# A Boosted P' Method for Fully Parabolized Flows

Leon van Dommelen[*][†]
Department of Mechanical Engineering, FAMU-FSU College of Engineering, Tallahassee, Florida 32310, USA

January 27, 2011

**Abstract**

A novel iterative method for solving the incompressible, fully parabolized Navier-Stokes equations is described. It is a modification of the classical so-called P' method. The central modification is to boost the computed pressure changes during the iterations. The method is shown to produce robust convergence for the considered application in a simple way.

**NOMENCLATURE**

| | | | |
|---|---|---|---|
| $A, B$ | generic difference operators | $u, v, w$ | $x, y, z$ velocity components |
| $C_i, D_i$ | generic constants | $x$ | streamwise coordinate |
| $n$ | P' iteration number | $y$ | coordinate normal to the surface |
| $p$ | part of the pressure that varies over the $y, z$ cross section | $z$ | spanwise coordinate |
| | **Greek symbols** | | |
| $\alpha, \beta, \gamma$ | computational coordinates | $\delta$ | iterative change |
| $\Delta$ | central (half cell) difference operator | | |

## 1 Introduction

The parabolized Navier-Stokes equations are simplified versions of the Navier-Stokes equations that facilitate computations, [1]. The version of interest here was introduced by [2]. It assumes that the streamwise pressure gradient can be approximated by an average over the flow cross-section. (In the particular example discussed in this paper, this is not an approximation but exact due to the asymptotic nature of the problem.) The simplifications offered by parabolized equations have become less critical with the arrival of much faster computers. However, as pointed out by Hall, [3, 4], the same equations are also the governing equations in various asymptotic problems of theoretical interest, especially Görtler vortices. Furthermore, the steady fully parabolized equations include the two-dimensional unsteady incompressible Navier-Stokes equations as a special case: the streamwise spatial coordinate is equivalent to time if the streamwise velocity is unity. Therefore, the equations remain of interest.

[*]Address correspondence to Leon van Dommelen, Department of Mechanical Engineering, FAMU-FSU College of Engineering, 2525 Pottsdamer Street, room 229, Tallahassee, FL 32310-6046, USA. E-mail: dommelen@eng.fsu.edu

[†]This material is based in part upon work supported by the U.S. Army Research Laboratory and the U.S. Army Research Office under grant number W911NF-05-1-0295.

As one example, the present study resulted from the desire to solve an asymptotic problem in boundary layer separation governed by these equations. Such asymptotic problems tend to be characterized by semi-infinite domains, singular meshes, and flow singularities of various kinds when one asymptotic expansion gets replaced by another. This tends to make numerical solution more complicated, and simple, reliable methods desirable.

The novel iterative method described in this paper was developed for that purpose. It is a modification of the original P' method of Patankar & Spalding [1, 5, 6]. The objective of that method is to advance the solution of the parabolized Navier-Stokes equations from one streamwise position to the next. To do so, the method first uses an approximate pressure in the momentum equations to compute the velocities at the next station. The so-obtained solution will no longer satisfy the continuity equation, however. Therefore a correction is added to the pressure that intends to correct the errors in the continuity equation back to zero. Unfortunately, to find that correction exactly would require a combined solution of the momentum and continuity equations. To reduce that effort, greatly simplified momentum equations are used. In the simplest case, only diagonal terms in the momentum equations are retained. Therefore, when the velocities are recomputed from the momentum equations using the corrected pressure, the continuity equation will still not be satisfied exactly. As a result, iteration is needed.

Later variants of the method have improved convergence by abandoning the direct connection between the pressure and velocity changes. Raithby & Schneider found in a numerical tests that the original P' method requires noticeably more iterations than later methods, [7]. Similarly in a later discussion, Patankar [6] recommends a separate Poisson equation pressure updating instead of the older P' method. The reduction in iterations more than compensates for the effort in solving the additional equation. However, the study of Raithby & Schneider is not directly applicable to the flows of interest in this paper, as it assumes that there is no streamwise velocity. Then the diagonal coefficients in the cross-plane momentum equations are only produced through in-plane upwinding and viscous terms. In the parabolized steady flows of interest here, the streamwise velocity is not zero. That makes the streamwise direction time-like. The solutions are marched in this direction, and streamwise accuracy is required.

The original P' method remains appealing because of its relative simplicity. Mesh sizes that vary over large factors at the same streamwise station can make the selection of suitable relaxation factors difficult. The intuitive interpretation of the P' method also makes it easier to formulate local adaptations to deal with singularities. The basic iterative procedure proposed here uses the P' method with the retained terms in the momentum equations given by the streamwise derivatives. That produces immediate convergence for components of the error in pressure that have finite derivatives with respect to the transverse coordinates. However, it produces slow convergence for error components that vary more quickly in the transverse directions. To speed up the convergence of these components, the pressure changes computed by the P' method are boosted in magnitude.

This paper provides numerical evidence of the effectiveness of this modified P' procedure and an extension of it. To do so, the next section briefly describes the parabolized problem studied and its numerical discretization. The iterative procedures are explained in section 3, and their numerical performance on typical problems is discussed in section 4.

## 2 Numerical problem

In order to put the results for the iterative methods in context, some understanding of the actual type of problem solved and its numerical discretization may be helpful.

The governing parabolized Navier Stokes equations are

$$\begin{aligned} uu_{,x} + vu_{,y} + wu_{,z} &= u_{,yy} + u_{,zz} + u_e u_{e,x} \quad (1) \\ uv_{,x} + vv_{,y} + wv_{,z} &= v_{,yy} + v_{,zz} - p_{,y} + \kappa u^2 \quad (2) \\ uw_{,x} + vw_{,y} + ww_{,z} &= w_{,yy} + w_{,zz} - p_{,z} \quad (3) \\ u_{,x} + v_{,y} + w_{,z} &= 0 \quad (4) \end{aligned}$$

They correspond to the asymptotic boundary layer problem for large Reynolds number in which the spanwise scales are comparable in size to the boundary layer thickness. The given potential flow velocity above the boundary layer is $u_e$. The coordinate in the predominant streamwise direction is $x$, $y$ is the transverse coordinate through the boundary layer, and $z$ is the spanwise coordinate. The $x$-coordinate is time-like in the problem.

At the wall $y = 0$, a transpiration boundary condition of nonzero $v$ was given. The flow was taken to be periodic in $z$ with period $\lambda$. For $y \to \infty$, matching with the overlying potential flow requires

$$u \sim u_e \quad v \sim -u_{e,x} y + v_d \quad w \sim w_e$$
$$p \sim \tfrac{1}{2}\left(u_e u_{e,xx} - u_{e,x}^2\right) y^2 + \left(u_{e,x} v_d - u_e v_{d,x} + \kappa u_e^2\right) y \quad (5)$$

Note that $v$ and $p_{,y}$ become infinite at infinite $y$. That is an effect of the asymptotic nature of the problem. In the numerical scheme, the infinite growth was analytically subtracted from $v$ and $p$, so that the remainders $\bar{v}$ and $\bar{p}_{,y}$ that had to be evaluated numerically were nonsingular. Further, because of numerical problems associated with the initial singularity of asymptotic Blasius-type boundary layers, above a cut-off value of $y$ the $z$-momentum equation was replaced by the potential flow condition of zero streamwise vorticity. (Some earlier computations started from a initial boundary layer of finite thickness and used the parabolized equations above all the way to infinity. Those earlier computations used only schemes I and Ib of section 4; the effects of pressure boost were similar to those reported here.)

The computational domain was mapped to a unit cube using algebraic mesh stretchings of the form $\alpha = \alpha(x)$, $\beta = \beta(x, y)$, $\gamma = \gamma(z)$. Only the $\beta$ mapping is of some interest here:

$$\beta = B\frac{2}{\pi}\arctan\left(y/C_D\sqrt{x}\right) + (1 - B)\frac{2}{\pi}\arctan\left(y/C_\beta\lambda\right)$$

where $B$, $C_\beta$, and $C_D$ are empirically chosen constants. This mapping was designed to spread transverse mesh points over both the typical viscous layer thickness, estimated as $C_D\sqrt{x}$, as well as over the typical transverse inviscid length, which scales with the spanwise period $\lambda$. Initially and far downstream, these scales are of different magnitudes. Further, the arctangents allow the mesh points to be spread out over the complete semi-infinite range in $y$.

The $v$ and $w$ mesh points in cross-flow planes of constant $x$ were staggered according to the MAC scheme of Harlow and Welch, [8]. The streamwise velocity was positioned at the pressure locations. All derivatives were approximated by second order difference formulae. In particular, the streamwise $x$-derivatives were approximated by A-stable three-point backward differences. Derivatives in the cross-flow plane were mostly approximated with three point central differences. The convective $y$-derivatives were evaluated essentially centrally in regions in which the $y$-mesh size varies slowly with the mesh point index. However, an amount of additional viscosity was included based on the ratio of the two mesh spacings involved; that amount was chosen to turn the discretization into a fully upwind first order one in the case that the ratio of successive mesh cell sizes becomes zero or infinity. (This happens at the last mesh point before infinite $y$, and asymptotically also at the viscous/inviscid boundary for $x \to 0$ and $x \to \infty$.) Further, the convective $z$ derivatives were approximated by first order upwind derivatives, corrected to second order by adding the difference between central and upwind differences evaluated in the previous cross plane. The main reason for upwinding was that the substep of the P' method in which the momentum equations are solved was performed using multigrid iteration. Using symmetric Gauss-Seidel relaxation in the $z$-direction, during one of the two sweep directions, the full convective $z$-derivative is included. Multigrid relaxation in $z$ was also used to converge the Poisson equation substep. In the $y$-direction, a tridiagonal or block-tridiagonal algorithm was used.

## 3 Iterative procedure

The iterative procedure can be understood from a grossly simplified model problem. Assume that the streamwise momentum equation has already been solved, and that the $y$ and $z$ momentum equations are linear constant coefficient difference equations of the form:

$$Av + Bv + C_1\Delta_y p = 0 \qquad Aw + Bw + C_2\Delta_z p = 0$$

where $v$ and $w$ indicate the cross-flow velocities in the plane being computed, $A$ stands for the coefficient proportional to $u/\Delta x$, $B$ is a differential operator representing the convective and viscous derivatives, $\Delta_y$ and $\Delta_z$ represent the two-point difference operators approximating the pressure derivatives, and $C_1$ and $C_2$ are algebraic factors. The desired right hand sides are zero assuming that $v$, $w$, and $p$ are defined as the differences from the converged values $v^\infty$, $w^\infty$, and $p^\infty$. Similarly, assume the continuity equation to be

$$D_1\Delta_y v + D_2\Delta_z w = 0$$

Also, for simplicity assume periodic boundary conditions in both $y$ and $z$.

The basic P' method as implemented here can now be described as follows. Given an approximate starting pressure $p^n$ for iteration number $n$, the method first solves the momentum equations

$$Av^n + Bv^n + C_1\Delta_y p^n = 0 \qquad Aw^n + Bw^n + C_2\Delta_z p^n = 0$$

which will produce a residual $r^n$ in the continuity equation:

$$D_1\Delta_y v^n + D_2\Delta_z w^n = r^n$$

The second step of the iteration solves the continuity equation along with approximated momentum equations:

$$A(v^{n+1} - v^n) + C_1\Delta_y(p^{n+1} - p^n) = 0 \quad A(w^{n+1} - w^n) + C_2\Delta_z(p^{n+1} - p^n) = 0$$

$$D_1\Delta_y(v^{n+1} - v^n) + D_2\Delta_z(w^{n+1} - w^n) = -r^n$$

To keep the problem easy to solve, in the momentum equations only the changes in the streamwise derivatives are included; the values of the convective and diffusive terms are kept frozen. As a result, at the end of the iteration, the continuity equation is satisfied, but the momentum equations are no longer.

The approximate system above can be reduced to a Poisson-type equation for the pressure:

$$D_1\Delta_y A^{-1}C_1\Delta_y(p^{n+1} - p^n) + D_2\Delta_z A^{-1}C_2\Delta_y(p^{n+1} - p^n) = r^n$$

This equation can be compared with the one that would produce the converged pressure $p^{\infty}$:

$$D_1 \Delta_y (A+B)^{-1} C_1 \Delta_y (p^{\infty} - p^n) + D_2 \Delta_z (A+B)^{-1} C_2 \Delta_z (p^{\infty} - p^n) = r^n$$

For a constant coefficient problem in which the various operators can be assumed to commute, it follows that

$$(A+B)^{-1}(p^{\infty} - p^n) = A^{-1}(p^{n+1} - p^n)$$

hence

$$p^{\infty} - p^n = p^{n+1} - p^n + BA^{-1}(p^{n+1} - p^n) \tag{6}$$

It is seen that as long as $B$ can be ignored, the pressure $p^{n+1}$ will be the converged one. That is the case for Fourier modes of finite wave lengths in $y$ and $z$, because $A$ is large of order $1/\Delta x$. These Fourier modes will be converged after a single iteration. On the other hand, Fourier modes whose wave lengths scale with the mesh spacing will be greatly underestimated, because for these modes $B$ will be large and positive.

It should here be noted that the above procedure follows the version of the P' method described in [1, p. 589]. Typical other implementations, including those in [5, 6], use the full diagonal difference coefficient for $A$, not just the part due to the streamwise convective term. However, this can produce a severe overestimate for the Fourier modes of finite wave lengths, and severe under-relaxation must then be used as in [7]. Having to empirically set such under-relaxation parameters was felt to be impractical for the computations of interest, because of the infinite variation in mesh size and the complicating effect of the additional potential flow scheme above the cut-off. In the computations using the scheme as described here, under-relaxation was never needed nor used. The problem was not divergence, (except for the instability of Figure 3), but extremely slow convergence.

In fact, the proposed pressure boost can be thought of as an over-relaxation scheme. Indeed, note that Eq. (6) is trivial to solve for $p^{\infty}$ from the pressure difference computed in the P' iteration. That then is the pressure boost idea: after a normal iteration has been performed, compute a boosted pressure difference $p^{\infty} - p^n$ from Eq. (6). That will give the converged pressure, and the next solution of the momentum equations then produces the complete converged solution.

Of course, this only works for the above greatly simplified model problem. The true problem is far from constant coefficient, and it has nontrivial boundary conditions. However, these problems may be less of an issue for rapidly varying Fourier modes than for slowly varying ones. Therefore, after slowly varying modes have been converged, there might still be an advantage to boosting the pressure differences according to Eq. (6). Numerical experimentation confirmed this. Applying pressure boost every so many P' iterations could provide dramatic increases in convergence for slowly converging cases.

However, if boost was applied too frequently, it could suddenly cause the iterations to diverge. Since no way to predict this reliably was evident, the boost was implemented using undo: if the boost increased the residuals in the momentum equations, the unboosted solution was restored by solving the momentum equation once more, but now using the original unboosted pressure. This required some storage to save the unboosted pressure, and adds a little time to resolve the momentum equations. However, these disadvantages were found to be minor for the considered application.

In the final implementation, boost was only started after a certain number of unboosted iterations, because initially it is ineffective. This can be understood from the arguments above; as long as the slowly varying modes are not sufficiently converged, the assumptions that the effects of boundary conditions can be ignored and that operators commute are simply not reasonable. Also, if the number of unboosted iterations between boosted ones was too small, boost was found to become less efficient. Therefore, a minimum number of unboosted iterations between boosted ones was enforced, typically one or two.

Additionally, the starting point of the boosted iterations and the frequency of boosted iterations was allowed to slowly evolve depending on whether the previous boosted iteration was successful in increasing the convergence. This implementation was found to be very reliable, as the examples in the next section illustrate, and converged quite well for the more simple flows studied initially. It is not sensitive to the precise parameters used.

However, for more complicated flows, especially those with a nontrivial singularity at the start of the computation, convergence remained slow. At the start of such computations the $y$ derivatives are much larger than the streamwise ones. The self-evident way to correct for this was to include the $y$ derivatives in the operator $A$ above, giving what will be indicated as scheme II. In this scheme only the $z$-derivatives in the momentum equations are frozen in the solution of the continuity equation. For the flows in question scheme II did indeed increase convergence greatly. In fact, there was no significant slow-down in convergence observed and boost did no longer offer a benefit, as the examples in the next section illustrate. While scheme II required solutions of block, rather than scalar tridiagonal systems, the decrease in iterations more than makes up for it, as demonstrated by the examples.

However, subsequently it became necessary to compute much thicker boundary layers, for which the coefficients of the $z$-differences are relatively much larger. For these flows, even for scheme II convergence became slow. Unfortunately, boost as described above, (but using the modified $A$ and $B$), no longer removed the difficulty.

Maybe it would be possible to alternately take the operator $A$ to consist of the $y$ and $z$ differences. However, that would require multigrid interpolations in the severely, (and asymptotically singularly), stretched $y$-direction, and that was considered too complex.

When a typical example case of boost using the $y$-differences for $A$ was examined, the first pressure boost iteration was found to produce a significant decrease in error, but subsequent pressure boosts were ineffective. Underrelaxing the boost did not increase the convergence, it simply took more boosted iterations to achieve the same reduction in error compared to no boost. When the true $p^\infty - p^n$ was compared with the one computed from Eq. (6), it was found that while the rapid $z$-variation of the computed correction quite nicely matched the correct one, its amplitude was way off and varied greatly, but slowly with $y$.

That suggested the following scheme IIb:

- After solution of the continuity equation and approximate momentum equations, compute the velocities from the full momentum equations using the unboosted pressure $p_{jk}^{n+1}$, where $j$ is the transverse, $y$, mesh index and $k$ spanwise, $z$, one. Evaluate the corresponding residuals in the continuity equation $\varepsilon_{jk}^{0}$.

- Recompute the velocities from the momentum equations using the boosted pressure $p_{jk}^{b}$ found from Eq. (6). Evaluate the residuals $\varepsilon_{jk}^{b}$ in the continuity equation. If the root-mean-square residual has become smaller due to boost, accept the boosted solution as is and go do the next iteration.

- If not, the final pressure field $p_{jk}^{f}$ is to be chosen. The simplest choice would be

$$p_{jk}^{f} = p_{jk}^{n+1} + \lambda(p_{jk}^{b} - p_{jk}^{n+1})$$

with $\lambda$ some under-relaxation constant. Then, assuming that the residuals $\varepsilon^f_{jk}$ vary linearly with $\lambda$, the smallest least square errors would be obtained by taking

$$\lambda = -\sum_{jk} \varepsilon^0_{jk}(\varepsilon^b_{jk} - \varepsilon^0_{jk}) \Big/ \sum_{jk} (\varepsilon^b_{jk} - \varepsilon^0_{jk})^2$$

However, the example showed that the desired amount of boosted pressure to add varies strongly with $j$. For that reason, a local value of $\lambda$ depending on $j$ was defined as

$$T_j \equiv \sum_j \varepsilon^0_{jk}(\varepsilon^b_{jk} - \varepsilon^0_{jk}) \quad B_j \equiv \sum_j (\varepsilon^b_{jk} - \varepsilon^0_{jk})^2$$

$$\lambda_j = -\frac{B_{j-1} + 2B_j + B_{j+1}}{T_{j-1} + 2T_j + T_{j+1}}$$

The only justification for this procedure is that it works experimentally. The averaging of the sums with neighboring $j$ values was done to reduce introduction of short scale fluctuations between mesh lines. The $\lambda_j$ as computed above are next restricted to the interval from 0 to 1 to prevent changes in pressure that exceed what can be justified based on the estimated boosted pressure.

Recompute the velocities from the momentum equations using the obtained final pressure $p^f_{jk}$. Evaluate the residuals $\varepsilon^f_{jk}$ in the continuity equation. If the root-mean-square residual has not become smaller, restore the solution from the unboosted pressure.

Like for the scheme Ib, the undo allows the computation to run reliably unattended.

To find the boosted pressure requires boundary conditions at the wall and at the potential flow cut-off. At the wall a homogeneous Neumann condition was enforced. At the potential cut-off, $A^{-1}(p^{n+1} - p^n)$ was evaluated by ignoring the derivative operators in $A$. In the potential flow region, the pressure was not boosted.

Two additional small modifications to the P' method should be mentioned. Near separation, the convective motion away from the wall blows up, and very small streamwise step sizes are desirable to maintain accuracy. From the above it would seem that very small step sizes would increase the diagonal part of $A$, hence promote iterative convergence. However, it was found that for step sizes that were too small, the iterations could in fact fail to converge. Continuity equation iterations change the velocity components $v$ and $w$, and they in turn change the streamwise velocity $u$ because of their presence in the streamwise momentum equation. (Note that other parabolized computations may freeze the streamwise velocity component, [e.g. 1, 5]. That was not considered desirable in the present studies, as it makes the streamwise velocity depend on the initial guess for the pressure. Additionally, the streamwise velocity gradients are large near separation, producing important transverse velocities.) For small streamwise steps, the small changes in $u$ remain important because $u$ has a coefficient $1/\Delta x$ in the continuity equation. The solution to the convergence problem was found to be to add variations in streamwise velocity to the continuity iterations. That requires that an approximate streamwise momentum equation is added too. For that purpose the linearized $x$-momentum equation was used with all nondiagonal terms (i.e. the terms involving changes in neighboring $u$-values) ignored. This maintains the compact size of the stencil solved in the continuity iterations. It will however under-correct for the effect of $u$ on continuity unless the streamwise step is very small.

Second, if the boundary layer thickness is assumed to start from zero, as in the cases discussed here, the P' method diverges for the initial streamwise positions. The problem is that for a sufficiently thin boundary layer, the problem acquires a conventional boundary layer character. In a conventional boundary layer, the transverse momentum equation becomes an equation for the pressure across the boundary layer, not for the transverse velocity. Attempting to correct the residual in this momentum equation by means of transverse velocity changes requires excessive velocity changes, and the iterations diverge. The solution was to allow a limited amount of change to the pressure to occur during the solution of the momentum equations. In the relaxation of each line of constant $z$, the residual $\varepsilon_y$ in the $y$-momentum equation was set to zero by velocity and pressure changes $\delta p$ and $\delta v$ satisfying

$$\bar{A}\delta v + C_1 \Delta_y \delta p = -\varepsilon_y \tag{7}$$

(The operator $\bar{A}$ is not quite the same as $A$ earlier; $\bar{A}$ also includes the diagonal part of the spanwise derivatives.) If the pressure is no longer kept frozen, an equation is needed to determine how it varies. This equation was taken to be

$$f C_1 \Delta_y \delta v + \bar{A}\delta p = 0 \tag{8}$$

since this provides a well conditioned system when combined with Eq. (7). (Note that $\Delta_y$ becomes an imaginary factor in a Fourier analysis.) If the factor $f$ is zero, the standard P' solution of the continuity equations is obtained. On the other hand, if $f$ is infinite, the change in $v$ is computed from the continuity equation, ignoring changes in $u$ and $w$, and the one in $p$ from the momentum equation. That is how it needs to be done in a conventional boundary layer. In the computations, the factor $f$ was taken to be 1, except where noted. It is reasonable to expect that except at small streamwise positions, the difference from the standard P' method will be inconsequential, since the operator $\bar{A}$ is normally much larger than $C_1\Delta_y$. However, a few results deviate from that expectation.

## 4 Numerical results

This section gives a number of examples of the effect of pressure boost on convergence.

| mesh | method | $\overline{n}$ | $n_{\text{max}}$ | $n_{\text{total}}$ | time |
|---|---|---|---|---|---|
| $96 \times 36 \times 36$ | I | 690 | 2,824 | 55,192 | 19,796 |
| $96 \times 36 \times 36$ | Ib | 157 | 547 | 12,573 | 7,939 |
| $96 \times 36 \times 36$ | II | 20 | 39 | 1,577 | 1,549 |
| $96 \times 36 \times 36$ | IIb | 20 | 39 | 1,577 | 1,384 |
| $192 \times 72 \times 72$ | I | 1899 | 11,647 | 305,683 | 337,283 |
| $192 \times 72 \times 72$ | Ib | 279 | 2,370 | 44,972 | 119,362 |
| $192 \times 72 \times 72$ | II | 20 | 36 | 3,188 | 10,981 |
| $192 \times 72 \times 72$ | IIb | 20 | 36 | 3,188 | 11,019 |
| $384 \times 144 \times 144$ | I | >2,509 | >9,999 | >700,044 | >2,482,260 |
| $384 \times 144 \times 144$ | Ib | 950 | 9,727 | 305,094 | ~3,151,227 |
| $384 \times 144 \times 144$ | II | 20 | 33 | 6,295 | 82,886 |
| $384 \times 144 \times 144$ | IIb | 20 | 33 | 6,297 | 75,722 |

Table 1: Comparison of iterative procedures for Howarth flow with strong three-dimensional short-scale blowing.

The first example is Howarth flow, [9], with relatively strong three-dimensional blowing into it. Table 1 shows average and maximum number of iterations per spatial step, the total number of iterations, and the computational time as reported by the g77 "second()" function. These time values are for context only; the reported times were observed to vary randomly by roughly 10% on the SUN work stations on which the program ran. All iterations were continued until the maximum error in the governing momentum, continuity, and irrotationality equations, in stretched coordinates but not multiplied by a mesh size, was less than $10^{-8}$.

The finest computation using boosted scheme Ib had to be restarted twice, the second time due to a system down. That required 43% of the total run time to be estimated from the wall clock time. But there was very little difference between those times. The finest computation using unboosted scheme I terminated prematurely after 278 steps out of 320 for exceeding 9,999 iterations. Since it was expected that its total time would far exceed that of the Ib scheme, (which did 9 more steps in only 161,042 iterations and 1,161,160 seconds) it was not considered realistic to restart it.

As Table 1 shows, including the full transverse derivatives of the momentum equations in the solution of the continuity equation, as done in schemes II and IIb, has significant benefits for both iterations and computational time. In addition, for these schemes the number of iterations per step and the computational time per computed point are quite independent of mesh size. Both schemes are "fast" in the sense that the computational time is approximately proportional to the number of computed points; it goes up by a factor eight in each mesh halving. For schemes I and Ib both quantities deteriorate with mesh size.

The boost of scheme IIb offers no benefit compared to scheme II. (The reduced time is random variation from running on a less busy workstation, as seen from other results not reported here.) However, the boost of the simpler scheme Ib produces a dramatic decrease in iterations compared to the unboosted scheme I.

| mesh | method | $\overline{n}$ | $n_{\max}$ | $n_{\text{total}}$ | time |
|---|---|---|---|---|---|
| $96 \times 36 \times 36$ | I | 692 | 2,834 | 55,365 | 14,331 |
| $96 \times 36 \times 36$ | Ib | 157 | 546 | 12,572 | 4,551 |
| $96 \times 36 \times 36$ | II | 20 | 40 | 1,582 | 661 |
| $96 \times 36 \times 36$ | IIb | 20 | 40 | 1,582 | 674 |

Table 2: Results after reducing the inner iterations.

The decrease in computational time is significant, but less dramatic. One reason is that in each major iteration the multigrid subiterations of the momentum equations and continuity equations were converged out fully. That effort is largest during the initial iterations of each step because of the less accurate initial guess. It is also largely wasted, because the obtained results are then still not close to the correct ones anyway. This wasted effort will disproportionally affect the faster converging schemes, since they do less of the fast converging later iterations. To get an idea of how big the effect is, in Table 2 the coarsest mesh computations were repeated when the continuity and momentum equations subiterations were each time converged by no more than a factor ten from the initial guess. Note from the table that the number of iterations is hardly affected, and the time of the slowly converging scheme I is not much either. However, the times of the boosted scheme Ib, and especially the fast converging schemes II and IIb are significantly decreased.

Still, optimizing the number of subiterations also adds an additional confusing parameter to the analysis that is really not relevant to the question how effective pressure boost is. Therefore, the other reported data are based on fully converged subiterations at all stages. It makes the number of iterations an objective quantity, but the computational times may still be subject to significant improvement under fine-tuning of the overall algorithm.

If the factor $f$ in the P' modification Eq. (8) is set to zero, giving the original P' method, the computations diverge at the first streamwise station. If instead the factor is changed from 1 to 0.1, the number of iterations increases only slightly for schemes II and IIb. In particular, it takes a few more iterations for the first few steps, but the later streamwise stations are not affected. Strangely enough, for schemes I and Ib, the number of iterations increases by roughly a factor 3. For these schemes, the modification Eq. (8) has a significant beneficial effect not just at early stations, but also at later stations. The $384 \times 144 \times 144$ mesh computations were not attempted using $f = 0.1$.

| mesh | method | $n_1$ | $n_2$ | time |
|---|---|---|---|---|
| $645 \times 36 \times 36$ | I | 14,208 | 4,664 | 11,754 |
| $645 \times 36 \times 36$ | Ib | 919 | 398 | 1,474 |
| $645 \times 36 \times 36$ | II | 169 | 103 | 235 |
| $645 \times 36 \times 36$ | IIb | 36 | 27 | 103 |
| $1290 \times 72 \times 72$ | I | 50,748 | 9,373 | 150,255 |
| $1290 \times 72 \times 72$ | Ib | 2,175 | 759 | 14,001 |
| $1290 \times 72 \times 72$ | II | 288 | 98 | 1,105 |
| $1290 \times 72 \times 72$ | IIb | 38 | 25 | 422 |
| $2580 \times 144 \times 144$ | I | >99,999 | >0 | >2,064,463 |
| $2580 \times 144 \times 144$ | II | 4,939 | 1,941 | 172,390 |
| $2580 \times 144 \times 144$ | II | 529 | 162 | 11,300 |
| $2580 \times 144 \times 144$ | IIb | 38 | 24 | 1,983 |

Table 3: Comparison of iterative procedures for Blasius flow with a relatively thick blown boundary layer.

Table 3 shows convergence for a case in which the boundary layer is much thicker than the spanwise period. The number of iterations for two streamwise steps are shown; in the first step the solution is initialized to be equal to the one in the previous step. The second step was initialized using a third order accurate extrapolation from the three previous steps. There is again a significant benefit of the boosted scheme Ib over scheme I, and of the improved approximate momentum equations in schemes II and IIb.

In this case, the convergence of scheme II has deteriorated significantly. That can be understood from the fact that the spanwise terms not included in the approximate momentum equations are now much more significant compared to the included transverse ones. However, convergence for the boosted scheme IIb remains very good, and does not deteriorate much with mesh size in the range used in these studies. The computational time remains roughly "fast," going up by about a factor four at each mesh doubling, though the finest mesh seems to have slowed down a bit. Therefore boost now offers a significant benefit in number of iterations and computational time for scheme IIb.

To get more insight how the boost helps the iterations, Figure 1 shows the convergence history for the finest mesh. Circles are IIb iterative results obtained using boost, while squares are IIb iterative results for which boost failed. The boost has been started earlier than for the data in Table 3 to allow a failed boost to be demonstrated. Note that the convergence factor per iteration remains quite good. In the bottom graph, the minimum interval between boosts was increased to five. This changes the iterative history considerably, but the number of iterations increases by only one. Figure 2 shows the first 1,000 iterations to indicate the effect of boost on scheme I.

For these thick boundary layers, the P' modification Eq. (8) has no noticeable effect on any of

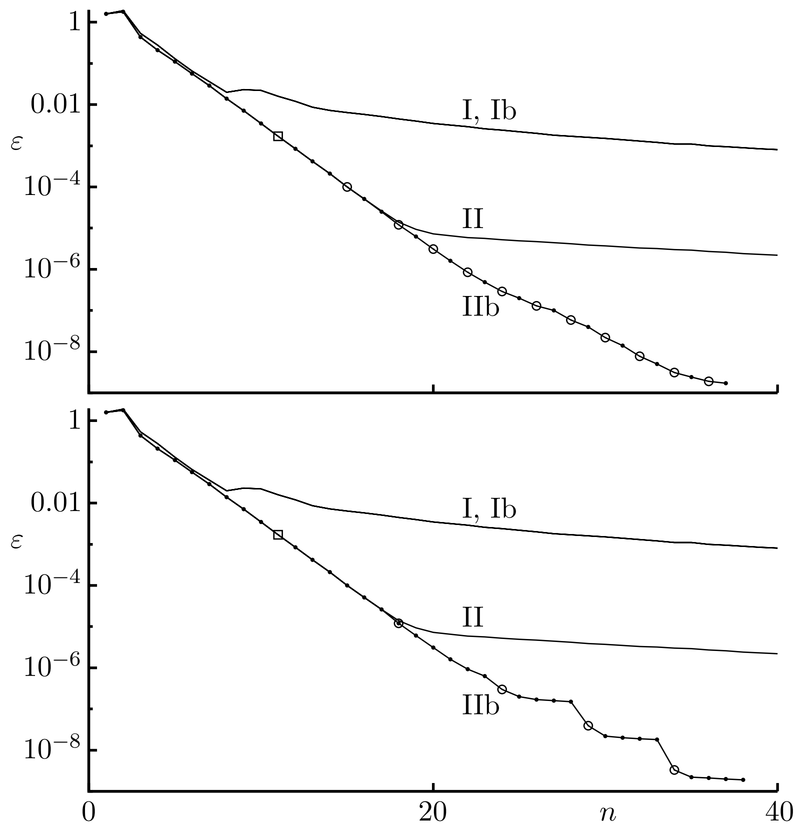

Figure 1: Convergence history for the $2580 \times 144 \times 144$ mesh.

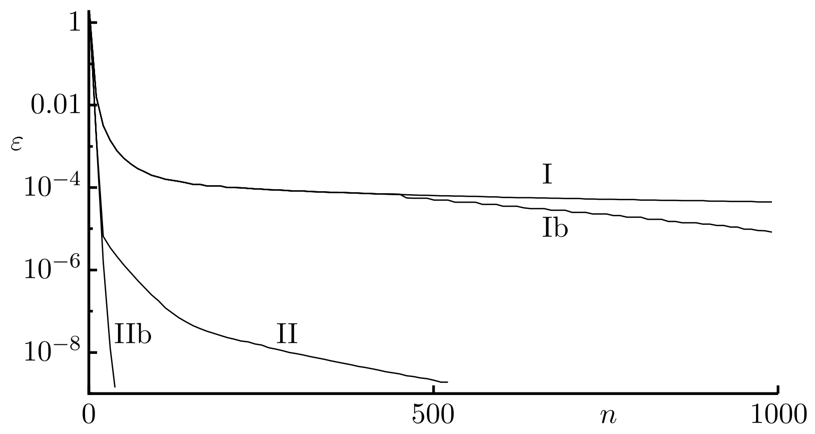

Figure 2: Convergence history for the first 1,000 iterations.

the four methods. The $2580 \times 144 \times 144$ computation was not attempted with $f = 0$ for schemes I and Ib.

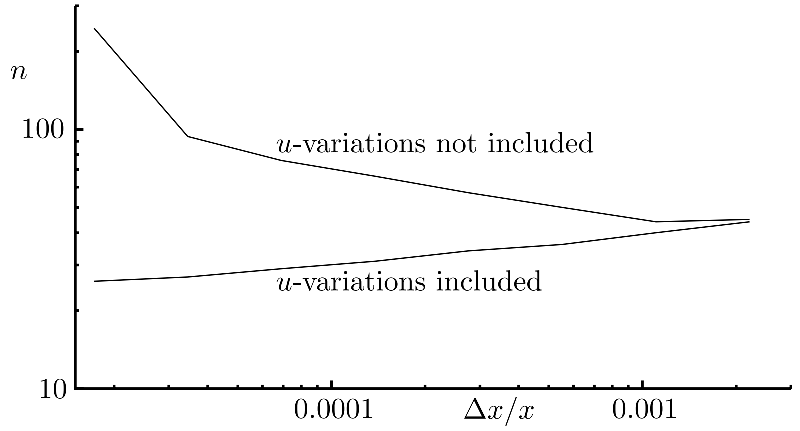

Figure 3: Effect of reducing the time-like step size.

Figure 3 shows the problems in convergence that arise when the time-like step size is reduced near separation. (Similar results are found at a much earlier stage, so the separation does not seem to have much to do with it.) If variations in $u$ are not included in the continuity equation, the iterations increase to 246 when the time-like step is reduced. After another step halving, the iterations diverge. If the correction for the variation in $u$ is included, this no longer happens. It is noted that with this scheme, after three more mesh halvings than shown, convergence to the imposed error of $10^{-8}$ is no longer achieved. That is believed to be due to round-off error, which becomes significant in the streamwise derivatives at such small step sizes. Divergence does not occur.

The above results are typical for a much larger amount of computations done in the context of the analytical studies. For most of the practical computations, a slightly larger, but more accurate, stencil for the convective $y$-derivatives was used. This reduced convergence rates of the unboosted schemes, making boost essential for being able to perform some computations at all.

## 5 Conclusions

While the theoretical justification for the pressure boost method is obviously semi-empirical, it can significantly increase the rate of convergence of the used P' method for problems of the type examined here. These problems include severe mesh stretching in the direction normal to the surface and streamwise velocities that can be very small. The method requires little additional coding effort and no tuning of under-relaxation factors. Because of the undo, it is not sensitive to the frequency at which boost is performed.

Including more of the momentum equations in the solution of the continuity equation was found to be even more efficient for these problems, but that requires more coding effort as block tridiagonal systems must be solved. Also the boost procedure gets more complex. However, a very robust convergence can be maintained over a wide parameter range.

## References

1. J.C. Tannehill, D.A. Anderson, and R.H. Pletcher, *Computational Fluid Mechanics and Heat Transfer,* 2nd ed., pp. 585-592, Taylor & Francis, London, 1997.

2. A.D. Gosman, and D.B. Spalding, The Prediction of Confined Three-Dimensional Boundary Layers, Salford Symposium on Internal Flows, Salford University, Greater Manchester, UK, Paper 19, Inst. Mech. Engrs., London, UK, 1971.

3. P. Hall, The Nonlinear Development of Görtler Vortices in Growing Boundary Layers, *J. Fluid Mechan.,* vol. 193 pp. 243-266, 1988.

4. P. Hall, Görtler Vortices in Growing Boundary Layers: The Leading Edge Receptivity Problem, Linear Growth and The Nonlinear Breakdown Stage, *Mathematika,* vol. 37, pp. 151-189, 1990.

5. S.V. Patankar and D.B. Spalding, A Calculation Procedure for Heat, Mass and Momentum Transfer in Three-Dimensional Parabolic Flows, *Int. J. Heat Mass Transfer,* vol. 15, pp. 1787-1806, 1972.

6. S.V. Patankar, *Numerical Heat Transfer and Fluid Flow,* pp. 126-134, Hemipshere, Washington, D.C., 1980.

7. G.D. Raithby and G.E. Schneider, Numerical Solution of Problems in Incompressible Fluid Flow: Treatment of the Velocity-Pressure Coupling, *Numer. Heat Transfer,* vol. 2 pp. 417-440, 1979.

8. F. Harlow and J. Welch, Numerical Calculation of Time-Dependent Viscous Incompressible Flow of Fluid with Free Surface, *Phys. Fluids,* vol. 8, pp. 2182-2189, 1965.

9. L. Howarth, On the Solution of the Laminar Boundary Layer Equations, *Proc. Roy. Soc. London A,* vol. 164, pp. 547-579, 1938.