NAVIER-STOKES SIMULATION OF HOMOGENEOUS TURBULENCE ON THE CYBER 205

C. T. WU,
J. H. FERZIGER,
AND
D. R. CHAPMAN
STANFORD UNIVERSITY
STANFORD, CALIFORNIA

AND

R. S. ROGALLO
NASA AMES RESEARCH CENTER
MOFFETT FIELD, CALIFORNIA

NAVIER-STOKES SIMULATION OF HOMOGENEOUS TURBULENCE ON THE CYBER 205

C. T. WU, J. H. FERZIGER, AND D. R. CHAPMAN

STANFORD UNIVERSITY, STANFORD, CALIFORNIA

AND

R. S. ROGALLO

NASA AMES RESEARCH CENTER, MOFFETT FIELD, CALIFORNIA

ABSTRACT

A computer code which solves the Navier-Stokes equations for three-dimensional, time-dependent, homogeneous turbulence has been written for the Cyber 205. The code has options for both 64-bit and 32-bit arithmetic. With 32-bit computation, mesh sizes up to 64^3 are contained within core of a 2 million 64-bit word memory. Computer speed timing runs were made for various vector lengths up to 6144. With this code, speeds a little over 100 Mflops have been achieved on a 2-pipe Cyber 205. Several problems encountered in the coding are discussed.

1. INTRODUCTION

Turbulent fluid motion is common to many branches of engineering and science. Since turbulence phenomena are highly nonlinear, they are not amenable to classical analytical approaches. Consequently, turbulence predictions are generally based on semi-empirical models. Experiments which generate model information are expensive, but are needed because current models are not generally accurate enough for engineering purposes. Detailed simulations of turbulent flows can help complement laboratory data. Direct numerical simulations of turbulent flows are more accurate than current semi-empirical computational methods and can be used to both generate physical understanding and to improve the models. In these simulations, turbulent flows are directly computed from the Navier-Stokes equations. Computations of this type are necessarily three-dimensional and time-dependent; they require a large number of grid points, and thus, long computation time. The Cyber 205 computer appears ideally suited for efficient numerical simulations of this type. Exploration of the use of the Cyber 205 for direct numerical simulation of turbulence is a principal objective of this work.

The basic code was written by one of the authors (RSR). It was modified to take advantage of the 205 compiler's automatic vectorizing capability. Vector syntax and special functions were applied to the code segments which could not be automatically vectorized. Finally, machine language instructions were used for the parts of the code that existing compiler could not handle.

In the next section, a description of the particular problem to be solved is given. In Section 3, the numerical methods used are discussed. This is followed by a brief description of the Cyber 205 at Colorado State University. The construction of long vectors is discussed in Section 5. In Section 6, performance data obtained to date are presented, and in Section 7, problems encountered are described. A typical simulation of homogeneous isotropic turbulence is presented in Section 8. In the final section, a brief statement of conclusions is presented.

2. PROBLEM STATEMENT

Homogeneous turbulent flows, of which there is a considerable variety, can be simulated numerically at low Reynolds number without using any turbulence model. In the flows we will consider, the computational domain contains a fixed mass of fluid within a rectangular parallelepiped, the opposing sides of which can move inward or outward with time. Thus, the cases which can be computed are quite varied: decaying homogeneous isotropic turbulence is generated if all six sides are stationary; turbulence undergoing uniform compression (or expansion) if all three pairs of sides move inward (outward) at same rate; turbulence undergoing one-dimensional compression, if one pair of sides moves inward; or turbulence undergoing plane strain if one pair of sides moves inward at the same rate a second pair moves outward, while the third pair remains stationary. Isotropic turbulence has been computed before, but turbulence undergoing compression or expansion has not. The compression cases are of interest, for example, in internal combustion engine modeling and in the interaction of turbulence with a shock wave.

It will be assumed that the Mach number is sufficiently small that the fluid is compressed uniformly in space, so that the fluid density depends only on time.

The governing Navier-Stokes equations for a fluid of uniform viscosity and uniform density in space are:

$$\frac{\partial u_i'}{\partial t} + u_j' \mathcal{U}_{i,j} + B_{kj}(u_i' u_j')_{,k} + B_{ki} P_{,k}' = \nu B_{kj} B_{\ell j} u_{i,k\ell}' \qquad i=1,3$$

$$B_{ji} u_{i,j}' = 0$$

where u^1_i p^1, v^2, and t are fluctuating velocity components, fluctuating pressure, kinematic viscosity and time respectively. The summation convention is implied. This set of governing Navier-Stokes equations allow us to simulate homogeneous turbulent flows in Lagrangian coordinate system that moves with the mean flow. Coordinate transformation tensor Bij is determined by:

$$\frac{dB_{ij}}{dt} + B_{ik} \mathcal{U}_{k,j} = 0$$

Note that mean strain rate tensor, $\mathcal{U}_{i,j}$ is zero and $B_{ij}=\delta_{ij}$ for isotropic homogeneous turbulence.

Periodic boundary conditions are applied in all three space directions. The velocity field is initialized to an isotropic state that satisfies continuity and has a given energy spectrum which approximates that of experimental isotropic turbulence.

3. NUMERICAL METHOD

The spectral method is used to compute all spatial derivatives. This method, which uses FFT's, is good for problems with periodic boundary conditions and has very high accuracy. To avoid aliasing in the nonlinear terms, both the truncation and phase shifting techniques are used.[1]

A second order Runge-Kutta method is used to advance the solution in time. Thus, all spatial derivatives need to be computed twice each time step. The time step was chosen small enough that no significant error is produced. It was determined by increasing the step size until the error was approximately 1 percent over the full integration period.

4. THE CYBER 205

The Cyber 205 we are using is the Colorado State machine with 2-pipes and a 2 million 64-bit word fast memory.[2] QTE Telenet has been used for data transfer between Stanford and CSU. We have found that both are reliable, convenient to use, and have provided satisfactory service so far.

Figure 1 shows the performance for add/multiply as function of vector length. The asymptotic performance which requires maximum vector length (65535) is 100 Mflops for 64-bit arithmetic and 200 Mflops for 32-bit arithmetic.[3]

It is obvious that the performance improves with vector length. Vector length 1000 (64-bit case) or 2000 (32-bit case) is required to reach 90 percent of the asymptotic performance. Constructing a code which uses long vectors is therefore important if maximum performance from the machine is to be obtained.

5. DATA MANAGEMENT

Based on the "longer vector gives better performance" philosophy, we chose to do the Fourier transforms in parallel. This will be explained in detail later.

In Figure 2, NX, NY, and NZ are the number of mesh points in the x, y, and z directions respectively; MY and MZ are called "pencil sizes".

On the first sweep, MZ x-y planes of data are Fourier transformed in the y direction in parallel. The transform length is NY, but by doing them in parallel, a vector length of NX/2*MZ*3 is achieved; the factor 3 is due to the simultaneous processing of three velocity components, and the factor 1/2 is due to only half of the modes are needed in wave space to represent a real function in physical space. To accomplish this, it is useful to lump every dependent variable into a single big array. The main array in our code is DATA(NX/2,NY,NZ,4,2); the dimensions represent x, y, z, a dependent variable index, and real and imaginary parts of a complex number.

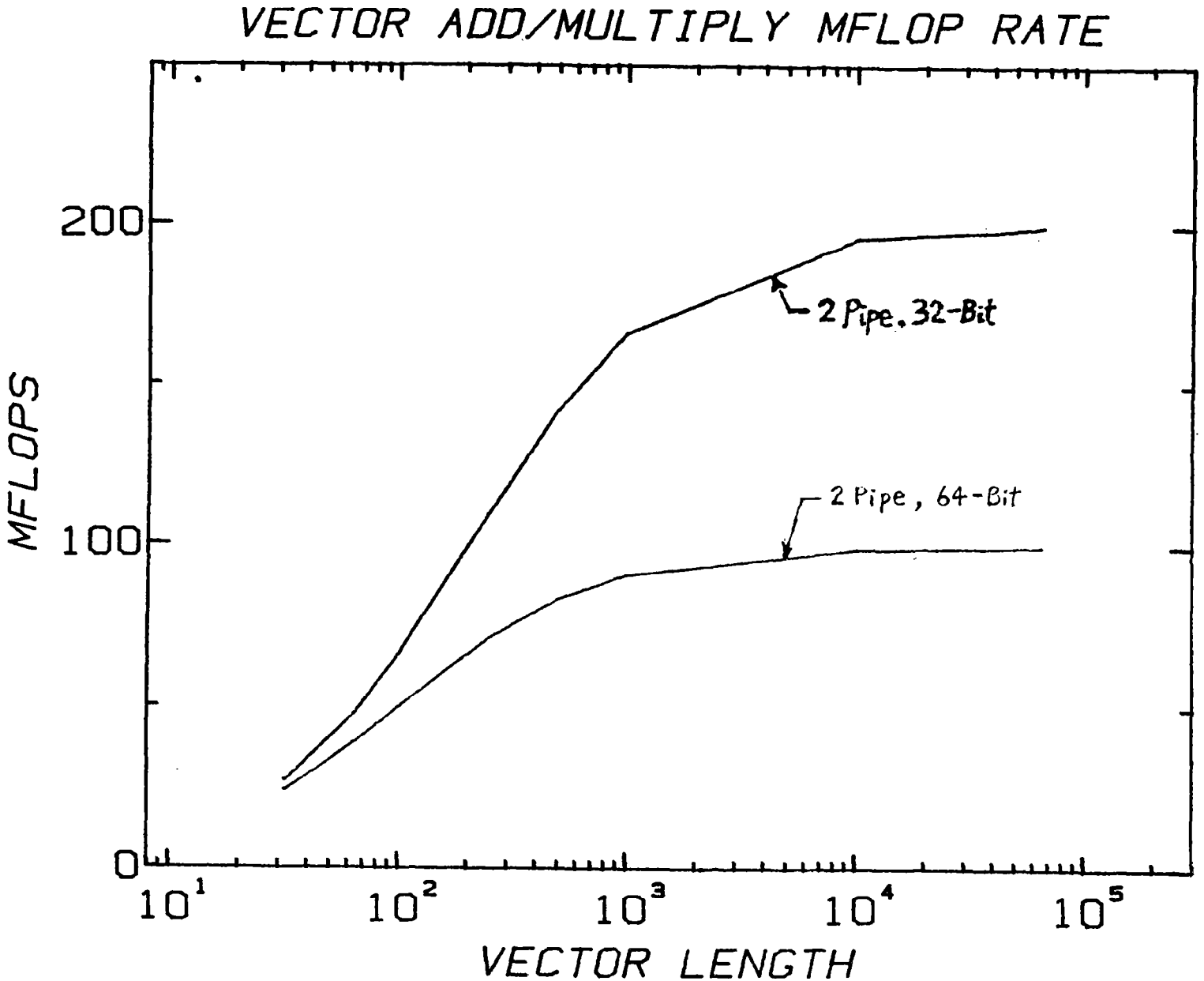
VECTOR ADD/MULTIPLY MFLOP RATE
200
100
0
MFLOPS
2 Pipe, 32-Bit
2 Pipe, 64-Bit
10^1
10^2
10^3
10^4
10^5
VECTOR LENGTH

Figure 1

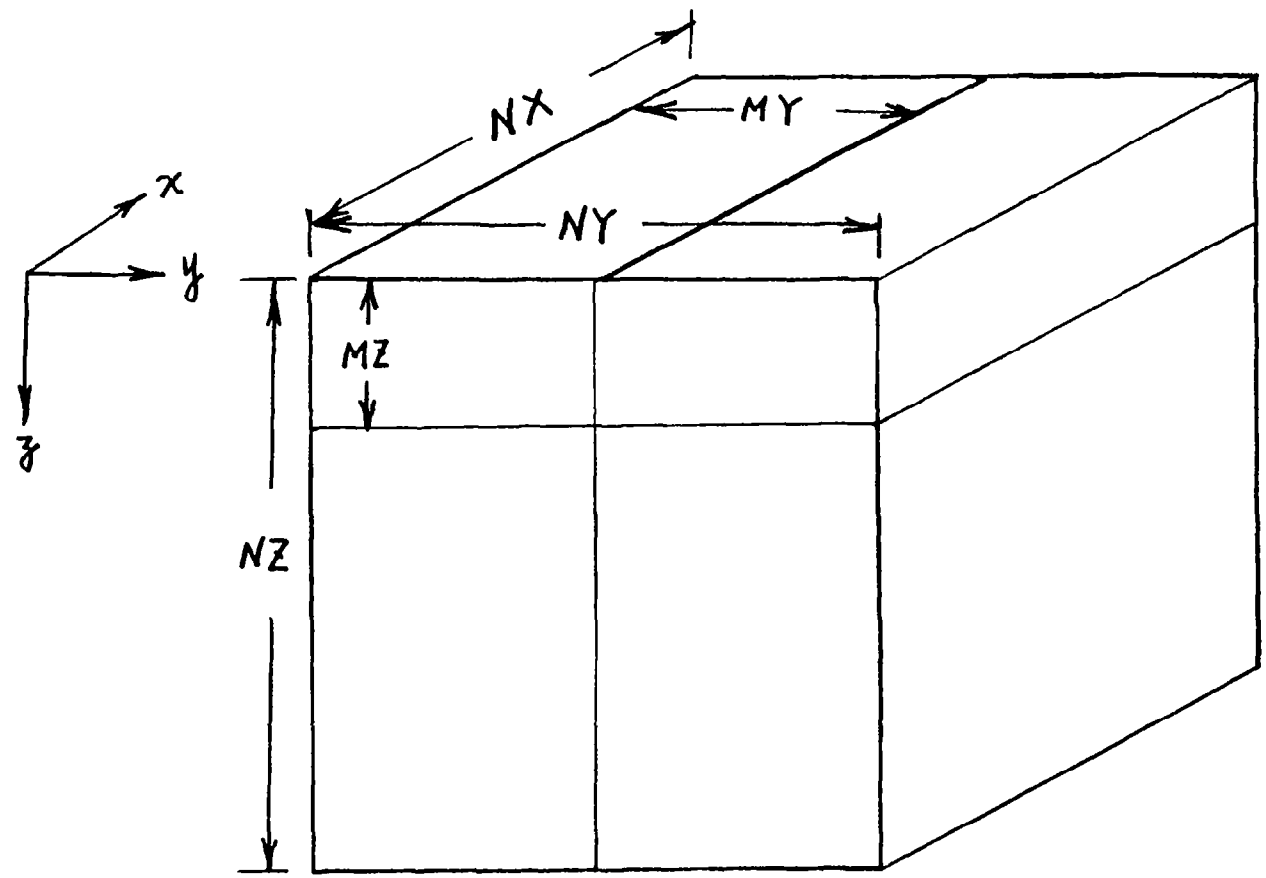

Figure 2

On the second sweep, MY x-z planes are processed. Fourier transforms in z and x directions are done on this sweep. The vector lengths are NX/2*MY*3 and NZ*MY*3 respectively.

A Cyber 205 vector is defined as a contiguous set of memory locations. Since the two sweeps are in different directions, an array transpose has to be done between sweeps and within the second sweep in order to keep processed data in a contiguous set of memory locations. The transpose is done by using gather instructions. The gather instruction puts array elements which are at various locations into a contiguous set of memory locations. An index vector is needed to pick up desired elements. Q8VGATHR function (64-bit) or Q8VXTOV subroutine (32-bit) is used to do the transposing. As the array gets bigger, so does the index vector length, and an appreciable amount of overhead working space is needed. In the 64^3 (32x16) run, the index vector has 17,408 elements.

6. COMPUTER PERFORMANCE

The performance data obtained to date, based on a hand count of the number of operations per time step, are presented in Table 1. The mesh size is given in column 2 (each node requires 7 words of data storage). The pencil size is given in column 3; this, together with mesh size, determines the vector length shown in column 4. The computational precision is given in column 5, the CPU time in column 6, and the CPU computation rate in column 7. The I/O time per step in seconds is meaningful only for runs with virtual memory paging. Explicit I/O would reduce I/O time considerably, but we have not yet attempted to use explicit I/O.

Figure 3 shows computation rate as function of vector length for our code on the 2-pipe CSU Cyber 205. It approaches an asymptote as vector length increases.

Comparing Runs 3 and 4, and Runs 5 and 6 in Table 1, it is found that the CPU time for a 32-bit (half) precision run is 60 percent of that for the corresponding 64-bit (full) precision run. We kept track of the timing in the transpose part of the code and found an interesting fact. In full precision runs, the transpose takes 15 percent of the CPU time; 85 percent of the CPU time is spent in floating point operations. In half precision runs, due to the lack of a half precision gather utility, the transpose takes the same amount of time as in full precision runs, while the floating point operations require only half of the full precision CPU time. Consequently, for half precision run, the transpose takes 25 percent of the total time.

Detailed timing from Run 8 shows that 51 percent of the CPU time is spent in the FFT subroutine, which contains 78 percent of the floating point operations. In other words, the FFT operates at 157.6 Mflops. The remaining 22 percent of the floating point operations are executed at 95 Mflops due mainly to shorter vector lengths and IF statements.

7. PROBLEMS ENCOUNTERED

Runs 7 and 8 of Table 1 require 3.5M words storage, and hence, do not fit within the 2M core memory at CSU with full precision. Thus, we must use 32-bit computation for efficient use of the CSU Cyber 205. Half-precision computation is sufficiently accurate for this code, and twice the operating speed is achieved.

TABLE 1.--PERFORMANCE OF CYBER 205 AT CSU
(2 PIPES WITH 2M 64-BIT WORD)

RUN	MESH SIZE (NODES)	PENCIL SIZE (NODES)	VECTOR LENGTH (IN FFT)	PREC. (BITS)	CPU TIME PER STEP (SEC.)	MFLOPS	I/O TIME PER STEP (SEC.)	MEMORY (M WORDS)	COMMENTS
1	8x8x8	8x8	192	64	0.014	23.5	-	0.02	in core
2	32x32x32	4x4	384	64	0.690	31.0	-	0.30	in core
3	32x32x32	32x32	3,072	64	0.399	53.6	-	0.69	in core
4	32x32x32	32x32	3,072	32	0.240	89.2	-	0.69	in core
5	64x64x64	16x16	3,072	64	3.378	59.6	56.6	2.70	paging
6	64x64x64	16x16	3,072	32	2.022	99.6	-	2.70	in core
7	64x64x64	32x16	4,608	32	1.980	101.7	-	3.47	in core
8	64x64x64	32x32	6,144	32	1.914	105.2	8.7	3.52	paging

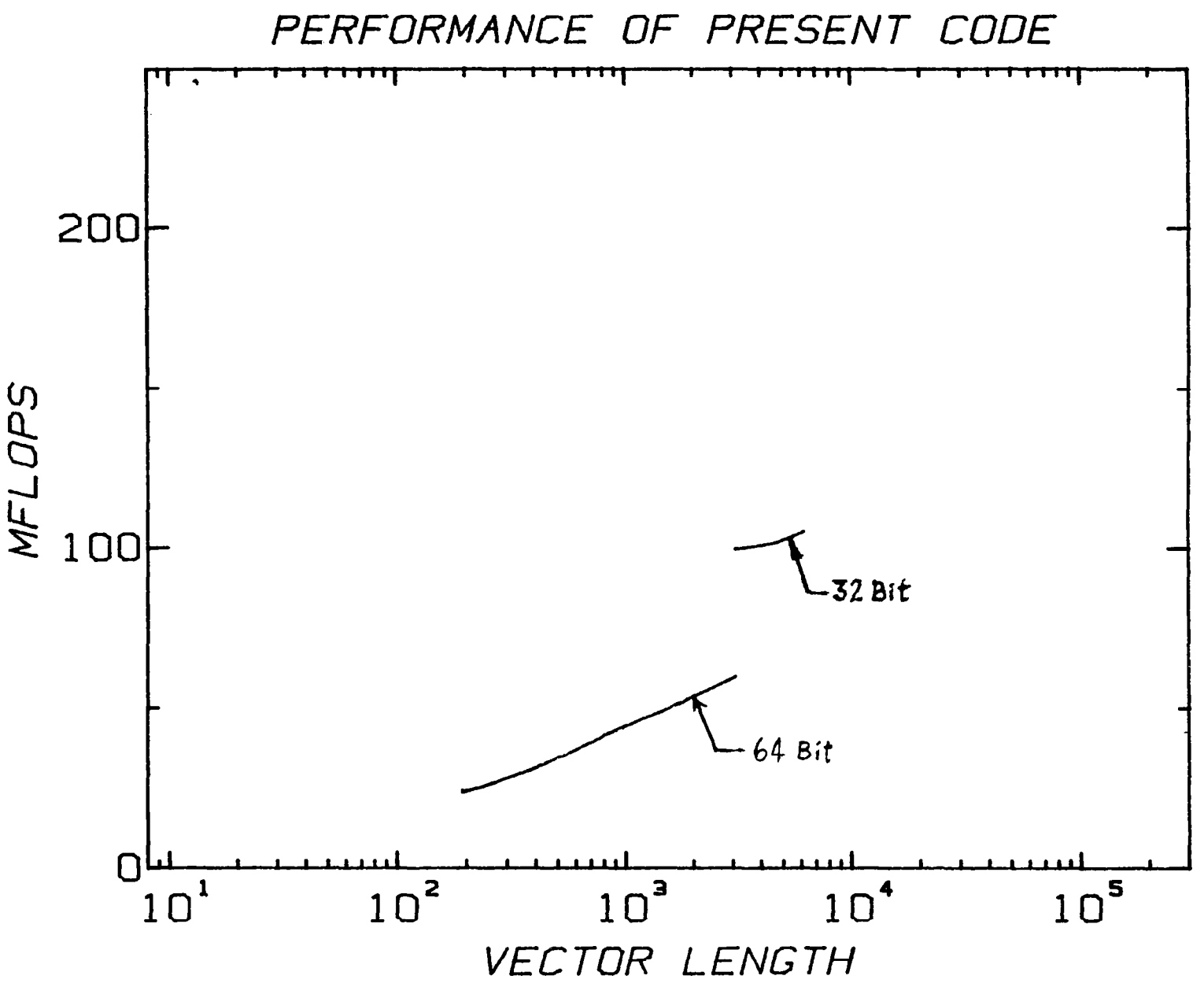
PERFORMANCE OF PRESENT CODE
200
100
0
MFLOPS
32 Bit
64 Bit
10^1
10^2
10^3
10^4
10^5
VECTOR LENGTH

Figure 3

Since there is no compiler available yet for half precision gather/scatter[4] calls, we have to use special Q8 calls[5] (machine instructions) to get the half precision code to compile properly on the CSU Cyber 205; the special Q8 instructions execute at full precision speed. Mr. Herbert Rothmund of CDC Sunnyvale was most helpful to us in providing these utilities.

It is apparent that the I/O rate is not balanced with the CPU time. The reason is that the CSU Cyber 205 has only two channels to transfer data between fast memory and disk and they are inherently slow. Solid-state backing memory (or equivalent) would speed up the data transfer rate. For our problem, faster I/O would allow us to go to 128^3 mesh size.

Since December 1982, three different compilers have been used: cycles 201109, L575, and 575B. Cycle 201109 did not have the half precision feature. Cycle L575 had half precision but lacked some automatic vectorization features. Cycle 575B, the most recent version, does not have gather/scatter in half precision. Further improvements are needed if users are to get optimum performance from this machine.

8. SIMULATION OF ISOTROPIC HOMOGENEOUS TURBULENCE

A typical simulation of homogeneous isotropic turbulent flow is presented in this section. Figure 4 shows the time history of the three-dimensional energy spectrum from initial time step to 300 time steps. Figure 5 shows the 3-D spectra of the components of the turbulent kinetic energy at time step 300. The flow is slightly anisotropic at low wavenumbers. This is due to the small number of modes at low wavenumbers.

All of these results are in excellent agreement with both experiments[6] and previous simulations.[7] Thus, we are confident that the code is performing satisfactorily and we will proceed to the simulation of compressed flows. The code presently runs at 1.9 second per time step for a 64^3 mesh on the 2-pipe Cyber 205; this compares with 5 seconds for the same type of code on the CRAY-1S in VECTORAL language.

9. CONCLUSION

In summary, we have written, debugged, and tested a code for solving the Navier-Stokes equations and for computing various turbulence statistical quantities. Most of the operations are readily vectorized, and 100 Mflops has been obtained for 64^3 mesh size in-core runs on a 2-pipe Cyber 205. The major problems encountered so far are concerned with the lack of compiler utilities, such as half-precision compiling capability for transpose operations.

The program works well and has been validated for homogeneous isotropic turbulence. The code will next be used to help develop turbulence models for compressed flow in engines.

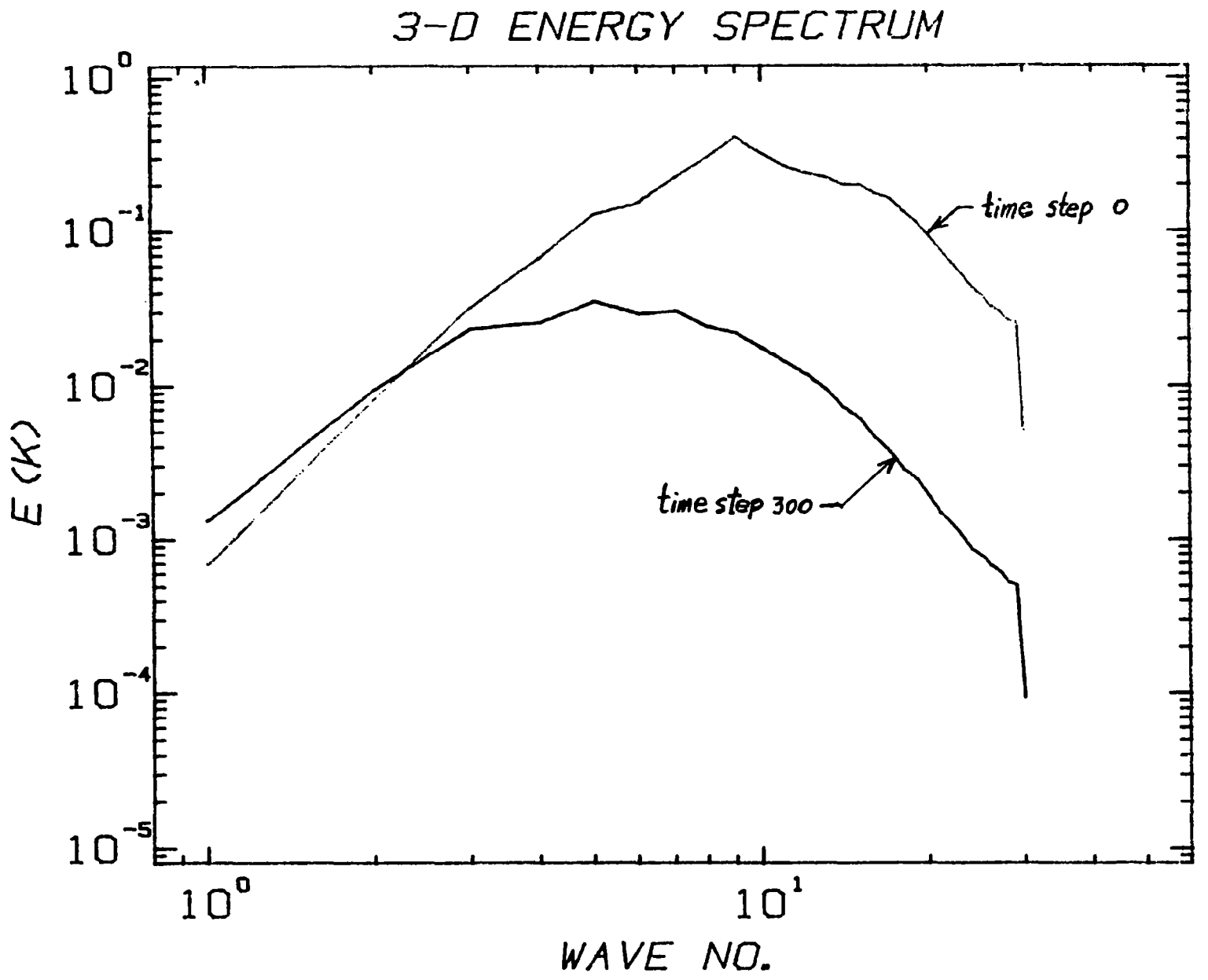
3-D ENERGY SPECTRUM
10^0
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
E (K)
time step 0
time step 300
10^0
10^1
WAVE NO.

Figure 4

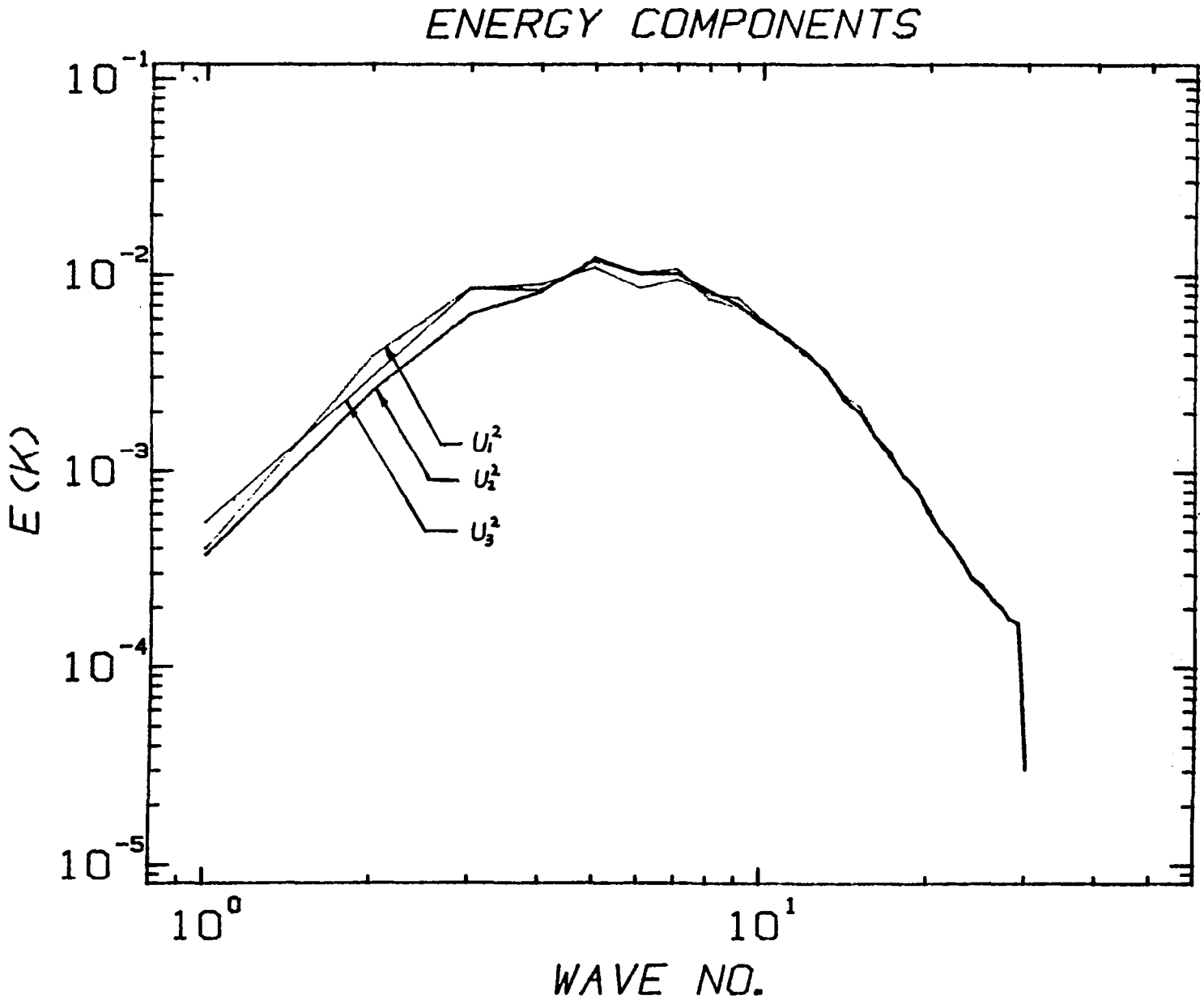
ENERGY COMPONENTS
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
E(K)
U_1^2
U_2^2
U_3^2
10^0
10^1
WAVE NO.

Figure 5

ACKNOWLEDGMENTS

This work was supported by Control Data Corp. under project No. 82S302. The authors would like to thank Mr. Herb Rothmund of CDC Sunnyvale, Mr. Art Lazanoff, Ms. Jeanne Adams and liaison staff of Colorado State University, and Dr. Nagi Mansour for their important contributions to this work.

NOMENCLATURE

B_{ij}	Coordinate transformation tensor
MY	Pencil size in Y-direction
MZ	Pencil size in Z-direction
NX	Number of mesh points in X-direction
NY	Number of mesh points in Y-direction
NZ	Number of mesh points in Z-direction
p'	Pressure fluctuations
t	Time
$\overline{U}_{i,j}$	Mean strain rate tensor
u_i	Velocity fluctuations in i-direction
x	Space coordinate
y	Space coordinate
z	Space coordinate
δ_{ij}	Kronecker delta
ν	Kinematic viscosity

REFERENCE

1 Rogallo, R. S., "Numerical Experiments in Homogeneous Turbulence," NASA TM81315, September 1981.

2 "Guide to Vector Processing Services," CSU Computer Center and the Institute for Computational Studies, October 1982.

3 Kascic, M. J., Jr., "Vector Processing on the Cyber 200," Workshop I, CSU, December 1982.

4 "CDC Cyber 200 Fortran Version 2 Reference Manual," Control Data Corporation, November 1981.

5 "CDC Cyber 200 Model 205 Computer System Hardware Reference Manual," Control Data Corporation, March 1981.

6 Comte-Bellot, G., and Corrsin, S., "Simple Eulerian Time Correlation of Full- and Narrow-Band Velocity Signals in Grid-Generated 'Isotropic' Turbulence," J. Fluid Mech. (1971), Vol. 48, part 2, pp. 273-337.

7 Shirani, E., Ferziger, J. H., and Reynolds, W. C., "Mixing of a Passive Scalar in Isotropic and Sheared Homogeneous Turbulence," Rept. No. TF-15, Thermosciences Division, Department of Mechanical Engineering, Stanford University, Stanford, Calif. May 1981.