Parallel DNS of a separating turbulent boundary layer

Martin Skote and Dan Henningson
Department of Mechanics, KTH, S-100 44 Stockholm, Sweden
Naoki Hirose, Yuichi Matsuo and Takashi Nakamura
Computational Science Division, National Aerospace Laboratory Jindaiji-Higashi
7-44-1, Chofu-shi, Tokyo 182-8522, Japan

Abstract

A direct numerical simulation of a turbulent boundary layer has been performed on parallel super computers. The boundary layer is subjected to a strong adverse pressure gradient which creates a separation bubble followed by a region with small, but positive, skin friction. This flow case contains features that has proven to be difficult to predict with turbulence models. The data from the simulation are used for investigation of the scalings near the wall, a crucial concept with respect to turbulence models. One of the largest and fastest parallel computers for this type of calculations has been used for the simulation, the Numerical Wind Tunnel at the National Aerospace Laboratory in Tokyo. The parallelization of the code for computers with distributed memory has been done using MPI (Message-Passing Interface).

1 Introduction

The near wall scaling of the mean velocities are very important for the correct behavior of wall damping functions used when turbulence models are used in the Reynolds averaged Navier-Stokes equations (RANS). For a zero pressure gradient (ZPG) boundary layer, the damping functions and boundary conditions in the logarithmic layer are based on a theory where the friction velocity, u_τ, is used as a velocity scale. However, in the case of a boundary layer under an adverse pressure gradient (APG), u_τ is not the correct velocity scale, especially for a strong APG and moderate Reynolds number. In the case of separation this is clear since u_τ becomes zero. The combination of a pressure gradient and moderate Reynolds number give a flow that deviates from the classical near wall laws. The near wall behavior of a turbulent boundary layer close to separation is studied using direct numerical simulations (DNS) on massively parallel computers. The results are analyzed and can be used to improve the near wall behavior in turbulence models for flows with separation.

2 Numerical method and parallelization

The code used for the direct numerical simulations (DNS) is developed at KTH and FFA [1]. The numerical approximation consists of spectral methods with Fourier discretization in the horizontal directions and Chebyshev discretization in the normal direction. Since the boundary layer is developing in the downstream direction, it is necessary to use non-periodic boundary conditions in the streamwise direction. This is possible while retaining the Fourier discretization if a fringe region is added downstream of the physical domain. In the fringe region the flow is forced from the outflow of the physical domain to the inflow. In this way the physical domain and the fringe region together satisfy periodic boundary conditions.

Time integration is performed using a third order Runge-Kutta method for the advective and forcing terms and Crank-Nicolson for the viscous terms. A 2/3-dealizing rule is used in the streamwise and spanwise direction.

The numerical code is written in FORTRAN and consists of two major parts (figure 1), one linear part (`linear`) where the actual equations are solved in spectral space, and one non-linear part (`nonlin`) where the non-linear terms in the equations are computed in physical space. All

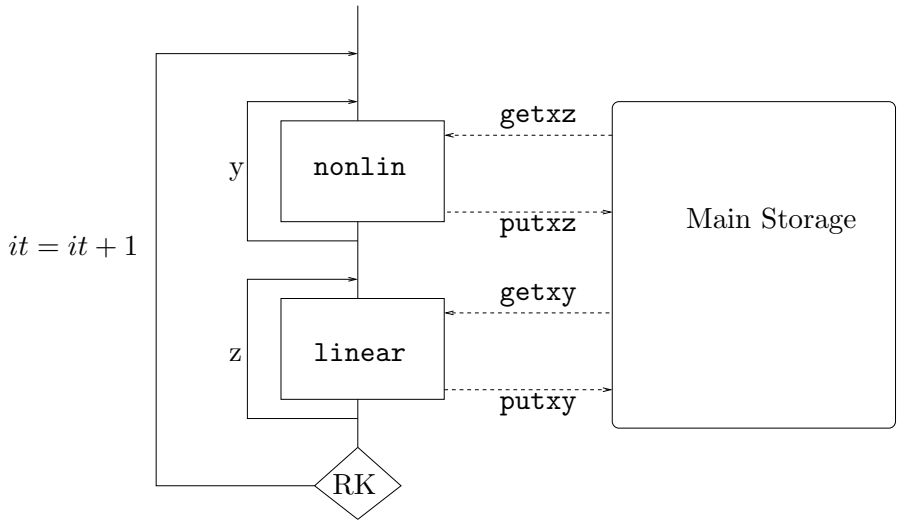

Figure 1: The main structure of the program.

spatial derivatives are calculated in the spectral formulation. The main computational effort in these two parts is in the FFT.

In the linear part one xy-plane is treated separately for each z -position. The field is transformed in the y direction to spectral space, a solution is obtained and then transformed to physical space in the y direction. This is performed with an loop over all z values where the subroutine `linear` is called for each z. The xy-planes are transferred from the main storage with the routine `getxy` to the memory where the actual computations are performed. The corresponding storing of data is performed with `putxy`.

In the non-linear part the treatment of the data is similar to that in the linear part. One xz-plane is treated separately for each y- position. The field is transformed in both the x and z directions to physical space where the non-linear terms are computed. Then the field is transformed in the x and z directions to spectral space. This is performed with a loop over all y values where the subroutine `nonlin` is called to at each y. The xz-planes are transferred from the main storage with the routine `getxz` to the memory where the actual computations are performed. The corresponding storing of data is performed with `putxz`.

Communication between processors is necessary in the two different parts of the code. The data set (velocity field) is divided between the different processors along the z direction, see figure 2a. Thus, in the linear part, no communication is needed since each processor has data sets for z-positions (xy-planes). When the non-linear terms are calculated, each processor needs data for a horizontal plane (xz-planes). The main storage is kept at its original position on the different processors. In the non-linear part each processor collects the two dimensional data from the other processors, on which it performs the computations, and then redistributes it back to the main storage. Figure 2b shows an example of the data gathering for one processor.

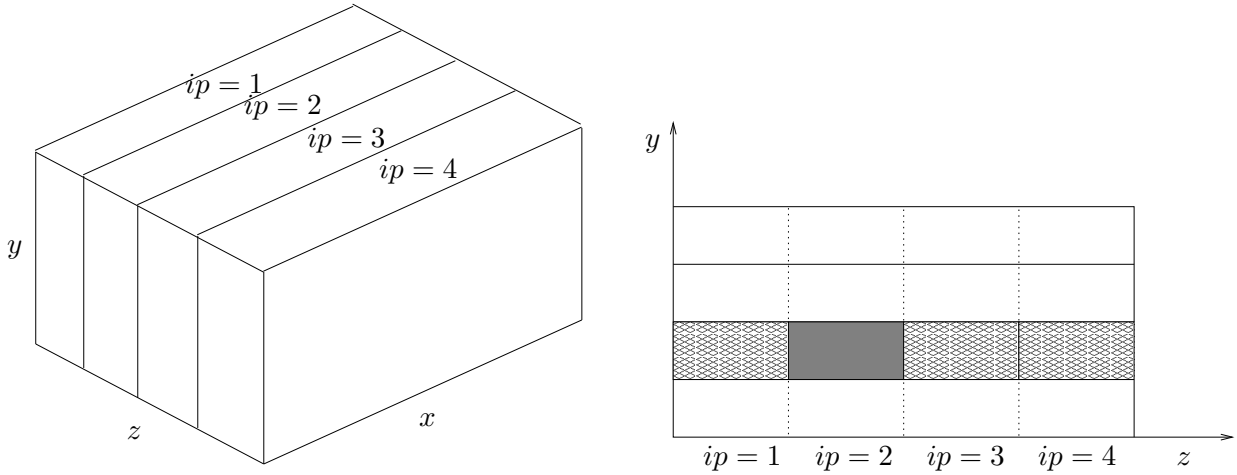

Figure 2: a) The distribution of the main storage on four processors $ip = 1, ..., 4$. b) The gathering of data in the nonlinear part (`nonlin`) of the code for processor number two. The completely shaded area is local on the processor and need not to be received from the others, and the half-shaded area is sent to processor number two. The x-direction is omitted for clarity.

2.1 Numerical parameters

The simulation was performed on various computers. The tuning of the pressure gradient for the desired flow situation was performed on a Cray T3E at NSC in Linköping, using 32 processors. After the design of the pressure gradient, a simulation with 20 million modes was performed on a IBM SP2 at PDC, KTH in Stockholm, using 32 processors. The results presented here are from the second simulation with 40 million modes performed at the National Aerospace Laboratory (NAL), Tokyo. The same code was used on all three computers, using MPI (Message-Passing Interface) for the communication between the processors.

The computer used at NAL was the Numerical Wind Tunnel (NWT), a parallel computer that consists of 166 vector processors from Fujitsu. The maximum performance on each processor is 1.7 Gflop/s. The main difference from the other two computers (CRAY T3E and IBM SP2) is the type of processor. While the other two consist of super-scalar processors, the NWT utilizes vector processors, which give a higher performance for each of the processing elements. For comparison between the three computers for the full simulation, see table 1.

	T3E	SP2	NWT
peak processor performance	600	640	1700
code performance per processor	30	50	320
total performance on 64 processor	1900	3200	20500

Table 1: The performance of the code given in Mflop/s for the 20 million mode simulation on T3E and SP2, and 40 million mode simulation on NWT.

The simulations start with a laminar boundary layer at the inflow which is triggered to transition by a random volume force near the wall. All the quantities are non-dimensionalized by the freestream velocity (U) and the displacement thickness (δ^*) at the starting position of the simulation ($x = 0$) where the flow is laminar. At that position $R_{\delta^*} = 400$. The length (including the

fringe), height and width of the computation box were 700 × 65 × 80 in these units. The number of modes in this simulation was 720 × 217 × 256, which gives a total of 40 million modes or 90 million collocation points. The fringe region has a length of 100 and the trip is located at $x = 10$.

The simulations were run for a total of 7500 time units (δ^*/U), and the sampling for the turbulent statistics was performed during the 1000 last time units.

2.2 Performance of the code

In figure 3 the performance of the code on the two super scalar computers is shown as Mflop/s together with the optimal speed. This is a small test case and the performance is lower than for a real simulation on many processors. The scaling is better on the T3E, while the overall performance is better on the SP2, which is approximately twice as fast as the T3E. The NWT is over six times as fast as the SP2, which give a performance of 20 Gflop/s on 64 processors for the simulation with 40 million modes, table 1.

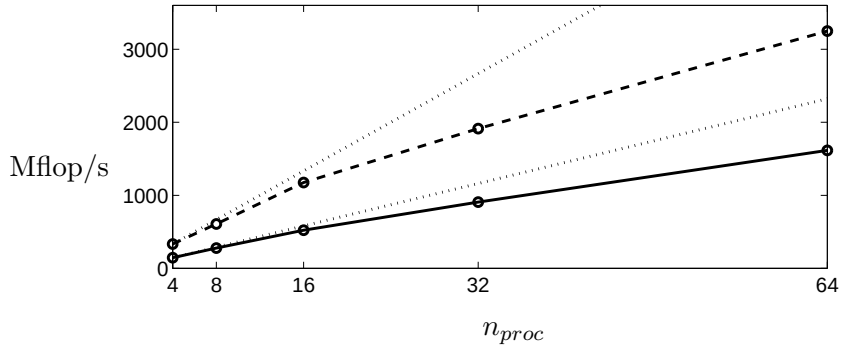

Figure 3: Mflop/s rates for different number of processors for a small test case. — T3E - - SP2 ⋯ maximum speed up.

3 Results

Results from smaller simulations with weaker pressure gradients have been fully analyzed and presented in [2]. These simulations were an important step towards the strong APG case presented here. The free stream velocity varies according to a power law in the down stream coordinate, $U \sim x^m$. In the present simulation the exponent m is equal to -0.25. The friction velocity, u_τ, is negative where separation, i.e. reversed flow, occurs. The boundary layer has a shear stress very close to zero at the wall for a large portion in the down stream direction as seen from figure 4. The separated region is located between $x = 160$ to $x = 370$.

For a zero pressure gradient (ZPG) boundary layer the velocity profile in the viscous sub-layer is described by $u^+ = y^+$, where superscript + denotes the viscous scaling based on u_τ. Under a strong APG this law is not valid and from the equations the following expression can be derived,

$$u^p = \frac{1}{2} y^{p2} - \left(\frac{u_\tau}{u_p}\right)^2 y^p. \tag{1}$$

The viscous scaling based on u_τ has to abandoned since u_τ becomes zero at separation. A different velocity scale, based on the pressure gradient, can be derived from the equations valid in the near wall region. This velocity, u_p, replaces u_τ as the scaling parameter and the scaled quantities are denoted by superscript p instead of +.

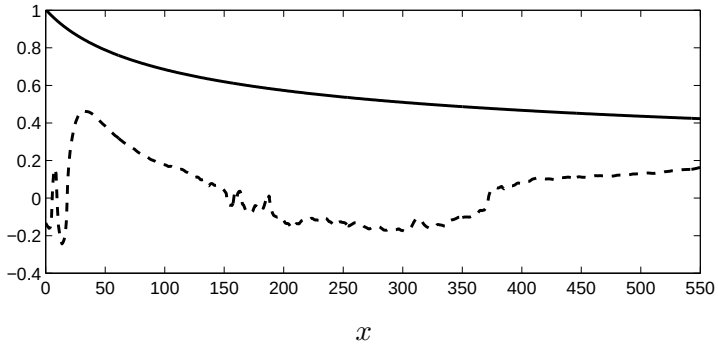

Figure 4: — U; - - $u_\tau \times 10$.

Comparing velocity profiles from DNS with the profile above is done in figure 5. This figure shows velocity profiles near the wall for $x = 250$ and $x = 300$ in pressure gradient scaling. The higher profile is located at $x = 250$. Both profiles are from within the separated region. The solid lines are DNS data and the dashed are the profiles given by equation (1). The data from DNS follows the theoretical curve in a region close to the wall. Equation (1) is valid only in a region where the flow is completely governed by viscous forces. This region is very small at low Reynolds numbers, hence the limited overlap in figure 5. The agreement of DNS data and the theoretical expression in the near wall region indicates that boundary conditions for turbulence models can be improved to give proper results even in a separated region. The near wall behavior is a crucial step in turbulence modeling, and the new results from this simulation has a potential to dramatically improve the wall damping functions.

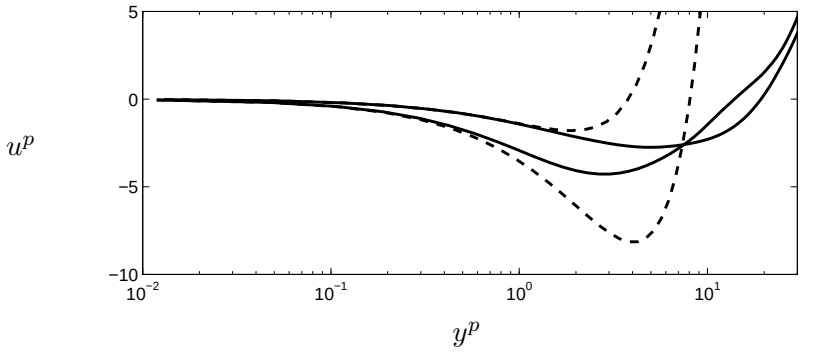

Figure 5: — u^p; - - $\frac{1}{2}y^{p2} - (\frac{u_\tau}{u_p})^2 y^p$.

References

[1] A. Lundbladh, S. Berlin, M. Skote, C. Hildings, J. Choi, J. Kim, and D. S. Henningson. An efficient spectral method for simulation of incompressible flow over a flat plate. Technical Report TRITA-MEK 1999:11, Royal Institute of Techmology, Stockholm, 1999.

[2] M. Skote, R. A. W. M. Henkes, and D. S. Henningson. Direct numerical simulation of self-similar turbulent boundary layers in adverse pressure gradients. *Flow, Turbulence and Combustion*, 60:47–85, 1998.