Further investigation on the solution of the incompressible Navier-Stokes equations by Krylov subspace and multigrid methods

Report 93-93

S. Zeng
C. Vuik
P. Wesseling

TU Delft
Faculteit der Technische Wiskunde en Informatica
Faculty of Technical Mathematics and Informatics

Technische Universiteit Delft
Delft University of Technology

ISSN 0922-5641

Copies of these reports may be obtained from the bureau of the Faculty of Technical Mathematics and Informatics, Julianalaan 132, 2628 BL Delft, phone +3115784568.
A selection of these reports is available in PostScript form at the Faculty's anonymous ftp-site. They are located in the directory /pub/publications/tech-reports at ftp.twi.tudelft.nl

Further Investigation on the Solution of the Incompressible Navier-Stokes Equations by Krylov Subspace and Multigrid Methods

S. Zeng C. Vuik P. Wesseling

September 28, 1993

Abstract

As a sequel to [7], a further study is carried out. Three iterative methods, that is, GMRESR, which consists of GCR with GMRES as inner loop, GCR with multigrid as inner loop and a multigrid method, are investigated by means of numerical experiments. The multigrid algorithm in the second and the third methods uses an ILU smoother. Numerical experiments are performed on a workstation and a vector mini supercomputer, and robustness and efficiency of the three methods are studied. Efficiency of the second and the third methods are compared with the corresponding two methods discussed in [7], where an alternating Jacobi line smoother is used in the multigrid algorithm. It appears that good methods would come out of some suitable combinations of GCR type methods and multigrid methods.

1 Introduction

In [7] three iterative methods, i.e., GMRES ([2]) with ILU preconditioning (Method 1^*), GCR ([1]) with multigrid as its inner loop (Method 2^*) and a (standard) multigrid method (Method 3^*), are investigated for the solution of the incompressible Navier-Stokes equations discretized by a finite volume method on staggered grids in general coordinates. Concerning robustness and computational efficiency of the three methods, the following observations are obtained. Method 1^* and Method 2^* are equally robust; Method 3^* is the least robust one. On a scalar computer, Method 1^* is the most efficient on coarser grids, while Method 2^* and Method 3^* become more efficient as the grid is refined. Combining the advantages of Method 1^* and Method 3^*, Method 2^* seems very promising.

The above conclusions, however, are drawn from results obtained on a scalar machine; on vector computers, results may be different. The reason is that most arithmetic operations involved in a GMRES and GCR type method are matrix-vector multiplications and vector-vector additions, which are easily vectorizable. As a consequence, such methods would have higher efficiency, especially when the grid is fine, which leads to larger vector lengths. But for a multigrid method, use of vectors of shorter lengths is inevitable, since use of coarser grids is necessary. This hampers multigrid efficiency on vector computers. Combination of a GCR type method with multigrid, using multigrid as inner loop, will suffer from the same disadvantage. On the other hand, a good multigrid method, in general, has a reduction factor (almost) independent of grid size, whereas the performance of GMRES and GCR type methods deteriorates as the grid is refined, due to the fact that the number of iterations is increased. This raises the question whether for GMRES and GCR type methods the gain from increasing computational speed due to better vectorization can balance the loss due to the growth of number of iterations when refining the grid, and whether multigrid methods as well as combinations of GCR and multigrid can still achieve high efficiency on vector machines as they do on scalar machines when the grid gets finer. To tackle these questions is the purpose of this paper.

Apart from efficiency, robustness is also an important aspect of a numerical method. For a multigrid method, using powerful smoothers such as those of ILU type makes it robust and this is necessary for difficult problems as in general coordinates, where simple smoothers do not work very well and often fail. Furthermore, more robust smoothers can give smaller reduction factors and so need fewer iterations. But this does not simply imply higher efficiency, because usually robust smoothers have less vectorization potential than simpler smoothers and one iteration is therefore more costly. Efficiency consideration cannot be ignored for a practical method. So this brings us back to the consideration of efficiency: whether more powerful smoothers can still be acceptably efficient while improving robustness.

In this paper, based on the above considerations, we investigate three methods: a GMRESR method which uses a preconditioned (with RILU) GMRES method as inner loop (Method 1), and GCR using multigrid as inner loop (Method 2), and a multigrid method (Method 3). In the multigrid algorithm used in Methods 2 and 3, an ILU smoother is employed. The three methods differ from the three methods studied in [7] as discussed above. The reason for this change is that Method 1 is more efficient and usually more robust than Method 1^*, and the same is true for Methods 2, 3 and Methods 2^*, 3^*, as will be seen later.

It is thought better to let the best methods available compete.

The outline of this paper is as follows. For principles of the methods, we refer to [7] and the references therein; here only those that have not been described in [7] will be presented in section 2. The results as well as analyses and comparisons are given in section 3. Section 4 summarizes the observations.

2 Solution Methods

2.1 GMRESR with GMRES as Inner Loop

The GMRESR method is described in [3]. Let the linear system to be solved be

$$\mathbf{A}\mathbf{x} = \mathbf{b}. \tag{2.1}$$

The inner loop in GMRESR, which is briefly denoted as $C(\mathbf{A}, \mathbf{r})$ in [7], gives an approximation of the solution to the equation system

$$\mathbf{A}\mathbf{u} = \mathbf{r}, \tag{2.2}$$

where $\mathbf{r} = \mathbf{b} - \mathbf{A}\mathbf{x}$ is the residual. $C(\mathbf{A}, \mathbf{r})$ is regarded as preconditioning. Here $C(\mathbf{A}, \mathbf{r})$ is obtained by K GMRES iterations with RILU preconditioning, as explained in [5] and [6], with $K = 5$. The maximum number $nt - 1$ of outer iterations is taken to be 10 for the solution of the momentum equations and 20 for the solution of the pressure equation. When the number of outer iterations is reached and the required accuracy is not yet obtained, the so-called *min alfa* version ([4]) of the truncated GMRESR method is used, and computation is continued, until the required accuracy is obtained.

2.2 GCR with Multigrid as Inner Loop

In this method, the approximation $\tilde{\mathbf{u}}$ of (2.2) is given by applying one F-cycle of multigrid iteration, as explained below. Here we use the *trunclast* version (see also [4]) of the truncated GCR and set nt equal to 15, which is never reached, however, for the accuracy requirement that we set. Thus the outer loop of GCR is not truncated and is actually the full GCR.

2.3 The Multigrid Method

An ILU smoother is used, using the standard 9-point non-zero pattern. The momentum equations are smoothed in a decoupled way, successively in all directions. Update of variables is carried out with damping; the damping factor ω is fixed at 0.8, based on numerical experiments. The F-cycle is employed. The coarsest grid is fixed at 2×2, on which a direct solver is used to solve the equation system.

3 Numerical Experiments

The four test problems of [7], namely the square driven cavity problem, the non-uniform square driven cavity problem, the skewed driven cavity problem and the L-shaped driven cavity problem, are used in numerical experiments. For convenience of reference, these problems are designated as Problems 1 to 4, respectively. Three time intervals, namely $\Delta t = 0.0625$, 0.125, 0.25, are considered, but the number of time marching steps is fixed at 40. We solve the equation systems at each time step until the ratio of the present norm $\|\mathbf{r}\|$ of the residual to the norm $\|\mathbf{r}_0\|$ of the initial residual at the beginning of the present timestep satisfies $\|\mathbf{r}\|/\|\mathbf{r}_0\| < tol$, with $tol = 10^{-4}$ for the momentum equations and $tol = 10^{-6}$ for the pressure equation. Computations are carried out on an HP 735 workstation and a Convex 3840 mini supercomputer. When operating in scalar mode, the Convex is slower than the HP.

In Tables 3.1 to 3.4 are presented the total CPU time t_t, the CPU time t_v for the solution of the momentum equations and the CPU time t_p for the solution of the pressure equation, on the HP machine. All CPU times are measured in seconds. At the final time step, the number of outer iterations for Method 1 and Method 2 and the number of multigrid iterations for Method 3 are counted and are denoted as k_v and k_p for the solution of the momentum equations and the pressure equation, respectively. The corresponding reduction factors, ρ_v and ρ_p of the multigrid algorithm, used in Method 2 and Method 3, are also listed for the last iteration at the final time step.

From these tables it is obvious that all three methods have better robustness and efficiency than the corresponding three methods studied in [7]. It is easier to compare the performance of the three methods by means of figures instead of tables of data. So in Figure 3.1 through Figure 3.8, the CPU times t_t and t_p are plotted for $n \times n$ grids, with n along the abscissas and the ordinates indicating the CPU time per grid point for the 40 time steps. We see that for the solution of the pressure equations, the curves (in the figures on the right columns) for different Δt are almost identical in each method. This is natural since the pressure matrix does not change with Δt for a problem. The minor differences are probably caused by not very accurate measurement of the CPU times.

It is clear that for Method 2 and Method 3, the CPU time per grid point is almost constant, independently of grid size and Reynolds number. However for Method 1, the CPU time increases significantly as the grids are refined, especially for the solution of the pressure equation and for the low Reynolds number cases. There is an exception for Problem 2 at $Re = 1000$, where Method 1 does not loose efficiency when the grid becomes finer (see the left-top figure in Figure 3.4); the reason is not clear. In this figure, there are some test points missing due to divergence. Usually on coarser grids Method 1 is the most efficient, but on finer grids Method 2 and Method 3 are more economical. So for a computation case there exists a cross-over point with respect to grid size, beyond which Method 2 and Method 3 are more efficient than Method 1. It seems that Method 1 is more suitable for solving the momentum equations, especially at high Reynolds numbers where some superlinear convergence occurs, and Methods 2 and 3 are preferable for solving the pressure equation. Compared with Method 3, Method 2 seems not to be so superior, and sometimes gives even somewhat worse performance. But the loss of efficiency in Method 2 for some cases is very small. This indicates that when a multigrid algorithm is sufficiently powerful, the gain by accelerating it with GCR type methods is small, but still helpful, as we will see when Methods 2 and 3 are compared on the Convex. On the other hand, when a multigrid algorithm is not so strong, as in the case of Method 3*, combination of it with GCR, leading to Method 2*, turns out

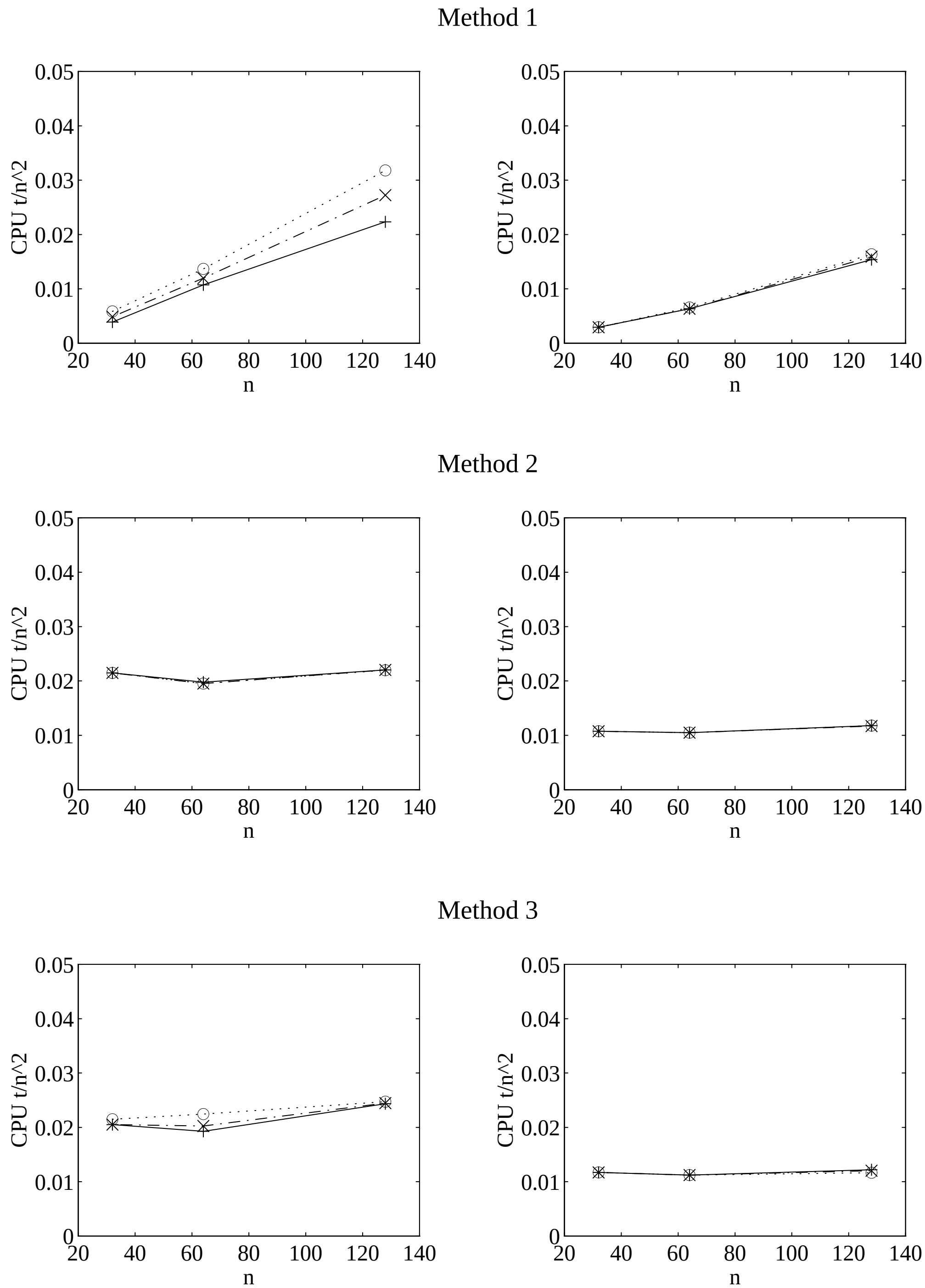

Figure 3.1: Problem 1 at $Re = 1$ on the HP: CPU time per grid point on different grids for the solution of the momentum equations (left) and for the solution of the pressure equation (right). Solid lines and plus marks: $\Delta t = 0.0625$; dashdot lines and x-marks: $\Delta t = 0.125$; dotted lines and circles: $\Delta t = 0.25$

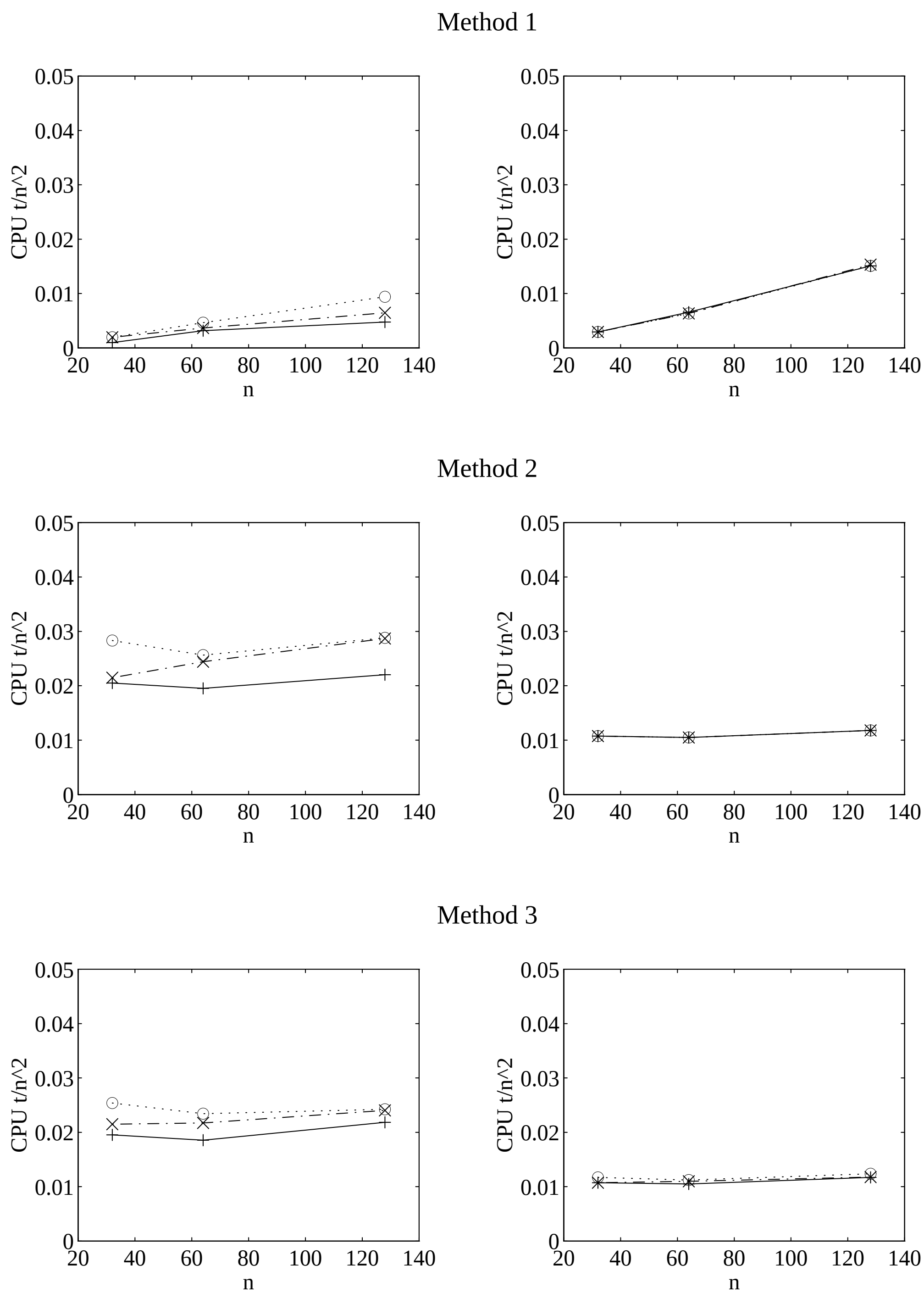

Figure 3.2: Problem 1 at $Re = 1000$ on the HP: CPU time per grid point on different grids for the solution of the momentum equations (left) and for the solution of the pressure equation (right). Solid lines and plut marks: $\Delta t = 0.0625$; dashdot lines and x-marks: $\Delta t = 0.125$; dotted lines and circles: $\Delta t = 0.25$

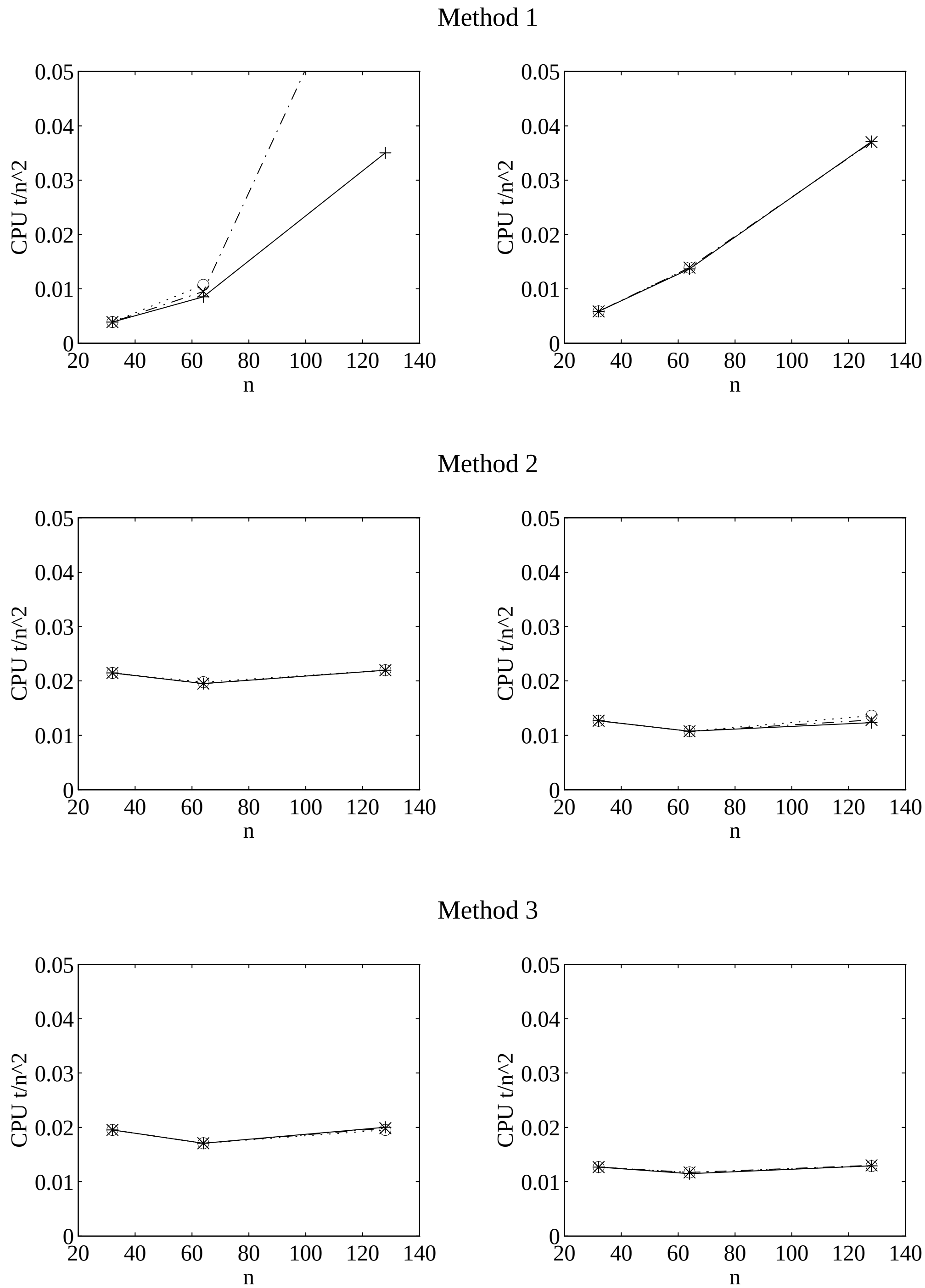

Figure 3.3: Problem 2 at $Re = 1$ on the HP: CPU time per grid point on different grids for the solution of the momentum equations (left) and for the solution of the pressure equation (right). Solid lines and plus marks: $\Delta t = 0.0625$; dashdot lines and x-marks: $\Delta t = 0.125$; dotted lines and circles: $\Delta t = 0.25$

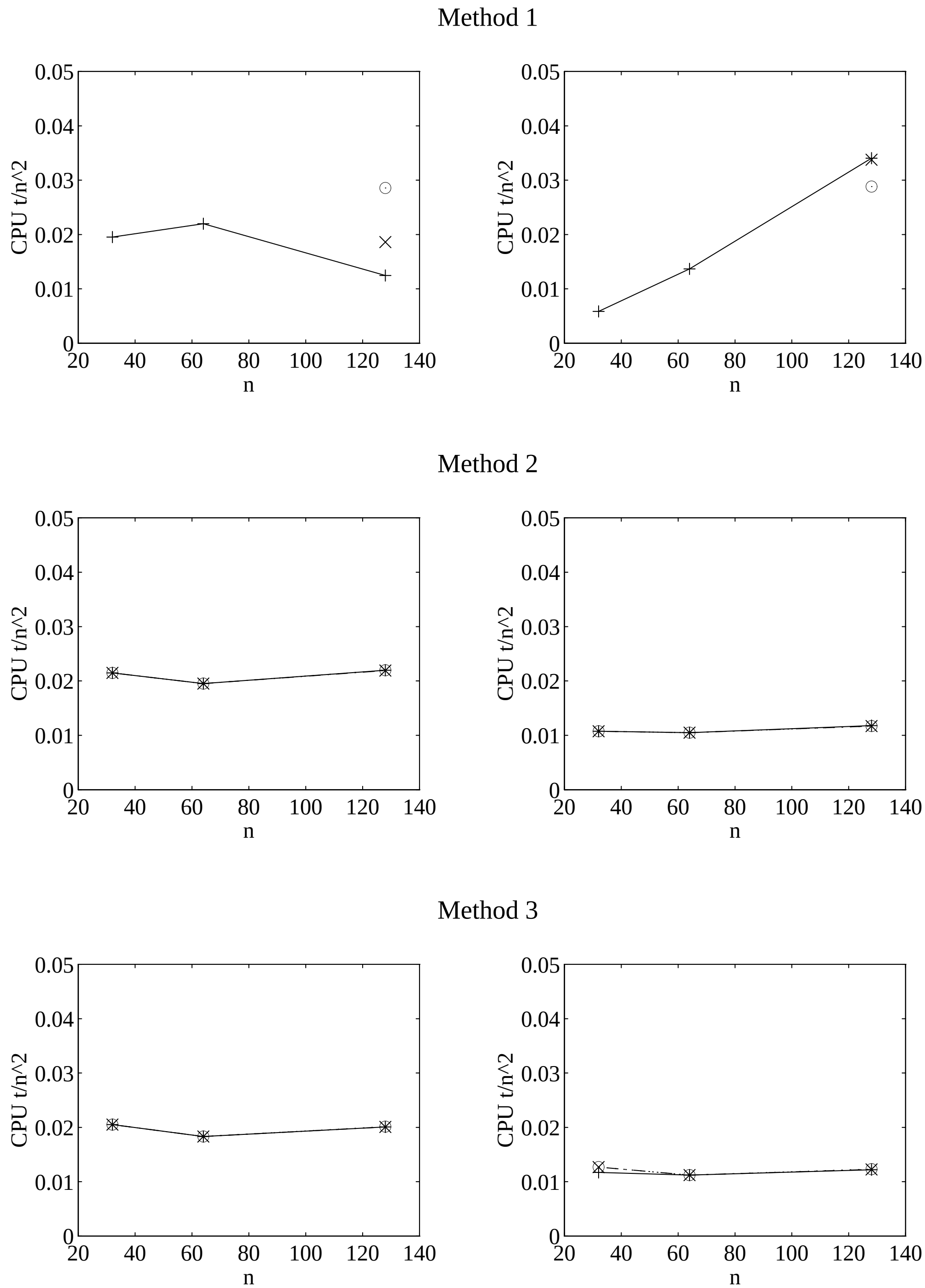

Figure 3.4: Problem 2 at $Re = 1000$ on the HP: CPU time per grid point on different grids for the solution of the momentum equations (left) and for the solution of the pressure equation (right). Solid lines and plus marks: $\Delta t = 0.0625$; dashdot lines and x-marks: $\Delta t = 0.125$; dotted lines and circles: $\Delta t = 0.25$

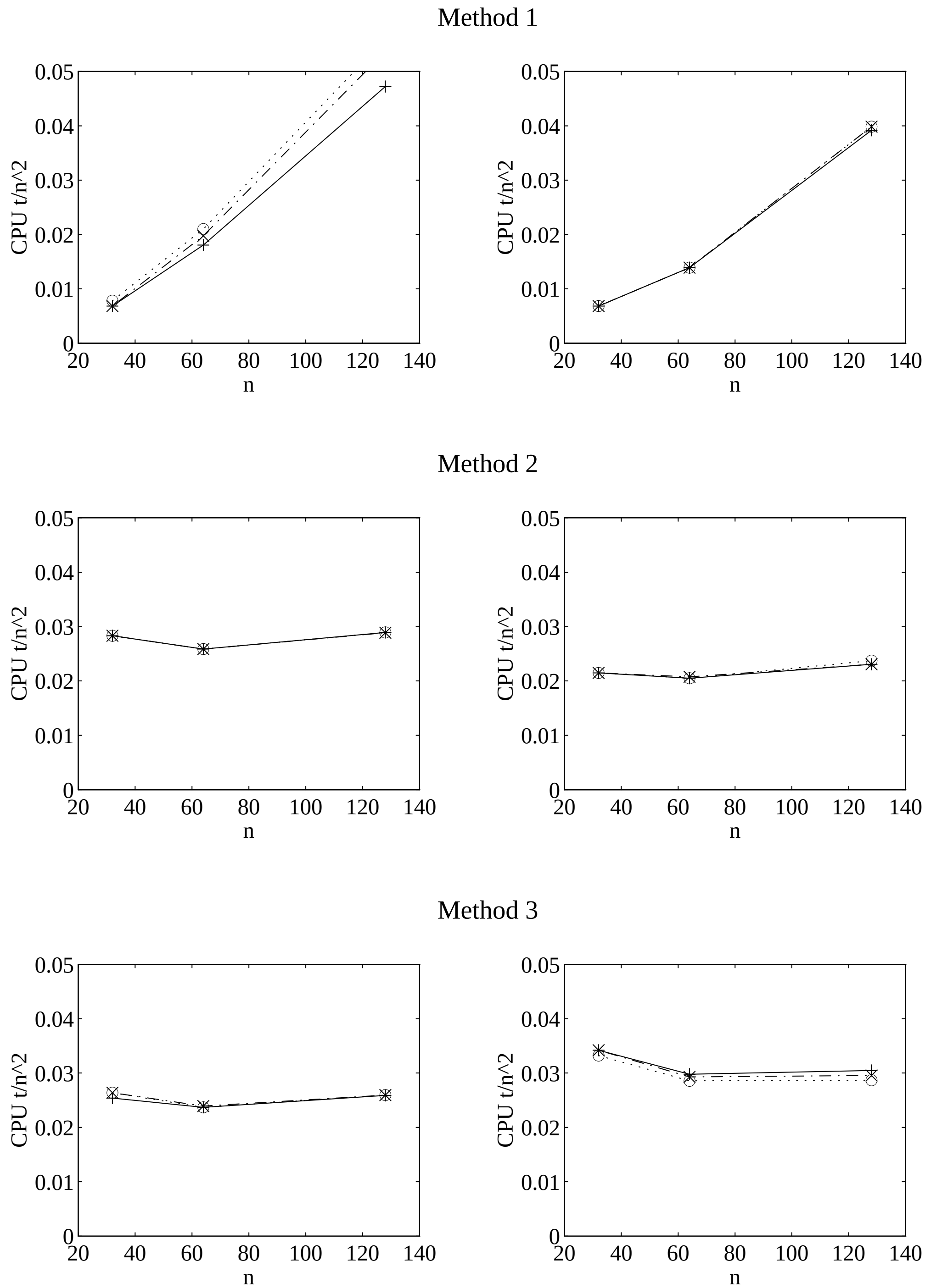

Figure 3.5: Problem 3 at $Re = 1$ on the HP: CPU time per grid point on different grids for the solution of the momentum equations (left) and for the solution of the pressure equation (right). Solid lines and plus marks: $\Delta t = 0.0625$; dashdot lines and x-marks: $\Delta t = 0.125$; dotted lines and circles: $\Delta t = 0.25$

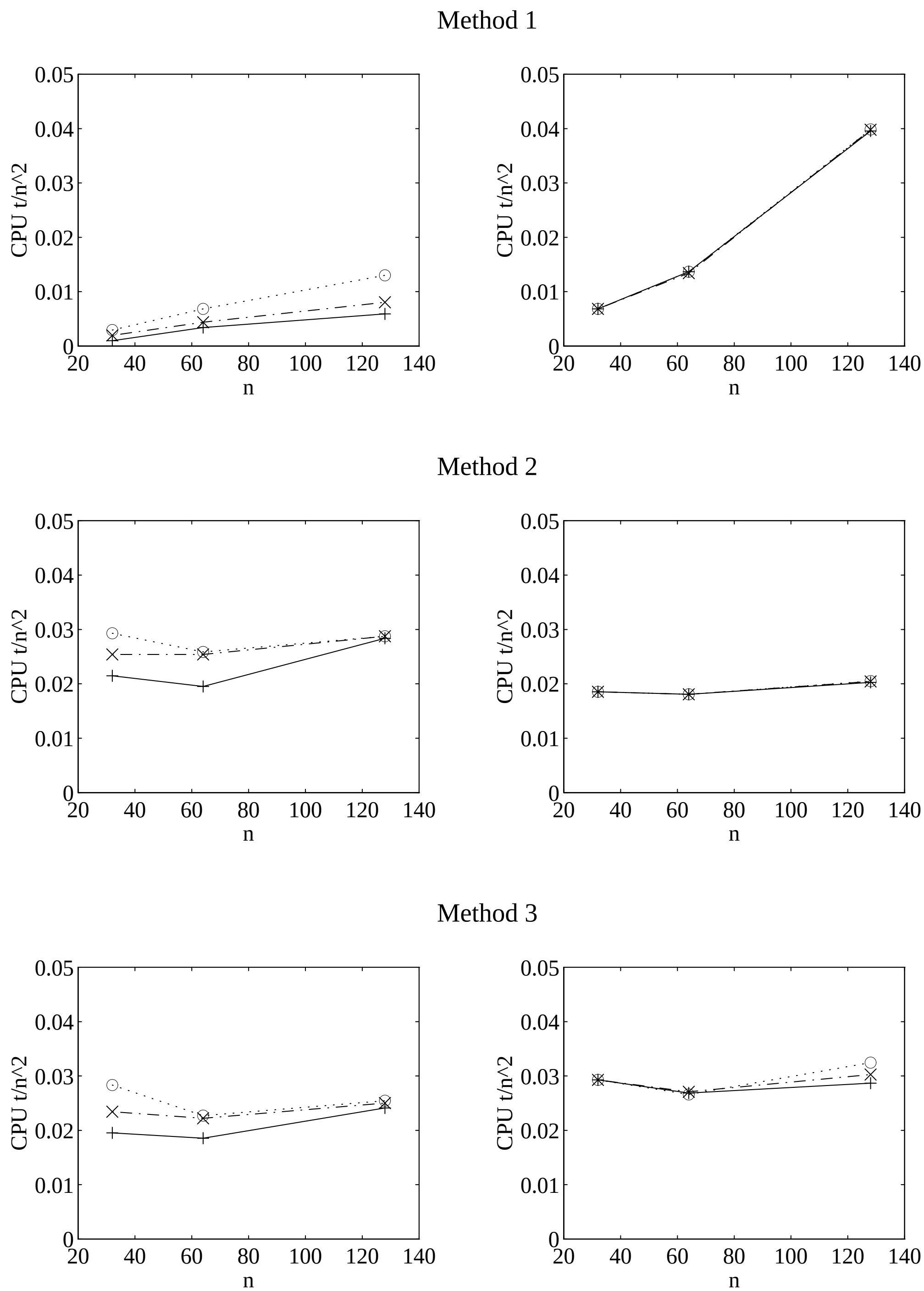

Figure 3.6: Problem 3 at $Re = 1000$ on the HP: CPU time per grid point on different grids for the solution of the momentum equations (left) and for the solution of the pressure equation (right). Solid lines and plus marks: $\Delta t = 0.0625$; dashdot lines and x-marks: $\Delta t = 0.125$; dotted lines and circles: $\Delta t = 0.25$

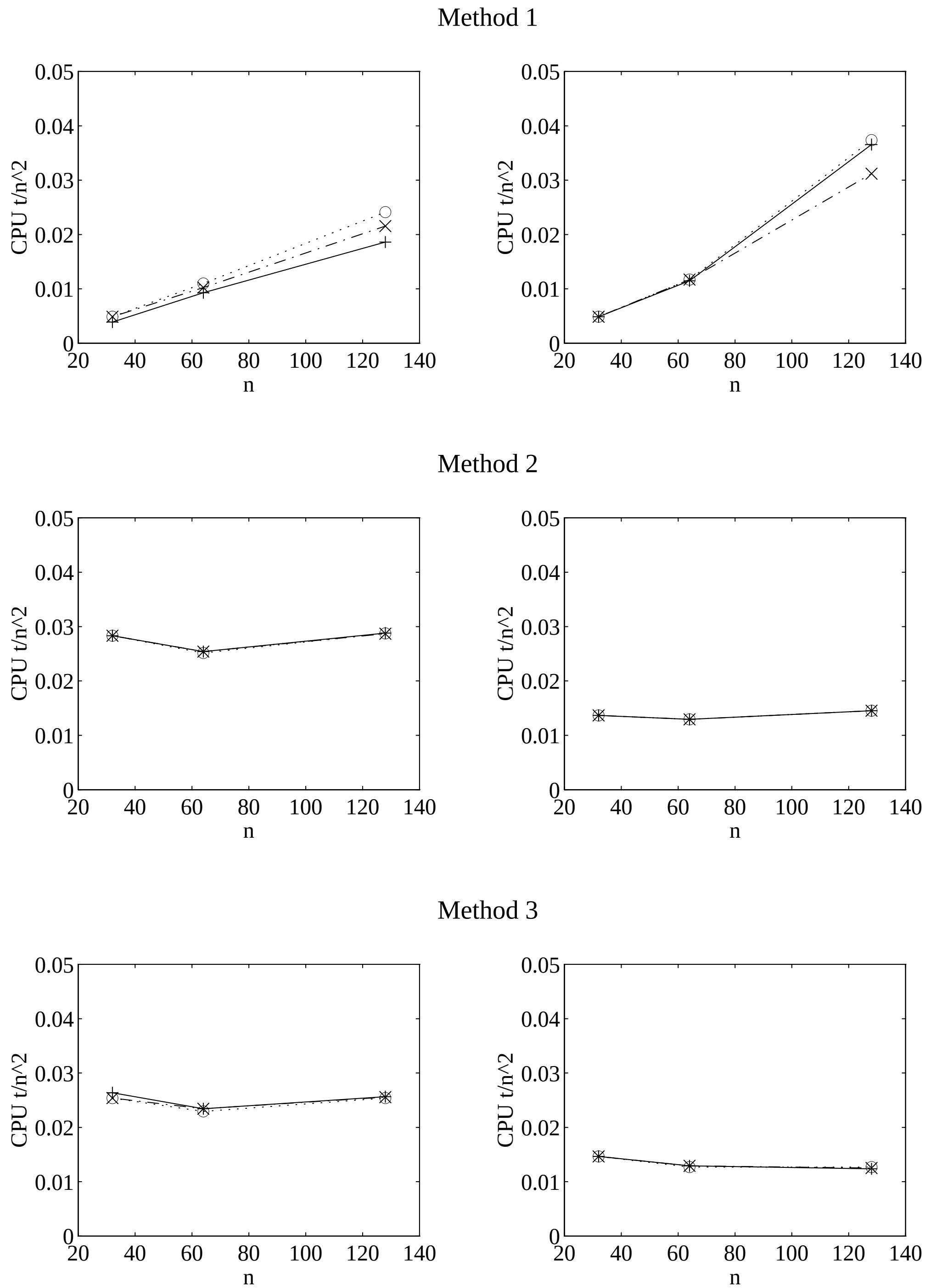

Figure 3.7: Problem 4 at $Re = 1$ on the HP: CPU time per grid point on different grids for the solution of the momentum equations (left) and for the solution of the pressure equation (right). Solid lines and plus marks: $\Delta t = 0.0625$; dashdot lines and x-marks: $\Delta t = 0.125$; dotted lines and circles: $\Delta t = 0.25$

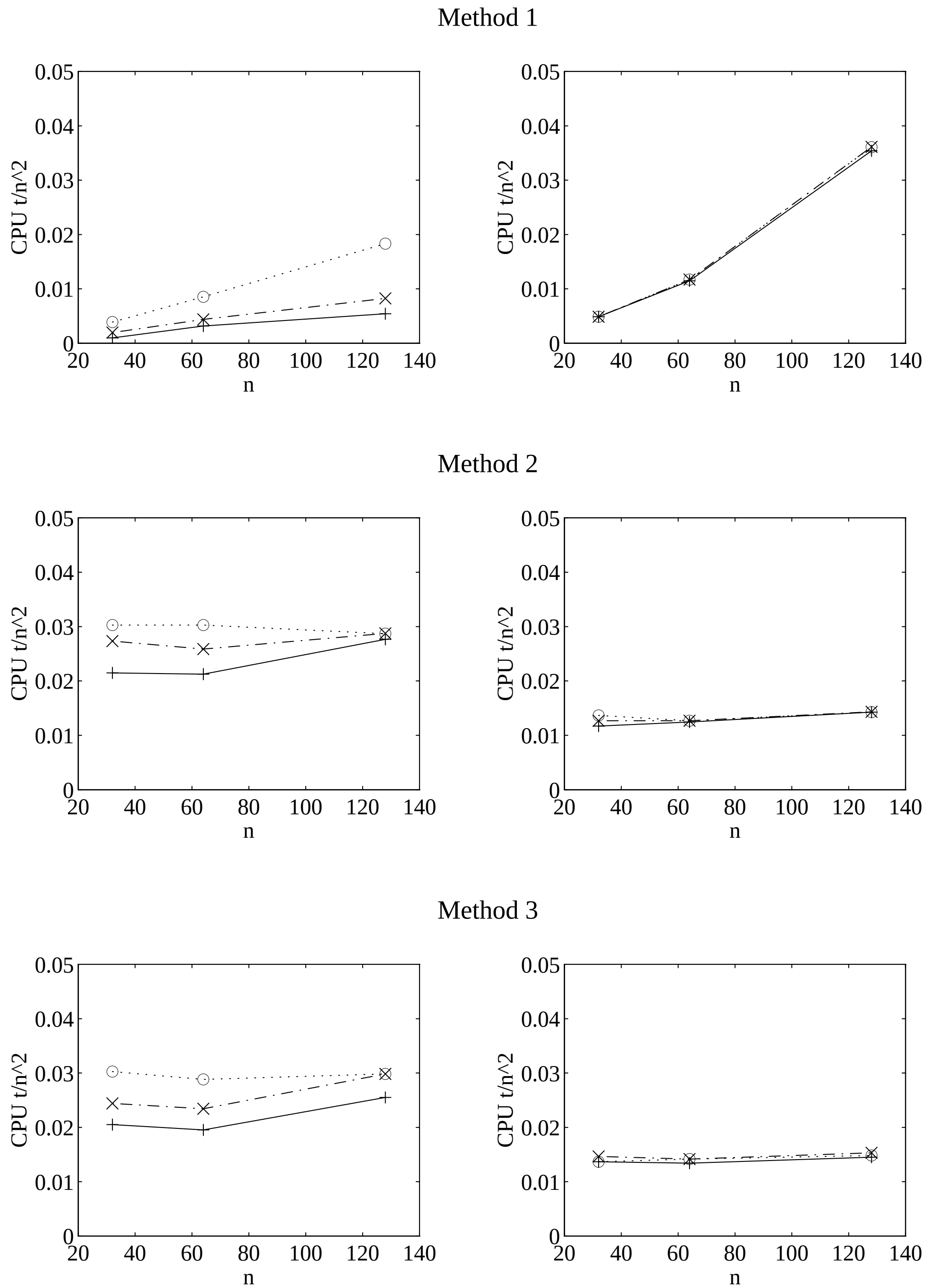

Figure 3.8: Problem 4 at $Re = 1000$ on the HP: CPU time per grid point on different grids for the solution of the momentum equations (left) and for the solution of the pressure equation (right). Solid lines and plus marks: $\Delta t = 0.0625$; dashdot lines and x-marks: $\Delta t = 0.125$; dotted lines and circles: $\Delta t = 0.25$

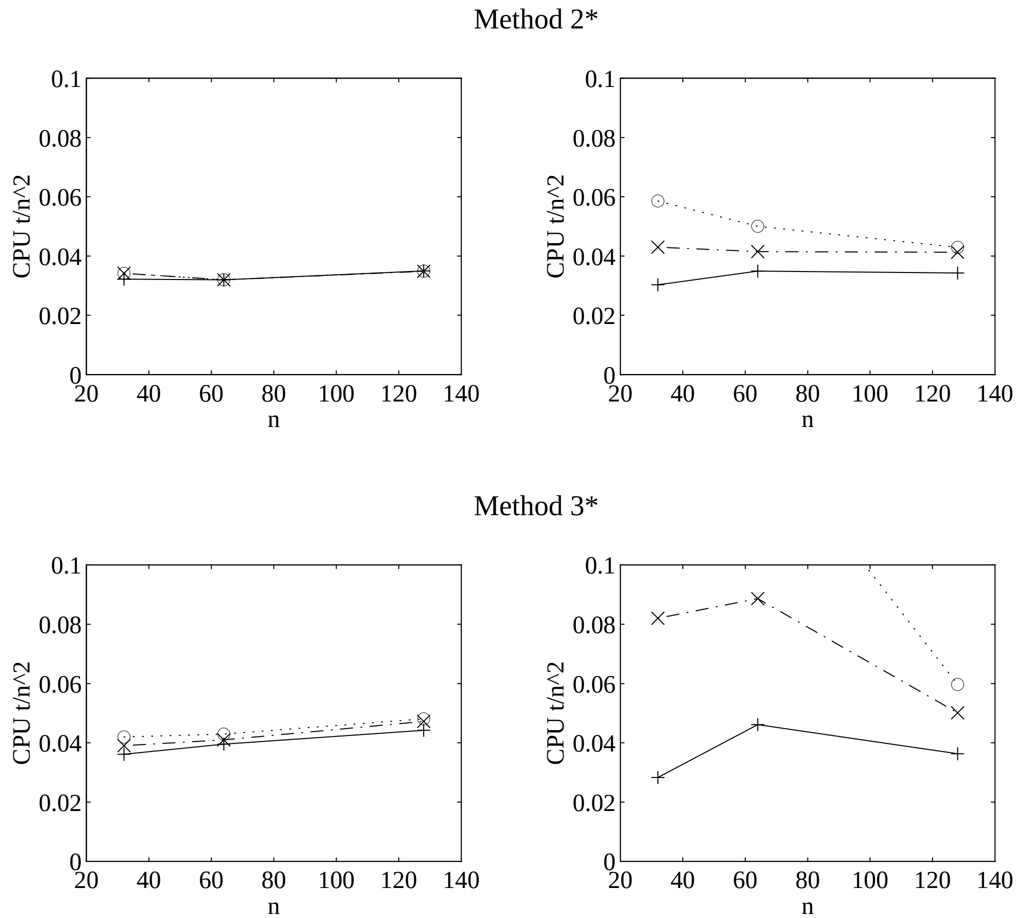

Figure 3.9: CPU time per grid point against grid size for the solution of the momentum equations for Problem 1 on the HP. The Left column: $Re = 1$; the right column: $Re = 1000$. Solid lines and plus marks: $\Delta t = 0.0625$; dashdot lines and x-marks: $\Delta t = 0.125$; dotted lines and circles: $\Delta t = 0.25$

to be much better than multigrid itself. In Figure 3.9, we demonstrate the performance of Methods 2*and 3* for the solution of the momentum equations for Problem 1, and see clearly that the combination helps a lot indeed, especially for the high Reynolds number case.

Next, we will see what happens on the Convex. The results are presented in Table 3.5. The k_v's, k_p's, ρ_v's and ρ_p's are the same as in Tables 3.1 to 3.4, of course, and therefore are not given again. Only the cases with $Re = 1$ and $\Delta t = 0.0625$ are considered; the other cases do not add new insights. Again, we plot the CPU times per grid point against grid size in Figure 3.10. In order to bring out better the effect of vector length on this vector computer, computations on a 256×256 grid are included for Problem 4. Figure 3.10 show the same behaviour as on the HP: the efficiency of Method 1 deteriorates and that of Methods 2 and 3 improves with grid refinement, but cross-over points when Method 2 and Method 3 surpass Method 1 move to finer grids. On the 32×32 grids, on the Convex Methods 2 and 3 are

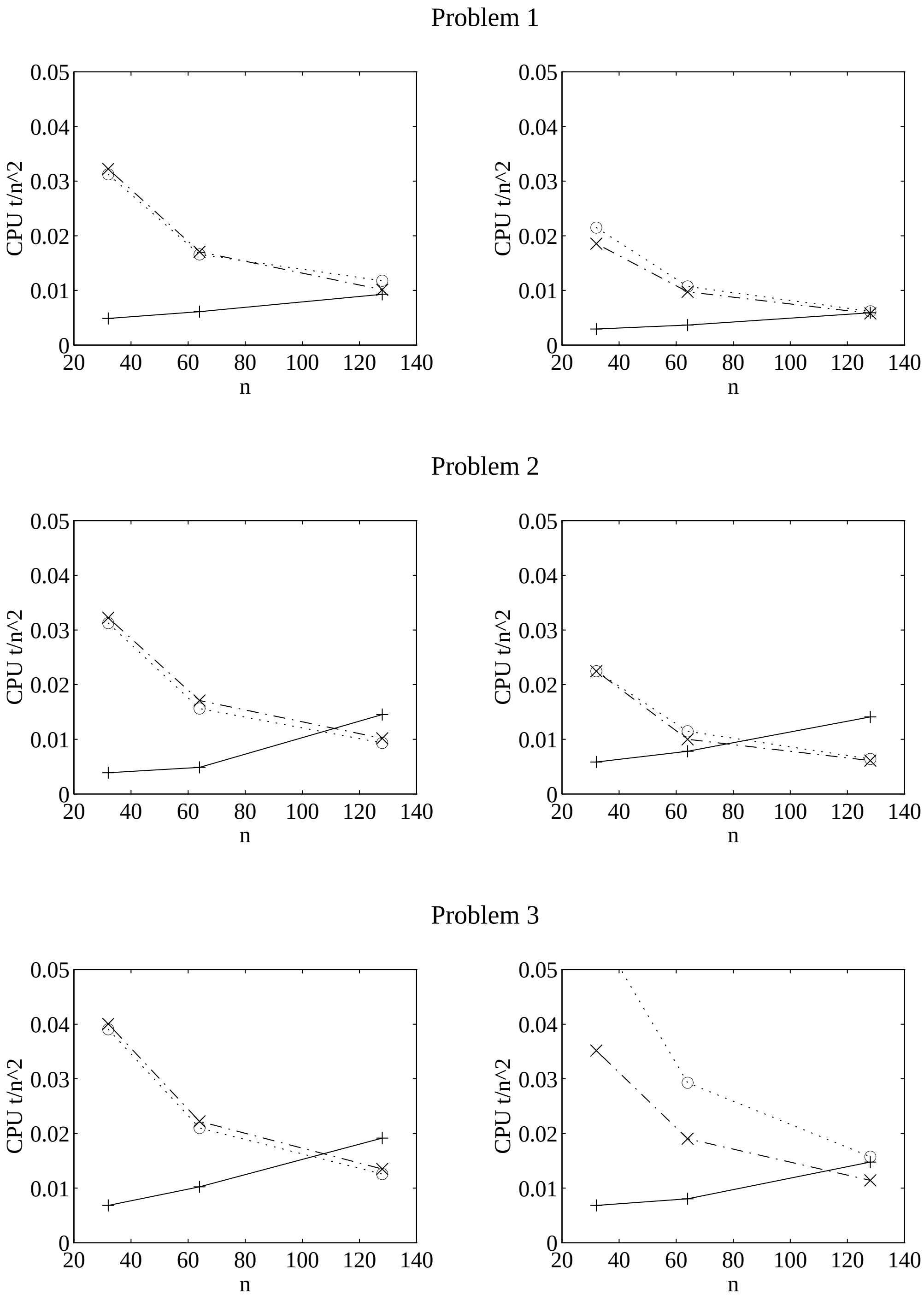

Figure 3.10: (Continued on the next page)

(Continued)

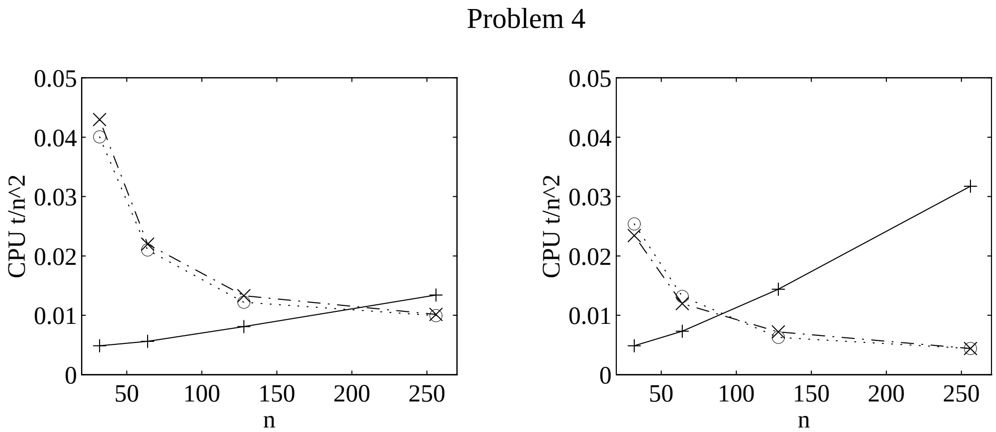

Figure 3.10: CPU time per grid point against grid size for the solution of the momentum equations (left column) and for the solution of the pressure equation (right column) by the multigrid using an ILU smoother on the Convex. Plus marks and solid lines: Method 1; x-marks and dashdot lines: Method 2; circles and dotted lines: Method 3, $Re = 1$, $\Delta t = 0.0625$

slower than on the HP, while Method 1 keeps the same speed. The results for Problem 4 show, that on fine grids, the solution of the pressure equation dominates computing time for Method 1. For Problems 1, 2 and 4, Methods 2 and 3 are equally efficient, although for some cases Method 3 does a little better job; but for Problem 3, Method 2 is obviously better.

Now we investigate further the efficiency of Method 2 and Method 3, and Method 2* and Method 3* as well, by making some comparisons. The four methods have been used on the HP machine. Results show that Methods 2 and 3 are more efficient than Methods 2* and 3* on the HP, in addition to being obviously more robust. Because the smoother in Methods 2* and 3* uses a simple alternating Jacobi line smoothing, which is thought to have greater vectorization potential than the ILU smoothing used in Methods 2 and 3, whether Method 2 and Method 3 still possesses higher efficiency on the Convex remains a question. Before making any comparisons, we note that the efficiency of an algorithm depends not only on the portion of code that can be vectorized in theory, but also on many practical factors such as how vectorization is realized and how memory is accessed. Here we avoid discussions of these matters, and just present numerical results in Table 3.6 for Problem 1, obtained by running the same code on the Convex. In this table, the speed-up factors, defined here as the ratio of the CPU time on the HP to the CPU time on the Convex, are also given. For example, $s_t = t_t(\mathrm{HP})/t_t(\mathrm{Convex})$. Comparing the results here with the data for Methods 2 and 3 in Table 3.1, we may conclude that Methods 2* and 3* are less efficient than Methods 2 and 3 on the 128×128 grid, although on coarser grids they seem to be the same. But it is hard to say from the results that Methods 2* and 3* will be less efficient with further grid refinement, since, the speed-up is larger than for Methods 2 and 3, which implies that

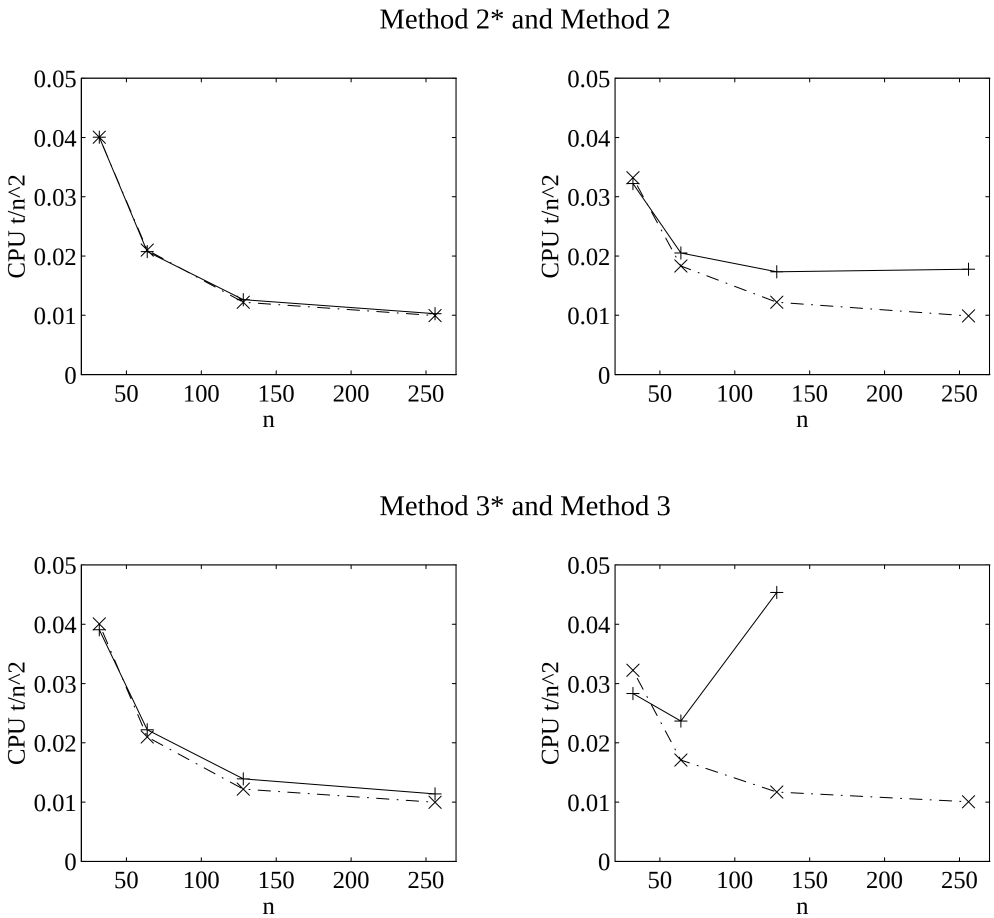

Figure 3.11: Efficiency comparison of Methods 2* and 3* with Methods 2 and 3 on the Convex for the solution of the momentum equation for Problem 4. Left column: $Re = 1$; right column: $Re = 1000$. Solid lines and plus marks: Method 2* or Method 3*; dashdot lines and x-marks: Method 2 or Method 3. $\Delta t = 0.0625$

Methods 2* and 3* indeed have better vectorization properties. The test problem used is simple perhaps, because the numbers of iterations for these methods do not differ very much. Going to more difficult problems, say Problem 3, Methods 2* and 3*, in spite of their larger speed-up factors, will be less efficient since they require a larger number of iterations than Methods 2 and 3 (cf. [7]). In this case, the speed-up can hardly balance the loss of efficiency resulting from the increase of number of iterations. This is verified by carrying out a further test, for example for Problem 4. The results are plotted in Figure 3.11, which shows clearly that Methods 2* and 3* are not able to beat Methods 2 and 3. Although for the low Reynolds number Methods 2* and 2, Methods 3* and 3 have similar performance, for the high Reynolds number Methods 2 and 3 are superior to Methods 2* and 3*. Method 3* does not work well on finer grids and even fails on the 256×256 grid. The curves become flat when going to finer grids, which indicates that the efficiency gain from vectorization is exhausted. In this case, further improvement of efficiency must rely on improvement of the mathematical efficiency (for example, smaller reduction factor) of the algorithm. Looking carefully at Method 2* going from the grid 128×128 to the grid 256×256 for the case of $Re = 1000$, we see that the efficiency on the finer grid is slightly lower than on the coarser one. The reason is that the reduction factor grows with refinement of the grid.

4 Conclusions

We have investigated numerically three iterative methods, namely, Method 1: GMRESR: GCR with GMRES as inner loop, Method 2: GCR with multigrid as inner loop and Method 3: a multigrid method, in the context of application to the solution of the incompressible Navier-Stokes equations in general coordinates. The smoother in the multigrid algorithm for Method 2 and Method 3 is an ILU smoothing.

Numerical experiments are carried out on a sequential computer and on a vector machine. Numerical results demonstrate that the three methods have different numerical behaviour. In particular, Method 1 is different from Method 2 and Method 3; Method 2 and Method 3 show similar performance. On the sequential computer, some facts observed in [7] also apply here. On coarser grids, Method 1 is much more efficient than Method 2 and Method 3. When going to finer grids, Method 1 becomes less efficient due to a considerable increase of the number of iterations. For Method 1, it seems that the pressure equation is harder to solve than the momentum equations, but for Methods 2 and 3, none of the equations causes special difficulties and both methods show a typical convergence property of multigrid methods, i.e., convergence being (almost) independent of grid size. When moving to the vector computer, Method 1 always benefits from vectorization, while Method 3 and therefore also Method 2 loose efficiency if the grid is not sufficiently fine (here the 32×32 grids). But the tendency is that Methods 2 and 3 are still more efficient than Method 1 if the grid is fine enough, as in the case on the 128×128 grid.

It is found that when the smoother is strong enough, combination of GCR type methods with multigrid does not pay off a lot and actually sometimes gives slightly poorer performance than multigrid itself, but it does help in general. When the smoother is rather weak, combination of both methods improves both robustness and efficiency of the methods. We think that combination of GCR with multigrid would lead to better algorithms than either one of them alone.

Although methods 2 and 3 vectorize less well than methods 2* and 3*, we still expect them to be faster on most vector computers, because of their significantly faster rate of convergence.

The results suggest that the following solution strategies are advantageous for the future implementation of algorithms:

1. To switch to Method 2 or Method 3 from Method 1 if the grid is fine enough. This depends on the specific computer being used and the problem.

2. A more dynamic way is to first carry out a few GMRES type iterations, for example, a few iterations of Method 1. If the problem is found to be difficult to solve by,

say, checking convergence rate, then the solution procedure switches to Method 2 or Method 3.

3. A further combination, in addition to combination of GCR with multigrid as in Method 2, may be to use GMRES type methods like Method 1 in multigrid on coarser grids such that the coarsest grid is not too coarse, making use of the property that GMRES type methods may be more efficient on coarser grids than multigrid algorithms.

Table 3.1: Problem 1 on the HP: the total CPU time t_t, the CPU times t_v and t_p, the numbers of iterations k_v and k_p at the final time step, and the reduction factors ρ_v and ρ_p of the multigrid algorithm in the last iteration at the final time step, the underrelaxation factor $\omega = 0.8$

		$Re = 1$				$Re = 1000$			
Grid	Δt	t_t	t_v, t_p	k_v, k_p	ρ_v, ρ_p	t_t	t_v, t_p	k_v, k_p	ρ_v, ρ_p
Method 1									
32	.0625	12	4, 3	3, 4		9	1, 3	1, 4	
×	.125	13	5, 3	4, 4		10	2, 3	2, 4	
32	.25	13	6, 3	4, 4		10	2, 3	2, 4	
64	.0625	88	44, 26	5, 6		58	13, 27	2, 6	
×	.125	94	49, 26	5, 6		60	15, 26	2, 6	
64	.25	102	56, 27	6, 6		64	19, 26	2, 6	
128	.0625	697	366,252	7, 9		405	78,247	2, 10	
×	.125	787	446,261	9, 9		438	106,251	3, 10	
128	.25	870	521,268	10,11		481	154,247	4, 10	
Method 2									
32	.0625	46	22, 11	3, 4	.0804,.0513	45	21, 11	3, 4	.0684,.0424
×	.125	46	22, 11	3, 4	.0814,.0512	46	22, 11	3, 4	.116 ,.0454
32	.25	46	22, 11	3, 4	.0902,.0491	52	29, 11	4, 4	.130 ,.0486
64	.0625	161	81, 43	3, 4	.0851,.0509	160	80, 43	3, 4	.0907,.0418
×	.125	160	80, 43	3, 4	.0955,.0530	181	100, 43	3, 4	.0938,.0440
64	.25	161	80, 43	3, 4	.115 ,.0534	185	105, 43	4, 4	.105 ,.0463
128	.0625	705	361,193	3, 4	.109 ,.0547	703	361,193	3, 4	.0812,.0445
×	.125	703	361,192	3, 4	.118 ,.0563	813	470,193	4, 4	.106 ,.0452
128	.25	703	360,193	3, 4	.129 ,.0565	814	472,193	4, 4	.115 ,.0463
Method 3									
32	.0625	46	21, 12	3, 5	.0877,.0687	43	20, 11	3, 4	.0829,.0504
×	.125	46	21, 12	3, 5	.0871,.0700	46	22, 11	3, 4	.120 ,.0553
32	.25	47	22, 12	3, 5	.0927,.0709	51	26, 12	4, 5	.156 ,.0568
64	.0625	163	79, 46	3, 5	.0913,.0710	157	76, 43	3, 5	.106 ,.242
×	.125	167	83, 46	3, 5	.0997,.0714	172	89, 45	3, 5	.122 ,.244
64	.25	175	92, 46	3, 5	.113 ,.0719	180	96, 46	4, 5	.173 ,.0579
128	.0625	749	399,200	4, 5	.124 ,.0719	700	358,192	3, 4	.111 ,.0574
×	.125	748	401,197	4, 5	.135 ,.0721	737	394,193	4, 4	.140 ,.0559
128	.25	746	405,191	4, 5	.148 ,.0725	750	397,203	4, 5	.143 ,.0615

Table 3.2: Problem 2 on the HP: the total CPU time t_t, the CPU times t_v and t_p, the numbers of iterations k_v and k_p at the final time step, and the reduction factors ρ_v and ρ_p of the multigrid algorithm in the last iteration at the final time step, the underrelaxation factor $\omega = 0.8$

		$Re = 1$				$Re = 1000$			
Grid	Δt	t_t	t_v, t_p	k_v, k_p	ρ_v, ρ_p	t_t	t_v, t_p	k_v, k_p	ρ_v, ρ_p
Method 1									
32	.0625	15	4, 6	3, 8		31	20, 6	12, 7	
×	.125	15	4, 6	3, 7		d(9)			
32	.25	15	4, 6	3, 7		d(3)			
64	.0625	111	35, 56	4, 12		166	90, 56	14, 12	
×	.125	116	39, 57	5, 12		d(5)			
64	.25	121	44, 57	5, 12		d(2)			
128	.0625	1268	574,608	13, 21		848	204,558	5, 18	
×	.125	2039	1347,606	32, 21		945	305,553	7, 18	
128	.25	d(1)				1126	468,572	11, 21	
Method 2									
32	.0625	48	22, 13	3, 4	.0838,.0577	46	22, 11	3, 4	.0774,.0470
×	.125	48	22, 13	3, 5	.0846,.0477	46	22, 11	3, 4	.0891,.0560
32	.25	47	22, 13	3, 5	.0968,.0474	46	22, 11	3, 4	.0978,.0492
64	.0625	162	80, 44	3, 4	.0679,.0608	161	80, 43	3, 4	.0849,.0699
×	.125	162	80, 44	3, 4	.0681,.0647	161	80, 43	3, 4	.0820,.0644
64	.25	163	81, 44	3, 4	.0685,.0631	161	80, 43	3, 4	.0839,.0625
128	.0625	713	360,202	3, 4	.0633,.0934	703	360,193	3, 4	.0750,.0832
×	.125	720	360,210	3, 4	.0630,.0924	701	359,192	3, 4	.0719,.0858
128	.25	733	360,223	3, 4	.0627,.0818	703	360,192	3, 4	.0719,.0808
Method 3									
32	.0625	46	20, 13	3, 5	.0935,.0725	46	21, 12	3, 5	.103 ,.0606
×	.125	46	20, 13	3, 5	.0925,.0766	46	21, 13	3, 5	.104 ,.0742
32	.25	46	20, 13	3, 5	.0942,.0737	46	21, 13	3, 5	.104 ,.0581
64	.0625	155	70, 47	3, 5	.0842,.0774	159	75, 46	3, 5	.0913,.0691
×	.125	156	70, 48	3, 5	.0838,.0852	159	75, 46	3, 5	.0885,.0619
64	.25	156	70, 48	3, 5	.0834,.0863	159	75, 46	3, 5	.0876,.0722
128	.0625	690	328,212	3, 5	.0916,.107	679	329,200	3, 5	.0854,.0694
×	.125	688	325,213	3, 5	.0915,.111	680	329,201	3, 5	.0867,.0800
128	.25	682	320,211	3, 5	.0913,.118	683	330,202	3, 5	.0886,.103

Table 3.3: Problem 3 on the HP: the total CPU time t_t, the CPU times t_v and t_p, the numbers of iterations k_v and k_p at the final time step, and the reduction factors ρ_v and ρ_p of the multigrid algorithm in the last iteration at the final time step, the underrelaxation factor $\omega = 0.8$

		$Re = 1$				$Re = 1000$			
Grid	Δt	t_t	t_v, t_p	k_v, k_p	ρ_v, ρ_p	t_t	t_v, t_p	k_v, k_p	ρ_v, ρ_p
Method 1									
32	.0625	19	7, 7	5, 9		13	1, 7	1, 8	
×	.125	19	7, 7	5, 8		14	2, 7	2, 8	
32	.25	19	8, 7	5, 8		15	3, 7	3, 8	
64	.0625	151	74, 57	8,13		90	14, 56	2, 12	
×	.125	158	81, 57	9,12		93	18, 55	3, 11	
64	.25	162	86, 57	10,13		104	28, 56	4, 12	
128	.0625	1501	774,642	14,22		830	97,648	2, 22	
×	.125	1617	879,653	18,22		870	132,652	4, 22	
128	.25	1655	917,653	20,23		951	213,653	6, 23	
Method 2									
32	.0625	64	29, 22	4, 8	.203,.346	54	22, 19	3, 7	.0792,.214
×	.125	64	29, 22	4, 8	.209,.350	58	26, 19	4, 7	.117 ,.233
32	.25	64	29, 22	4, 8	.213,.357	63	80, 19	5, 7	.201 ,.231
64	.0625	228	106, 84	4, 8	.216,.233	192	80, 74	3, 7	.0932,.247
×	.125	228	106, 85	4, 8	.217,.260	216	104, 74	4, 7	.117 ,.232
64	.25	227	106, 84	4, 8	.210,.234	217	106, 74	4, 7	.193 ,.244
128	.0625	1003	474,378	4, 8	.215,.234	947	465,332	4, 7	.142 ,.219
×	.125	1001	473,378	4, 8	.202,.231	957	471,335	4, 7	.156 ,.252
128	.25	1004	474,379	4, 8	.185,.223	958	471,336	4, 7	.169 ,.229
Method 3									
32	.0625	74	26, 35	4,14	.246,.371	63	20, 30	3, 12	.0871,.366
×	.125	74	27, 35	4,14	.240,.373	66	24, 30	4, 11	.142 ,.313
32	.25	74	27, 34	4,14	.226,.372	72	29, 30	6, 11	.250 ,.346
64	.0625	257	97,122	4,14	.229,.370	224	76,110	3, 12	.0933,.351
×	.125	255	98,120	4,14	.215,.371	240	91,111	4, 11	.138 ,.366
64	.25	252	97,117	4,13	.200,.370	240	93,109	4, 11	.214 ,.357
128	.0625	1073	424,499	4,13	.203,.370	1015	395,470	4, 10	.163 ,.354
×	.125	1058	425,484	4,13	.194,.370	1056	410,496	4, 11	.179 ,.338
128	.25	1045	425,470	4,12	.190,.368	1099	417,532	4, 12	.191 ,.358

Table 3.4: Problem 4 on the HP: the total CPU time t_t, the CPU times t_v and t_p, the numbers of iterations k_v and k_p at the final time step, and the reduction factors ρ_v and ρ_p of the multigrid algorithm in the last iteration at the final time step, the underrelaxation factor $\omega = 0.8$

		$Re = 1$				$Re = 1000$			
Grid	Δt	t_t	t_v, t_p	k_v, k_p	ρ_v, ρ_p	t_t	t_v, t_p	k_v, k_p	ρ_v, ρ_p
Method 1									
32	.0625	14	4, 5	3, 6		12	1, 5	1, 6	
×	.125	15	5, 5	3, 6		12	2, 5	2, 6	
32	.25	15	5, 5	3, 7		14	4, 5	4, 6	
64	.0625	105	38, 47	4, 10		81	13, 47	2,11	
×	.125	110	42, 48	4, 11		87	18, 48	3,11	
64	.25	113	45, 48	5, 11		103	35, 48	6,10	
128	.0625	992	305,599	6, 20		756	89,580	2,21	
×	.125	1051	353,611	7, 21		814	135,592	4,21	
128	.25	1094	395,612	8, 22		979	300,591	12,21	
Method 2									
32	.0625	55	29, 14	4, 5	.114,.111	47	22, 12	3, 5	.0563,.105
×	.125	55	29, 14	4, 5	.117,.111	54	28, 13	4, 5	.110 ,.0982
32	.25	55	29, 14	4, 5	.126,.102	58	31, 14	5, 5	.211 ,.0860
64	.0625	195	104, 53	4, 5	.118,.131	176	87, 51	3, 5	.0893,.126
×	.125	195	104, 53	4, 5	.124,.130	195	106, 52	4, 5	.187 ,.120
64	.25	194	103, 53	4, 5	.131,.130	213	124, 52	6, 5	.301 ,.133
128	.0625	861	472,238	4, 5	.124,.111	838	453,234	3, 5	.110 ,.129
×	.125	858	470,238	4, 5	.128,.109	856	471,235	4, 5	.0818,.132
128	.25	859	471,238	4, 5	.136,.125	855	470,234	4, 5	.123 ,.129
Method 3									
32	.0625	54	27, 15	4, 6	.193,.110	47	21, 14	3, 6	.0746,.117
×	.125	54	26, 15	4, 6	.192,.113	53	25, 15	4, 7	.148 ,.435
32	.25	54	26, 15	4, 6	.189,.127	58	31, 14	5, 6	.214 ,.141
64	.0625	186	96, 53	4, 6	.191,.335	172	80, 55	3, 7	.0954,.153
×	.125	186	96, 53	4, 6	.188,.364	191	96, 58	4, 7	.202 ,.178
64	.25	184	94, 52	4, 6	.184,.400	213	118, 58	6, 6	.387 ,.146
128	.0625	773	420,203	4, 5	.188,.0847	806	418,238	3, 7	.117 ,.437
×	.125	774	419,205	4, 5	.190,.0968	890	489,251	5, 7	.198 ,.468
128	.25	775	416,208	4, 5	.194,.0819	881	489,243	5, 6	.202 ,.178

Table 3.5: CPU times measured on the Convex: the total CPU time t_t, the CPU times t_v and t_p, the underrelaxation factor $\omega = 0.8$, $\Delta t = 0.0625$, $Re = 1$

	Problem 1		Problem 2		Problem 3		Problem 4	
Grid	t_t	t_v, t_p	t_t	t_v, t_p	t_t	t_v, t_p	t_t	t_v, t_p
Method 1								
32×32	12	5, 3	14	4, 6	18	7, 7	14	5, 5
64×64	53	25, 15	65	20, 32	89	42, 33	68	23, 30
128×128	299	152, 97	520	238,231	606	314,242	421	133,236
Method 2								
32×32	90	33, 19	93	33, 23	110	41, 36	105	44, 24
64×64	170	70, 40	171	70, 41	225	91, 78	199	90, 49
128×128	376	166, 95	384	167,100	523	221,187	452	218,118
Method 3								
32×32	89	32, 22	92	32, 23	131	40, 59	102	41, 26
64×64	168	68, 44	170	64, 47	262	86,120	200	86, 54
128×128	408	193,101	375	153,105	577	206,258	419	200,103

Table 3.6: CPU times measured on the Convex for Methods 2* and 3* and speed-up factors for Methods 2*, 3*, 2 and 3, for Problem 1 with $\Delta t = 0.0625$

Grid	t_t	t_t, t_p	s_t	s_v, s_p	s_t	s_v, s_p
Method 2*					Method 2	
32×32	83	37, 15	.75	.89, .80	.51	.67, .58
64×64	170	85, 33	1.3	1.5, 1.3	.95	1.2, 1.1
128×128	409	218, 82	2.3	2.6, 2.4	1.9	2.2, 2.0
Method 3*					Method 3	
32×32	89	39, 18	.71	.95, .78	.52	.66, .55
64×64	187	98, 37	1.3	1.7, 1.4	.97	1.2, 1.0
128×128	450	255, 86	2.4	2.8, 2.6	1.8	2.1, 2.0

References

[1] Eisenstat, S.C., H.C. Elman and M.H. Schultz, *Variable iterative methods for non-symmetric systems of linear equations.* SIAM J. Numer. Anal. **20**, 345–357, 1983.

[2] Saad, Y. and M.H. Schultz, *A generalized minimal residual algorithm for solving non-symmetric linear systems.* SIAM J. Sci. Statist. Comput. **7**, 856–869, 1986.

[3] Vorst, H.A. van der and C. Vuik, *GMRESR: A family of nested GMRES methods.* Report 91-80, Faculty of Technical Mathematics and Informatics, TU Delft, The Netherlands, 1991. To appear in J. Numer. L. A. A.

[4] Vuik, C., *Further experiences with GMRESR.* Report 92-12, Faculty of Technical Mathematics and Informatics, TU Delft, The Netherlands, 1992.

[5] Vuik, C., *Solution of the discretized incompressible Navier-Stokes equations with the GMRES method.* Int. J. Num. Methods in Fluids, **16**, 507–523, 1993.

[6] Vuik, C., *The solution of the discretized incompressible Navier-Stokes equations with iterative methods.* Proceedings of 26th International Symposium on Automative Technology and Automation, 69–76. Dedicated conference on supercomputer application in the automative industries, Aachen, Germany, Sept.13–17, 1993. Automative Automation Limited, Croydon, 1993.

[7] Zeng, S., C. Vuik and P. Wesseling: *Solution of the incompressible Navier-Stokes equations in general coordinates by Krylov subspace and multigrid methods.* Report 93-64, Faculty of Technical Mathematics and Informatics, TU Delft, The Netherlands, 1993.