency of the method that is usually limited by the network of communications, i.e. the more messages i.e. Schwarz iterates, the less is the parallel efficiency. To conclude this section we observe that our Bratu problem's example is characterized by a very smooth solution and our results cannot be reproduced for non smooth problems. For example, the Richardson extrapolation methods break down for the cavity flow problem with a finite difference computation of $\omega - \psi$ formulation even for modest Reynolds number of order 100, because of the existence of a singularity in the flow field at the corner [SGAW01] We proceed therefore with a modification of the Richardson extrapolation method that may work with numerical method with varying order of approximation.

Number of subdomains	2	3	4	5	6
Flops ratio versus no DD method	1.69	1.06	0.95	0.95	0.99
Mflops used	89.5	142	120	120	115
Number of Schwarz iterates on G1	79	100	119	150	202
Number of Schwarz iterates on G2	88	131	149	168	196
Number of Schwarz iterates on G3	8	15	12	13	14

Table 1: *Cascade-Newton-Schwarz on 2D Bratu problem with 65 $\times$ 61 unknowns.*

3 A Generalized Extrapolation Method

In order to present the idea, we restrict ourselves to problems in one space dimensions with regular grids. We refer to the report [GS01] to provide more details on the analysis and application results in higher space dimensions. Let us consider two continuous functions $U(x,h_1)$ and $U(x,h_2)$ that are approximations of a continuous function $U(x),\ x \in (0,\pi)$. A general consistent linear extrapolation formula writes:

$$\alpha\, U^{(1,h)} \ +\ (1-\alpha)\, U^{(2,h)}.$$

If the approximation method to build $U(x,h)$ is order q, the extrapolation formula becomes

$$\tilde{U}(x,h) = \frac{h_2^q U(x,h_1) - h_1^q U(x,h_2)}{h_2^q - h_1^q}.$$

Let us consider now a linear differential problem $L[U]=0$, and a sequence of consistent approximation L^h obtained with Finite Differences or Finite Volume for instance. The following discussion is fairly general and can be reproduced for variational formulations. Let us suppose that the consistency error in some norm is of order q, and can be expanded as follows:

$$L^h[U(x,h_j] = R_q(x)h_j^q + R_{q+1}(x)h_j^{q+1} + \cdot. \tag{4}$$

Then q is the constant that minimizes the *asymptotic* order of the residual $L[\alpha U(x,h_1) + (1-\alpha)U(x,h_2)]$, with $\alpha = \frac{h_2^q}{h_2^q - h_1^q}$. Using a stability estimate on L^h, one can then prove that the Richardson extrapolation $\tilde{U}$ is a better approximation of U than any of the approximation $U(x,h_j),\ j=1,2$.

The main difficulty in practice, is that h_1 and h_2 may not be small enough to produce residual for which the first-order term R_q in the expansion (4) is significantly greater than the next order term $R_{q+1}h$. Even worst, R_q and R_{q+1} are space dependent functions and the approximations can behave as a q-order approximation in some subdomain and a $q+1$-order approximation elsewhere. Classical Navier-Stokes provides a large collection of such examples when boundary layer or transition layer occur. In addition, lift and drag calculations may not require method with uniform order approximation.

We propose therefore to define the following problem:
find $\alpha(x)$ in some vector space to be defined later on, that minimizes the residual:

$$L^h[\alpha U(x,h_1) + (1-\alpha)U(x,h_2)], \tag{5}$$

in some norm to be defined.

$\tilde{U} = \alpha(x)U(x,h_1) + (1-\alpha(x))U(x,h_2)$ will be then our new extrapolation formula.

We have now several difficulties: first, we deal with grid functions $U^{(j,h)}$ instead of continuous approximation $U(x,h_j)$. Second, following the lines of Sect 2, the practical purpose of this optimization problem is to provide a good solution on the fine grid G^3. Therefore the computation of the residual on G^3 requires a high order interpolation of $U^{j,h}, j = 1,2$ that is robust with respect to differentiation. Third the optimization problem should be well posed and its solution should cost much less than the fine grid computation on G^3.

We propose in the following a basic algorithm that seems to be a good candidate to improve the basic Richardson extrapolation procedure. We look for α as the shifted Fourier expansion

$$\alpha_{N_F} \; = \; \alpha_0 + \alpha_1 \; cos(x) + \Sigma_{j=2..N_F}\alpha_j \; sin((j-1)x), \tag{6}$$

that is solution of the least square problem

$$L^h[\alpha U^{(1,h)} + (1-\alpha)U^{(2,h)}] = 0, \; on \; grid \, G^3. \tag{7}$$

We observe that the solution of the least square problem $\alpha_N = \alpha \; on \; G^3$ with $\alpha \; in \; C^2(0,\pi)$ is a second-order approximation in maximum norm of α on G^3, and a third-order approximation away from the end boundaries. We observe however that at location where $\delta U = U^{(1,h)} - U^{(2,h)}$ is close to zero, and $U^{(1,h)}, \; U^{(2,h)}$ are not close to the exact discrete solution $U^{(3,h)}$ within asymptotic order $O(\delta U)$, we have a singularity. In practice, there is not much improvement that one may expect from classical extrapolation formula in such situation. We therefore can detect local failure of our grid solutions G^1 and G^2 with (6, 7) but not fix it. We found in practice more robust to use three levels of grids $G^j, j = 1..3$ in order to predict the fine grid solution on G^4 with the following least square problem:
find α and β *with expansion similar to (6), that is solution of the least square problem*

$$L^h[\alpha U^{(1,h)} + \beta U^{(2,h)} + (1-\alpha-\beta)U^{(3,h)}] = 0, \; on \; grid \, G^4. \tag{8}$$

Some details on the analysis of this method and its application to Navier Stokes problem can be found in [GS01]. But in this proceeding paper, we would like to show with the following classical Burgers problem,

$$-\epsilon u_{xx} - (\frac{1}{2}u^2)_x = 0, \; on \; (0,\pi), u(0) = \pi/2 + \epsilon, \; u(\pi) = -\pi/2 + \epsilon,$$

that even if the Richardson extrapolation methods fails to improve the underlines grid solutions $U^{(j,h)}, \;\; j = 2,3$, our least square extrapolation method may give a much better solution. Obviously since Burgers is a nonlinear problem, we apply recursively a least square linear solve to the linearized problem with a Newton like loop. Fig 1 shows the error curves with respect to the exact discrete solution on G_4, with central finite differences. The number of grid points on G^1, (resp. $G^2, \; G^3$) is $N_1 = 11$, (resp. $N_2 = 19$, $N_3 = 25$) and $\epsilon = 0.1$. As an interpolant for the grid functions $U^{(j,h)}$, we have used spline interpolant (Fig 1) as well as shifted Fourier interpolant similar to (6). N_F in the shifted Fourier expansion of α is limited to 6 and the number of Newton iteration to 3. In each case, the Richardson extrapolation assuming a first-order method (R1 curve), or a second-order method (R2 curve) are less accurate than the solution on the fine grid G3. Nevertheless the least square extrapolation improves significantly the accuracy of this fine grid solution. These results are quiet good and have been extended to multidimensional problems [GS01].

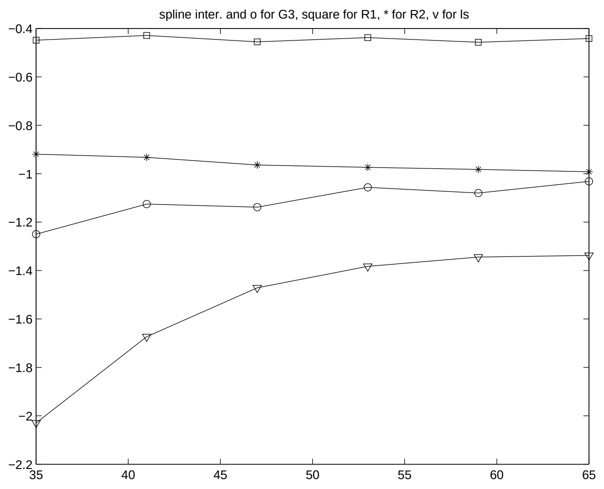

Figure 1: The x-axis is for the number of grid points on G_4. The y-axis gives the error in maximum norm in log_{10} scale on grid G_4.

4 Conclusions:

We have briefly presented, first in this paper some background on code verification, second a practical way of combining efficiency in the solution process and some code verification by using standard additive Schwarz algorithm with cascade method. In the development of our solution, we restrict ourselves carefully to a method that should be easily generalized to non-structured grid and FV or FE discretization. We have shown some practical limitation on the use of Richardson extrapolation and presented a least square extrapolation variant that looks promising for CFD application [S94]. Finally, we observe that multilevel methods give us the opportunity to provide solutions on several grids and that it should be an important tool used to understand better the convergence accuracy of a CFD code for which it is rare that all mathematical hypothesis are fulfilled correctly.

References

[BT89]D.P. Bertesekas and J. N. Tsitsiklis, *Parallel and Distributed Computation- Numerical Methods*, Prentice Hall 1989.

[EGH00]A. Ecer, M. Garbey and M. Hervin, Proceedings of 12th international conference Parallel CFD 2000, North-Holland publisher, Trondheim.

[GS01]M. Garbey and W. Shyy, *A Least Square Richardson Extrapolation Method for PDE's*, preprint CDCSP, 2001.

[OBA95]W.L. Oberkampf, F.G. Blottner and D.Aeshliman, Methodology for Computational Fluid Dynamics Code Verification and Validation, 26th AIAA Fluid Dynamic Conference, June 19-22, 1995/ San Diego, CA, AIAA Paper 95-2226.

[PARCFD]A. Ecer "et al.", *Parallel CFD test case*, Series of Parallel CFD Conferences, http//www.parcfd.org

[R98]P.J. Roache, *Verification and Validation in Computational Science and Engineering*, Hermosa Publishers, Albuquerque, New Mexico, 1998.

[S94]W. Shyy, *Computational Modeling for Fluid Flow and Interfacial Transport*, New York, Elsevier 1994.

[SGAW01]W.Shyy, M.Garbey, A.Appukuttan and J.Wu, *Evaluation of Richardson Extrapolation in Computational Fluid Dynamics*, to appear in Numerical Heat Transfer, PartB: Fundamentals.