Blasius pressure solver

September 26, 2004

1. Summary

The purpose of this report is to examine the Blasius Pressure Solver, to comment on the convergence criteria, and to discuss any possible improvements. A description of the current *direct* fft approach is provided with a discussion of the preferred alternative iterative conjugate gradient technique, and what would be required to implement in the Blasius Code.

The important issues of convergence to the solution are discussed, and an inter-comparison of some tests applicable to the microscale model is analysed. The speed of convergence of the GCR(k) algorithm is dependent on the required "accuracy" of the solution, not significantly the case for the current Blasius fft scheme. Overall, Blasius cpu cost is typically slightly greater, less than a factor of two, than for the GCR(k) approach. To implement the GCR(k) method would require an effective code rewrite, as the boundary conditions would need to be included in the solver.

In conclusion, therefore, it is suggested that the Blasius model is not particularly inefficient and where the steep gradient limitation does not present a problem, as in most meteorological flows, then "good" solutions are apparent. A rewrite of the code to include the boundaries in the pressure solver for implementation of the GCR(k) method is the other alternative.

2. Background

For Anelastic / Boussinesq models, the pressure solver presents a significant computational component. In the first of such models such as Clark(1977), and their recent derivatives, e.g. Smolarkiewicz(1994), pressure solvers can take 50% of the cpu time (unpublished communication with Piotr Smolarkiewicz), with Blasius (2003) taking perhaps 60-80% (Simon Vosper, unpublished communication). For iterative methods in the GCR(k) methods, the smaller the residual errors, the greater the costs. Current examples presented below support these cpu figures per time-step, but with the GCR(k) pressure solvers taking more cpu with increased accuracy. (In this case k=3, but typically k=?5)

Historically for atmospheric models, stability of the pressure solver was important, but now the accuracy of a solution of the Poisson type problem is regarded as just as important. Advection schemes such as Ultimate Quickest, TVD, centred differences have improved greatly (e.g. Wood, 2003). However, the issue of accurate pressure solvers have only been addressed more recently, especially when co-ordinate transformations are implemented. In non-hydrostatic compressible models, for example the Met Office New Dynamics, there is a need to solve a Helmholtz equation, or complete some form of time splitting for acoustic waves, as in RAMS. This is discussed later, $6 and chapters 14 and 15 and Appendix J in UM(2003). The Met Office code uses GCR(k) a similar conjugate gradient approach to that used in Smolarkiewicz (1997). The use of a preconditioner can improve convergence rates, Thomas (2003). This GCR(k) solver is of primary interest here.

3. Methodology

Initially, an analysis of the accuracy of the computations is discussed, primarily with the free slip lower boundary condition with a presentation of results of computation. This is followed by a discussion of the Blasius solver, the GCR(k) approach and how it is used in the UM and Smolarkiewicz model. Finally the implications are summarised.

4. Computational Results

A simplest of cases is implemented to examine the characteristics of the pressure solver. The set up has a neutral atmospheric profile, constant zonal wind of 10ms^{-1}, for a cyclic domain 50*50*50, dx=dy=60m, dz= 50m, with a centred Witch of Agnesi hill of half width of 300m, and an amplitude varying from 300m, 225m, 150m (and also 100m) with maximum gradients of 0.844, 0.633 and 0.422 respectively. Relevant parameters examined are the divergence, the time step, the non-dimensionalised divergence, non-dimensional divergence $dt/rho * div(rho * g^{half} * v)$ and nominal cfl value. (see section $1.5 Blasius, 2003).

The pressure solver establishes a pressure field consistent with the velocity field. Therefore, the divergence should ideally be zero after each computation; the magnitude, gives an indication of convergence independent of the time step used. Using a small time-step helps convergence, as the flow changes are small, so divergence non-dimensionalised by multiplying by dt is sometimes used for comparison.

The UM and the Smolarkiewicz model use the GCR(k) approach, but have different equation sets (compressible and anelastic). The Smolarkiewicz simulations use the same equation set to Blasius model, and therefore most applicable for the study. The comparison is not intended to define the ”reference flow” more to examine the differences for the pressure solver.

4.1 The effect of time-step on pressure solver convergence

A free slip condition was primarily implemented for this comparison. The Blasius model automatically adjusts the time step to satisfy the imposed cfl criteria. The boundary layer parameterisation scheme is a critical factor in determining the new time step value, the limitation between cfl upper and cfl lower limits (see subroutine monitr). Thus changing these limits is a method of controlling the time-step, and this was the method used to control the time-step.

Conventional experience suggests that Blasius cannot be used for steep slopes. Table 1 indicates that for the ”Witch of Agnesi hill”, with an amplitude and half width of 300m and maximum gradient 0.84. By reducing the time steps used, the simulations run for significantly longer before the pressure solver becomes non-convergent. The pressure solver operates by using the previous values, and by implication, a small time step with a small field change, enhances convergence. One might reasonably expect convergence to take iteratively longer for larger time-steps, but this result shows that unless the time step is small then a solution may not be possible, with an implication that this is limitation in the method of solution. For this setup, pressure gradient convergence was possible for a maximum gradient of $\sim$ 0.92.

Amplitude	cfl lower and upper limit	average time-step	solver convergence breakdown
300m	0.2 - 0.4	0.5s	89s
300m	0.05 - 0.1	0.005s	180s
300m	0.02 - 0.04	0.0008s	288s

Table 1: *Table to indicate the time of the Blasius pressure solver convergence breakdown, for a ”Witch of Agnesi” hill with an amplitude and half width of 300m. A smaller time step enables convergence to be satisfied for longer. This is a free slip simulation.*

4.2 The effect of hill amplitude on convergence

Table 2 shows a comparison of the pressure solver breakdown for different slopes. Using a relatively large cfl upper limit, the pressure solver has more difficulty with the steeper slopes. Reducing the cfl condition, as in Table 1, likewise maintains a convergent solution for longer. The time steps do not alter significantly for this comparison. The terrain following co-ordinate grid structure is the same for these runs. With a wind speed of 10ms^{-1} and cyclic continuity applied, the air will cross the domain in 300s, but it is interesting to note that even with 100m hill height, convergence breakdown occurs after 436s even when the flow could be expected to be well established.

Amplitude	cfl lower and upper limit	average time-step	solver convergence breakdown	max gradient
300m	0.2 - 0.4	0.5s	89s	0.84
225m	0.2 - 0.4	0.6s	160s	0.63
150m	0.2 - 0.4	0.6s	250s	0.42
100m	0.2 - 0.4	0.6s	436s	0.28

Table 2: *Table to indicate the time of the Blasius pressure solver convergence breakdown, for a ”Witch of Agnesi” hill, half width of 300m and for different amplitudes. The convergence breakdown occurs earlier for steeper slopes. This is a free free slip simulation.*

4.3 Pressure solver convergence dependency on tolerance

In the pressure solver, a variable ”ptol” (subroutine fft.F) controls the required accuracy limit for the pressure solver. Using the example of 225m amplitude, in Table 2, for a free slip run, the value of ptol was varied from 10^{-5} to 10^{-10}. Given 8 byte precision, this is well within the machine tolerance. The solutions varied only marginally for these simulations, with the smaller tolerance taking marginally longer, and the pressure solver breakdown occurring at the same time. Like-wise increasing the maximum number of iterations to 300 to satisfy the tolerance criteria, made no difference. Both confirm that the solution is not iteration limited. N.B. The latest versions of the tolerance criteria use a relative pressure error tolerance evaluation (S.V.). However, for this study of the effects of the solver, this is not thought to be critical.

4.4 Differences for pressure convergence for no-slip and free slip simulations

Blasius is normally used in a no-slip mode, since one of its advantages is a sophisticated boundary layer scheme, and here the differences are examined. For the free slip formulation, the maximum slope allowed is $\sim$ 0.92, corresponding to amplitude $\sim$ 320m. In the no-slip case, with a mixing length of 50m, the pressure solver does not converge for a 300m hill height, but does converge when the maximum gradient is $\sim$ 0.78, hill height of 275m, as discussed below. For a gradient of 0.66, an amplitude of 225m, the no-slip code runs efficiently, with an average time step of 1.2s (Table 3). The time step is approximately twice as large as for the free slip case, and generally, for all free-slip / no slip equivalent solutions, the pressure solver converges more quickly. The pressure solver is able to obtain a solution more easily when there is a boundary layer mixing scheme. Gradually growing the hill was not tested, but could be expected to increase the maximum allowable gradients.

Amplitude	cfl lower and upper limit	average time-step	solver convergence breakdown	lower b.c
225m	0.2 - 0.4	0.6s	160s	free slip
225m	0.2 - 0.4	1.2s		no slip

Table 3: *No-slip and free slip comparison for the Blasius pressure solver, corresponding to row 2, Table 2. The no slip formulation does not fail to converge. It is suggested that this is due to the elimination of large divergences at the lowest levels due to the boundary layer scheme’s modification of the near surface flow*

Reducing the time-step increases the length of integration, as ahown in Table 2. The tests were not exhaustive, and a convergent solution was obtained for the free slip simulation with a hill height of 320m, a maximum slope of $\sim$ 0.92, with a small time-step of 0.003s, for which the solution eventually becomes unstable at $\sim$ 280s. It is possible that with a still smaller time-step a stable solution could exist for longer, or even without an instability being generated.

Although not displayed in the tables, for both the free slip and the no slip models, a convergent Blasius solution is obtained for a 275m high hill (max gradient $\sim$ 0.78). For the no slip simulation, the time step, which is controlled by the cfl condition and the boundary layer turbulence, is more effected by velocity gradients and gradually increases from 0.02s to $\sim$ 1.2s; for the free slip simulation, the time step was limited to $\sim$ 0.2s.

In summary, the free slip version has pressure convergence problems with large time steps with the pressure solver instability dominated by large values of the divergence in the near surface layers, as discussed in the next section, further suggesting that the boundaries present difficulties for the scheme.

4.5 Divergence characteristics of the pressure solvers

The comparison between the two models indicates the effects of the use of the GCR(k) solver in the Smolarkiewicz code on the divergence characteristics.

As referred to in \$4.0, the divergence $div(rho * g^{half} * v)$ (section \$1.5 Blasius, 2003) should be zero at each time step. The solution iterates to minimise this value to the required tolerance level of $\sim 10^{-09}$.

Figures 1 and 2 for Blasius, and 3 Smolarkiewicz, summarise the divergence, cfl and velocity patterns obtained for a free slip lower boundary condition for the 300m amplitude hill solutions. Cyclic continuity ensures that the air flows across the domain in about 300s, but after about 250s the computed flow field appear to be stable.

Two levels are chosen; the surface, level 1, Figures 1(a) and (b) and $\sim$ 100m at level 3, Figures 2(a) and (b). For the free slip simulations, for the Blasius model, the greatest non-dimensional divergence values are always at the surface of the hill. In this test problem, the largest cfl value 2(a) is approximately in the region of the summit and the steepest slope of the hill. With relatively steep slopes, the cfl value has a significant contribution from the vertical velocity. For the Blasius run, the maximum non-dimensional divergence is $\sim 6*10^{-03}$, similar to the cfl value. This is of concern the pressure equation is consequently having difficulties in converging to produce a zero convergence, necessary for the solution of the equations. For these solutions, as given in Table 1, the time-step is $\sim$ 0.001s, and thus the actual divergence is $\sim 6s^{-1}$

At the region of the maximum cfl value, the non-dimensional divergence term is much smaller $\sim 1*10^{-03}$. This result is in common with other runs. In summary, it is the steep slopes at the lower boundary which provides the problem for the pressure solver.

Figure 3 displays results from the Smolarkiewicz model indicating the non-dimensional divergence directly equivalent to 1(b). The non-dimensional divergence, has a maximum value of 0.094, (dt = 2s, max cfl $\sim$ 0.66). For the Blasius simulation 1(b) this has a value of 0.006, (dt $\sim 10^{-03}$s , max cfl $\sim$ 0.006).

Evaluating the divergence, i.e. by dividing by the timestep, is the alternative way of highlighting the problems for the pressure solver. When the non-dimensional divergence is of the same order as the cfl condition, numerical problems arise. For the 300m hill the the actual maximum divergence (which should be zero for an ideal solution) is $\sim 0.05s^{-1}$ for the GCR solution and $\sim 6s^{-1}$ for Blasius. Soon after the Blasius code produces these values, the code integration ceases.

4.6 Model vertical velocity

For completeness, the values of the vertical velocity have been included. It is not clear that the free slip applied in Blasius and Smolarkiewicz is the same, and the boundary layer scheme in Blasius is definitely different from use no boundary layer scheme, and only Smagorinsky type closure in Smolarkiewicz. How-

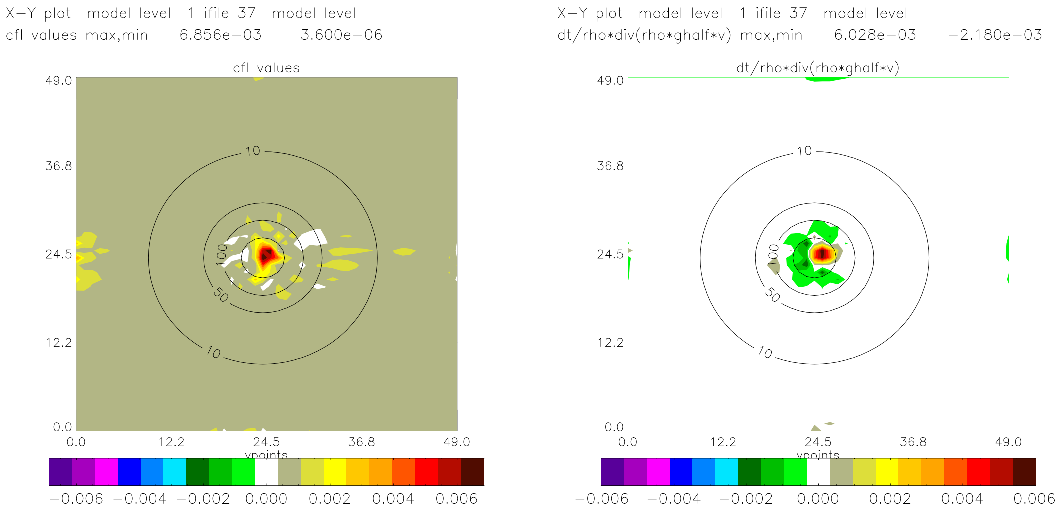

Figure 1: *(a) and (b). Model output from the Blasius model at level 1, the surface. (a) Left: Values of "cfl" and (b) Right: values of non-dimensional divergence.*

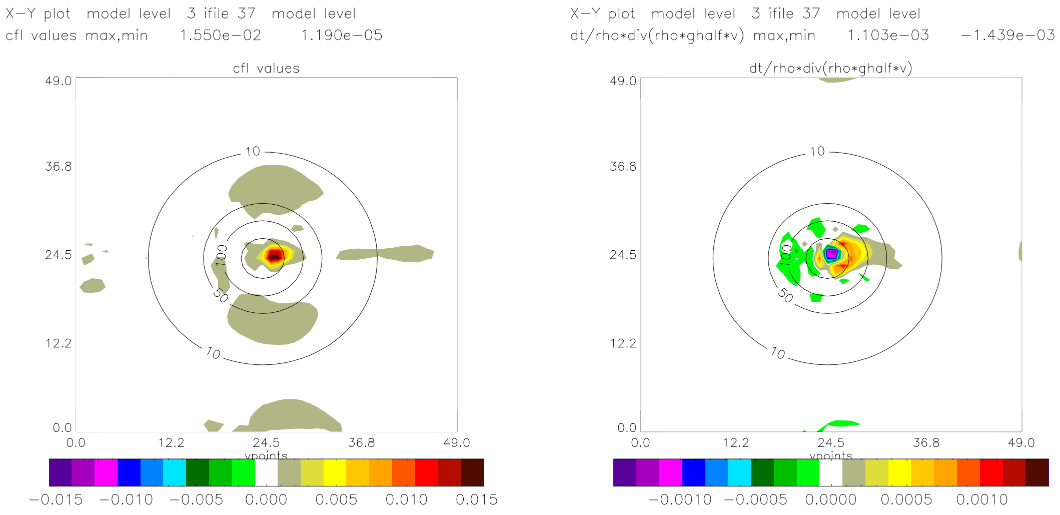

Figure 2: *(a) and (b). Model output from the Blasius model at level 3, approximately 100m. above the surface. (a) Left: Values of "cfl" and (b) Right: values of non dimensional divergence*

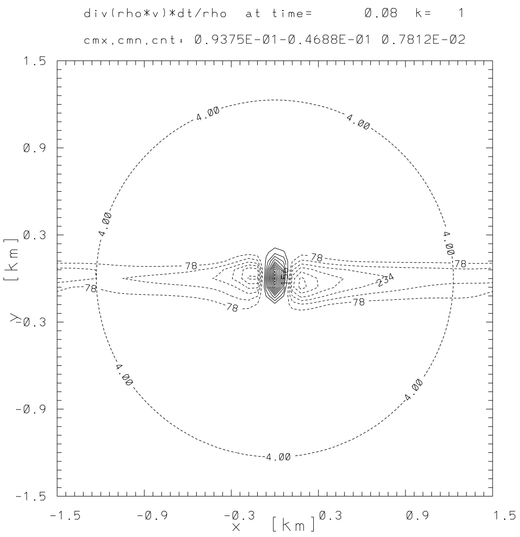

Figure 3: *Model output from the Smolarkiewicz model at the surface for values of non dimensional divergence. For actual divergence, the values should be divided by approximately 2s, the value for the timestep.*

ever, Figures 4 and 5 are included displaying the vertical velocity plots; 4(a) and 4(b) show the free slip and no slip Blasius velocities, and 5(b) and 5(c) are for Smolarkiewicz. I would argue that the values are similar, for what is a relatively unrealistic severe test.

5. Blasius' Pressure Solver

The case of interest is the pressure solver for 3-D with surface topography. Full details are given in (e.g.. Blasius, 2003), specifically $1.5, equation 1.28 in version 4, (equation 28 in version 3, as below) which refers to the derivation of the terms in the equation pressure equation. In brief:-

∇^2 Pressure = Source terms independent of Pressure + other Pressure terms

or

$$
\begin{aligned}
\partial_x^2\left(G^{\frac{1}{2}}P\right)+\partial_y^2\left(G^{\frac{1}{2}}P\right)+\partial_z^2\left(G^{\frac{1}{2}}P\right) \quad &= \partial_x^2\left(\left(G^{\frac{1}{2}}-1/G^{\frac{1}{2}}\right)P\right)\partial_x\left(G^{\frac{1}{2}}S_U\right)+\partial_y\left(G^{\frac{1}{2}}S_V\right)+ && (1)\\
&\left(1/G^{\frac{1}{2}}\right)\partial_z\left(G^{\frac{1}{2}}S_W+C_xG^{\frac{1}{2}}S_U+C_yG^{\frac{1}{2}}S_V\right) && (2)\\
&-\partial_xP_U-\partial_yP_V && (3)\\
&\left(1/G^{\frac{1}{2}}\right)\partial_z\left(C_x\partial_x\left(G^{\frac{1}{2}}P\right)+C_y\partial_y\left(G^{\frac{1}{2}}P\right)+C_xP_U+c_yP_V\right) && (4)\\
&\partial_t\left(\partial_x\left(G^{\frac{1}{2}}\rho U\right)+\partial_y\left(G^{\frac{1}{2}}\rho V\right)+\partial_z\left(G^{\frac{1}{2}}\rho w\right)\right) && (5)
\end{aligned}
$$

This equation is fully described in equation 1.28, Blasius (2003), Note that in line 5 the vertical partial derivate of the continuity equation is computed as a term on the rhs of the equation.

Sub-routine poibnt in the "fft" package includes the iteration of the rhs of the equation (above) depending on the tolerance, ptol, and an iteration count, lcount. The maximum relative pressure error is compared with the tolerance, ptol. N.B. See also comment in $4.3

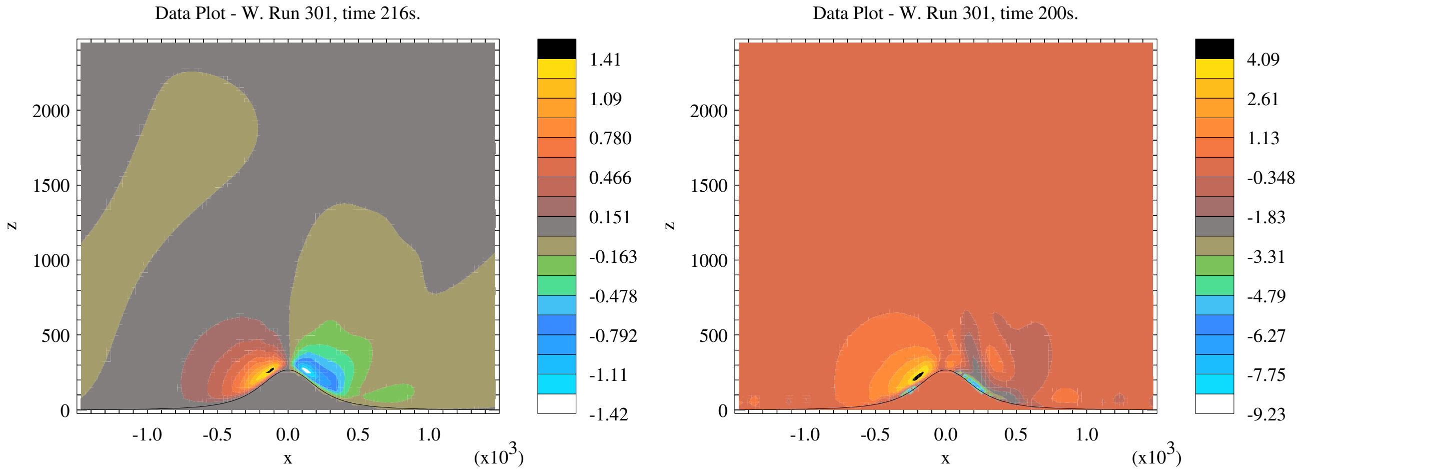

Figure 4: *(a) and (b). Model output from the Blasius model across the hill and parallel to the wind. (a) Left: no-slip and (b) Right: free slip.*

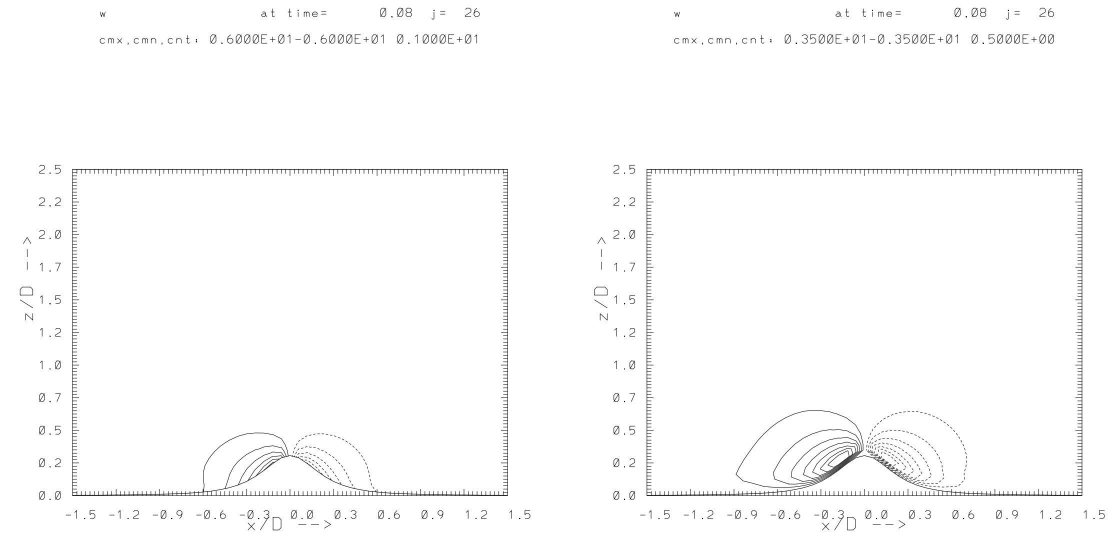

Figure 5: *(a) and (b). Model output from the Smolarkiewicz model across the hill and parallel to the wind. (a) Left: free slip and (b) Right: no slip.*

Usage of different tolerances, varying the default value of 10^{-9} from 10^{-05} to 10^{-11} had an insignificant effect on the solutions for a 225m hill (Table 2, row 2) with 8 byte real precision. The differences in the residual divergence were minor. Increasing the iteration maximum count, from 100 to 300 had no effect at the pressure solver breakdown point. This further suggests that the numerical scheme is "converging" and not the source of the error.

The fft solver applied to the equation above determines a new value for the pressure, and as described above, computation ceases when numerical tolerance is achieved. However, the rhs of the equation also contains other pressure terms. From the previous analysis of the results the instabilities arise from the near surface values, especially the surface generated pressure gradients. Use of the terrain following co-ordinates produces significant geometric distortions in the near surface geometry. The values of pressure at the surface are determined from those at level 2, and the fact that the boundary values are not recalculated each iteration could cause a problem. It would be interesting to ascertain if inclusion of the boundary conditions would overcome this. This would involve bringing in the pressure boundary terms into the code at this point, and would require a major code modification. This approach would also be needed for implementation of the GCR solver.

6. GCR(k) Pressure Solver as implemented in Smolarkiewicz and UM models

For background references, the original variational method is based on Eisenstat et al (1983). Thomas et al.(2003) explains the use of spectral preconditioners (and the GCR(k) method in appendix B), Smolarkiewicz et al. (1997) provides an analysis of the stopping criteria used in this technique. Appendix J, p.j9-17 UM, 2002, specifically discusses the GMRES solution applied to the UM model.

The aim is to solve a linear system P^{-1} Ax = P^{-1}b (UM notation), with convergence criteria $||$Ax - b$|| <$ e $||b||$. Iterative methods often work for well conditioned systems, with the boundary conditions included in matrix A and b. Difficulties arise because the rate of convergence relies on COND(A), and the meteorological systems are often ill conditioned. Multiplication of both sides by a suitable preconditioner can become important ... i.e. changing the problem from L(Ψ) = Q to P^{-1}[L(Ψ) - Q] = 0, (Smolarkiewicz notation). Theory suggests that closer clustering of the eigenvalues speeds convergence and this is the purpose of the preconditioning.

Preconditioning matrices generally can enhance convergence rates. In the Blasius solver, pressure terms can be found on both sides of the equation. Preconditioners apply to the lhs matrix operation L, in the GCR(k) case, but would also need to apply to the awkward rhs pressure terms in the Blasius rhs. All pressure terms need to be operated on by the preconditioning matrix, and it is difficult to see how this could be achieved without rewriting the discretisation structure; i.e. the "awkward" pressure terms on the rhs, would need to be moved. Preconditioning is primarily aimed at overcoming the anisotropy in the vertical, c.f. the horizontal. For some microscale flows the co-ordinates can be almost "rectangular", as in the above test examples and although an aid in the computation efficiency, in this case, preconditioning may not be the most important issue which produces much better convergence compared with the Blasius fft approach. This is also suggested by the above numerical example, which indicates the near boundary calculated values, rather than boundary conditions were providing the source of the non-zero divergence terms in the grid points close to the boundaries. It should be noted that over steep terrain, with curvilinear co-ordinates anisotropy is always introduced with the squashing of surface contours.

It is recognised that elliptic problems are dependent on the imposed boundary conditions. Smolarkiewicz's et al (1997) contribution is to apply the iterative techniques in this area. Ensuring that the normal component of the pressure gradient includes the x,y and z derivatives correctly is important.

7. Summary

With steep terrain and a Gal-Chen co-ordinate system which produces some anisotropy even in rectangular type problems, dx=dy=dz, with preconditioning the GCR(k) pressure solver does produce stable and

”converged” solutions where the Blasius type FFT based solutions does not appear to. The solutions for slopes up to 4̃5degrees do not differ significantly, and the computation time does not seem grossly different, (within a factor of two). There are heavy overheads on the first preconditioning iteration for the iterative scheme which slows down computations.

Elliptic problems are implicitly sensitive to the boundary conditions, and significant non-zero divergences are produced near the boundaries in the Blasius code, which is the source of the instabilities. An attempt was made to incorporate the GCR(k) solver into the Blasius code, but it was apparent that the pressure equation would need to be rewritten with the pressure terms on the lhs and full incorporation of the boundary conditions implemented at each step. It is concluded, that to implement an improved iterative pressure solver properly, the Blasius code should be rewritten.

7. References.

Alexander, G.D., J. Weinman, V. Karyampudi, W.Olson and A. Lee, 1999: The effect of assimilating rain rates derived from satellites and lightning on forecasts of the 1993 superstorm. *Mon. Wea. Rev.*, **127**, 1433-1457.

Blasius, 2003: Blasius version 4.00 (also 3.03) Documentation, UK Met office.

Clark, T.L. , 1977: *Jou. Comp. Phys*, 24, 186-215

Eisenstat, S. H. elman and M, Schultz, 1983: Variational iterative methods for nonsymmetric systems of linear equations. *SIAM J. Numer. Anal.* 2, 345-357

Thomas, S.J., J. Hacker, P. Smolarkiewicz and R. Stull, 2003: Spectral preconditioners for Non-Hydrostatic Atmospheric Models. *Mon. Wea. Rev.*, 131, 2464-2478

Smolarkiewicz, P. and L. Margolin, 1994: Variational solver for elliptic problems in atmospheric flows. *App. Maths Computer Sci.* 4,4, 527-51

Smolarkiewicz, P., L. Grubisic and L. Margolin, 1997: On forward in time differencing for fluids: Stopping criteria for iterative solutions of anelastic pressure equations. *Mon. Wea. Rev.*, 125, 4, 647-54

UM (2002) UM 5.3 documentation, UK Met Office.

Williams, E. R., and R. M. Dole, 1990: Meteorological and electrical conditions associated with positive cloud-to-ground lightning. *Mon. Wea. Rev.*, **118**, 470-487.

Wood, N.