Lending Copy

DARESBURY LABORATORY

INFORMATION QUARTERLY for MD & MC SIMULATIONS

An Informal Newsletter associated with Collaborative Computational Project No. 5
on Molecular Dynamics and Monte Carlo Simulations of Macroscopic systems.
Number 7 December 1982

Contents

Editor: Dr. William Smith

Science & Engineering Research Council,
Daresbury Laboratory, Daresbury,
Warrington WA4 4AD, England

Deputy Editor: Dr. David Heyes

Royal Holloway College,
University of London, Egham TW20 OEX
England

Editorial

We begin the newsletter, as usual, by thanking all those who have contributed to its production. David Heyes (Royal Holloway College) and David Fincham (Queen Mary College) have jointly contributed articles on shear viscosity calculations and the speed performance of various computers in molecular dynamics calculations. W. Smith (Daresbury) has contributed articles on the CCP5 Program Library and on the use of FFT methods in calculating correlation functions. Julian Clarke (UMIST) has contributed a review of the work currently in progress at UMIST. Lastly, Stuart Pawley has provided a short announcement on future computing trends using the ICL DAP. All are thanked for their cooperation and effort.

Once again we remind our readers that we are pleased to include articles of any length or source that may be of interest to the CCP5 community. The past issues bear testimony to the diversity of subjects we are willing to accept. Please bear the newsletter in mind for any communications you have in the future.

Contributors:

D. Heyes, Department of Chemistry, Royal Holloway College, Egham, Surrey TW20 OEX.

D. Fincham, DAP Support Unit, The Computer Centre, Queen Mary College, Mile End Road, London E1 4NS.

W. Smith, TCS Division, SERC Daresbury Laboratory, Daresbury, Warrington WA4 4AD.

J. Clarke, Department of Chemistry, UMIST, Sackville Street, Manchester M60 1QD.

G.S. Pawley, Department of Physics, University of Edinburgh, Mayfield Road, Edinburgh EH9 3JZ.

General News

1. The most important item of news concerns the renewal of CCP5 after its expiry on 30th September, 1983. If the project is to continue for a further three years it is necessary for the working group to submit a renewal proposal in the Spring of 1983. To assist with the collation of material for the proposal document a questionnaire is being prepared for circulation to all CCP5 participants. It is essential that everyone associated with CCP5 provides this information for the renewal proposal. Please do your best to comply. The opportunity is also available for participants to put forward new ideas for CCP5 and to express their opinions on the current organisation.

2. The eighth CCP5 conference on 'The Computer Simulation of Molecular Liquids and Liquid Mixtures' is to be held at Hull University on 29th/30th March, 1983. This follows on directly after the conference on 'Liquids and Liquid Mixtures' organised at Hull by the Statistical Mechanics and Thermodynamics Group of the Royal Society of Chemistry. Anyone wishing to contribute a talk at the CCP5 meeting should submit an abstract before 15th February, 1983, to Professor J.G. Powles, Physics Laboratory, The University of Kent, Canterbury, Kent CT2 7NR, U.K.

3. G.S. Pawley has provided the following item of news:

Desk-top number-cruncher

The future of the ICL DAP (Distributed Array Processor) now seems to be much clearer as ICL are developing a DAP to be used as a desk-top computer in conjunction with the PERQ. The PERQ has already been chosen for promotion by SERC as a desk-top computer in its own right, and to this a DAP will be added at a comparable cost. This DAP will be 32 × 32 processors, each of 16 K bits, as compared with the big DAP's 64 × 64 processors of 4 K bits each. The performance will therefore still be a considerable fraction of a CRAY-1, and a user does not have to share with 100 others.

Official statements are naturally hard to come by, but with luck these systems will be available early 1984. As part of our agreement with ICL whereby we have obtained a big DAP in Edinburgh, we have undertaken to provide some of the software for the system and so we should be in a good position to watch progress and help interested colleagues.

4. Further additions have been made to the CCP5 Program Library. D. Heyes has donated the programs HLJ5 and HLJ6 which are both atomic MD shifted force programs, the former using the leap-frog algorithm the latter a more accurate algorithm due to Toxvaerd. W. Smith has provided the density correlation programs DENCOR and CURDEN and with D. Fincham, the MD program MDMPOL which simulates molecules (or mixtures of molecules) with fractional charges. B. Jonsson (LUND, Chemical Centre) has donated the MC program CARLOS, which simulates a solute molecule in an aqueous cluster and the corresponding analysis program CARLAN.

Readers interested in obtaining copies of the programs, or any others listed in the attached program catalogue should contact the program librarian, Dr. W. Smith, SERC, Daresbury Laboratory, Daresbury, Warrington WA4 4AD, U.K. Please send a magnetic tape with your request. There are no charges associated with the CCP5 Program Library, however subscribers are requested to acknowledge the use of programs in publications. Readers wishing to contribute programs should also contact W. Smith.

5. Dr. N. Quirke has pointed out the following typographical errors in his article entitled 'Metropolis Monte Carlo Computer Simulation', which appeared in the September issue of this newsletter. All the errors concerned appear in the FORTRAN program therein:

(i) Lines 28/29 should read:

```
 EST = 0.5*(ABS(HT-H1) + ABS(HT-H2) + ABS(HT-H3)
X     + ABS(HT - H4))
```

(ii) The WRITE statement appearing at the end of the program (before STOP) should read:

```
WRITE(6,x)AVE,SS,NYES
```

(iii) The variables H and E are assumed to be initialised to zero at the start of the program.

CCP5 Program Library: Program Catalogue

Program	Author	Purpose
MDATOM	S. Thompson	MD on monatomic liquids
HMDIAT	"	MD on homonuclear diatomics
MDLIN	"	MD on linear molecules
MDLINQ	"	As MDLIN but with point quadrupole
MDTETRA	"	MD on tetrahedral molecules
MDPOLY	"	MD on polyatomic molecules.
MDATOM	D. Fincham	MD on monatomic liquids
MDDIAT	"	MD on homonuclear diatomics
MDDIATQ	"	As MDDIAT but with point quadrupole
MDIONS	D. Fincham and N. Anastasiou	MD on ionic liquids
HLJ1	D. Heyes	MD on monatomic liquids
HLJ2	"	As HLJ1, with velocity autocorrelation
HLJ3	"	As HLJ1, with link cells
HLJ4	"	As HLJ1, with constant P or T
HLJ5	"	MD on monoatomic liquids (shifted force)
HLJ6	"	As HLJ5 with Toxvaerd algorithm
MCRPM	"	MC dilute, charged, hard spheres
HSTOCH	W.F. Van Gunsteren and D. Heyes	Stochastic dynamics program
EWALD1	W. Smith	Subroutines to calculate potential, forces
EWALD2	"	or torques in multipole system.
MDMIXT	"	MD for molecular mixtures
MDMULP	"	MD polyatomic molecules with point multipole
DENCOR	"	Density correlation in monatomic liquids
CURDEN	"	Current density correlation in monatomic liquids
MDMPOL	W. Smith and D. Fincham	MD polyatomic molecules with fractional charges
MCN	N. Corbin	MC (Metropolis) simulation of monatomic liquids
SCN	"	MC (Rossky,Friedman, Doll) of monatomic liquids
SMF	"	MC (Path integral method) for monatomic liquids
CARLOS }	B. Jonsson and	MC of solute molecule in aqueous cluster
CARLAN	S. Romano	Analysis program for CARLOS

The CCP5 Program Library - An Overview

W. Smith

Since the creation of CCP5 early in 1980 a significant effort has gone into the writing and documentation of computer programs for general use by the CCP5 community. As a result of this effort (principally by S. Thompson, D. Heyes. D. Fincham and myself - together with the advice and guidance of Professor K. Singer) there now exists a fairly substantial library of about two dozen programs covering a fairly wide area of application in liquid simulation work. The emergence and availability of each new program and a list of all programs available is announced in the 'General News' section of the 'Information Quarterly' newsletter and readers interested in the programs are invited to obtain copies free of charge.

There is clearly a need for such a program library. A lot of unnecessary time, effort and expense is spared when one obtains a ready-made program that allows immediate simulation of a system of interest. Certainly, a small but significant number of people have been very excited to discover the library's existence and have made grateful use of it. However despite this kernel of interest the overall impression is that the program library has yet to take off and there exists a large pool of potential users who fail to take advantage of the programs available. As the librarian in charge of the program library this under-useage concerns me greatly and I wish here to highlight some of the suspected reasons for this condition as well as to invite comments and suggestions from the newsletter readers on the current and future requirements of the library.

To begin with, it would be instructive to present the purpose for maintaining the program library, since there may be some facilities that users have overlooked but would benefit from. (Equally it will show what the librarian has overlooked - but I am always willing to listen to ideas!).

Firstly, the prime function of the library is to provide users with good and proven software (and documentation!) for all MD and MC simulators. The software provided should be 'good' in a number of senses; the programs should be efficient in terms of computer memory usage and execution speed; they should be versatile in that they may be easily extended or modified for use in other subject areas without undue difficulty and they should function reasonably well on computer systems other than that for which they were written (failing that, variants should be available).

Secondly the organiser of the library should make efforts to extend the range of programs available either by writing or acquiring new programs and documentation. This exercise should be backed up with efforts to enhance 'good' qualities as described above.

Thirdly, the program library should interact with the CCP5 community over a wide area. This is a complicated matter; involving such diverse aspects as ascertaining the programming needs of the community, acquiring programs, obtaining commissions for programs, assisting with the adaptation of programs to other systems, de-bugging, publicising available programs and 'pooling' programming expertise so that other users may get help from it. No doubt the reader can think of other aspects of this interaction, but the main point is that the library should not be regarded as a passive entity.

These then, are the ideals of the program library. Hopefully, users who have made use of the library will recognise these principles, at least in germinal form. In general, however, the library probably falls short of these ideals, though I believe matters can be put right quite easily if there exists the will to do so.

On the subject of the provision of programs the library fails principally in that it is under-used. This is probably because it does not yet provide a sufficiently comprehensive range of programs. Clearly, with time, matters should improve but it should be recognised that CCP5 covers a wide area of research interests and there will always be areas not catered for. However, the main stumbling block is likely to be the fact that MD and MC programs are often relatively easy to write and users are often very happy to produce their own, despite the fact that testing and de-bugging even simple programs can be an expensive and exhausting business. Also one suspects that there already exists a significant informal traffic in programs between colleagues, which despite being satisfying to both parties, does have the effect of denying a third party the same experience. The antidote to both these problems is for the parties concerned to look to the CCP5 program library as the ideal source and repository of their programs. Thus with a little effort expended in this way a much more useful general service can be built up.

With regard to the development of programs for CCP5 the library has probably come closest to its ideal. Most of the programs now available have been written specifically for the program library and offer a reasonable range of facilities. For example, there exist programs that can simulate atomic and molecular liquids, liquids containing charged species, fractional charges or point multipoles and also, molecular mixtures. There are also some data processing programs to calculate density and current density correlation functions. It is a fair criticism, however, that the current software is too biased towards the CRAY-1 computer. Thus there is a larger proportion of MD programs (which are vectorisable) over MC codes (which are not so easily vectorised). This, however, should not deter potential users, as most of the programs are well documented and relatively easy to adapt to other systems (many of the documents contain advice on this). Equally, conversion of MD programs to MC programs should not be beyond most users as an interim measure. Nevertheless, these criticisms are recognised and efforts should be made to redress the balance in the future.

The business of collecting additional programs is clearly an area where a great deal is possible. Probably every laboratory has one or more programs that would be of use to the wider community. Many of them undoubtedly have seen little use since they were relegated to a cardboard box beneath the laboratory sink, but if they were once, or are still, useful programs and a modicum of documentation exists, there is always the chance that someone would find a use for them if their existence were known. This is not to say, however, that any old program is acceptable; the essential criterion should be that the programs were once used successfully in research. Other criteria, such as the availability of documentation, plus the qualities of 'goodness' outlined earlier, are also important. If the CCP5 community were to contribute such programs, there is no doubt that much benefit would accrue. Happily there are signs that some active programmers involved with CCP5 recognise this and have contributed useful programs. Without more such help, however, the growth of the library could be a slow process.

Lastly, on the subject of the interaction between the Program Library and the CCP5 community, there is much that can be done by the community itself. Principally through the "Information Quarterly" newsletter, the services of the CCP5 Program Library have been made known and there already exists the means for achieving the fullest interaction as outlined earlier. All that is required is for the community to recognise this and make use of it. This not only applies to those actively involved in CCP5 itself, but to anyone interested in simulation work.

A Review of Molecular Dynamics Simulation Work at U.M.I.S.T.

by J.H.R.Clarke

RHEOLOGICAL PROPERTIES OF MODEL MOLECULAR LIQUIDS.
(with Dr. M.Whittle and D.Brown)

This work was started using an early version of MDDIAT (kindly supplied by David Fincham) for homonuclear diatomics. Programs have now been developed for systems of molecules with up to six LJ centres (utilising either quaternions or a constraints algorithm) and also for systems employing the single centre 'ellipsoidal' interaction potential of Kushik and Berne. These models can be examined with or without homogeneous shear boundary conditions. Viscosities are determined from the computed steady state stress response to a constant applied strain rate. The form of stress (or related) correlation functions can be determined for short times (up to about 1 ps) using either the perturbation technique of Jacucci and Cicotti or by the Green Kubo method of analysing equilibrium fluctuations; for a range of systems and with due attention to statistics both methods give identical results (see also CCP5 Newsletter June 1982). The (off diagonal) stress correlation function, C(t), shows a characteristic decay in this time region, the form of which is strongly dependent on the molecular shape. As a result of statistical limitations and systematic errors, however, neither of these latter techniques is reliable for determining C(t) at times greater than about 1 ps. Since it is well known that these molecular liquids contain important long time relaxations, the methods are as yet not suitable for determining the shear viscosity which is related to the integral over C(t).

We have been able to make significant advances in identifying the role played by reorientational molecular motions in determining the shear viscosity of simple molecular liquids. Using a series expansion in suitable angular coordinates we have shown explicitly how the relative orientations of molecule pairs in a diatomic liquid contribute to the instantaneous shear stress. Stress correlation functions and steady state shear viscosities obtained using the expansion up to second order compare very favourably with exact MD results for anisotropic molecules such as nitrogen at normal liquid densities (1). This work represents an important use of computer simulation to evaluate and develop a molecular theory.

Using the Cray-1 we are currently studying the non-Newtonian behaviour of model liquid alkanes up to hexane. A number of interesting results have emerged from these studies including the effects of the coupling of intramolecular motions with intermolecular stress relaxation. These will be discussed in forthcoming publications (2) and we give only a short summary here. These projects would not have been feasible, however, without the facilities provided by the S.E.R.C. on the Cray computer.

The decrease in viscosity that is observed with increasing shear rate (shear thinning) turns out to be remarkably similar to that of real and apparently more complex liquids, provided that the strain rate is always scaled using the Maxwell viscoelastic relaxation time. Several proposed theories (Ree-Eyring, Kawasaki-Gunton, Hess, etc.) predict forms for this variation of viscosity, as well as that of molecular alignment, pressure, energy etc. as a function of shear rate in molecular liquids. However the experimental data with which to test these theories in a systematic way is very limited and will be extremely difficult to supplement. Comparison of theoretical predictions with exact simulation results on well defined model systems is therefore of some importance.

SHORT RANGE EFFECTIVE POTENTIALS FOR IONIC LIQUIDS.
(with S.M.Hayne)

For many years it has been thought that the long range charge ordering in ionic liquids (as observed in computer simulation or by experimental diffraction techniques) is a direct result of the long range of coulomb interactions. We have shown that both the short range structure and the long range charge ordering of simple ionic liquids such as KCl can be reproduced quite accurately using simple pairwise additive interactions with a range of no more than three ion diameters. The use of such short range effective potentials was first suggested by Woodcock. This approach permits the economical use of very large samples in molecular dynamics computer simulation (as many as 10 000 particles) with link cell coding to achieve fast computation of the forces. A perturbation approach towards the general properties of ionic liquids is being developed using such reference systems.

Substantial effort has been expended by many workers in implimenting the Ewald and related summation methods for evaluating the long range correction to the forces and energies based on effective ionic pair potentials in small periodic systems (there is a practical limit here of about 500-1000 particles). For highly accurate simulations such methods should continue to be used. However there is still some uncertainty regarding the best form for these ionic potentials. Also it must be born in mind that the experimental values of properties for comparison often are only known with relatively low precision. Furthermore many important problems remain (e.g. concerned with interfacial phenomena) which demand the use of very large samples for realistic simulation and for which useful answers can be obtained using these (approximate) short range potentials. A description of this work will appear in forthcoming articles (3).

GLASS TRANSITION PHENOMENA IN MODEL MIXTURES.
(with L.Copeland)

Following our work on the single component Lennard-Jones (LJ) liquid (4,5,6) we have extended studies to LJ monatomic mixtures. The interest here is to understand the importance of the relative interaction potentials between component pairs on the structural, thermodynamic and transport properties at high densities. The relevance of the work is that it provides models for glass formation in metal alloys which have potentially important practical and commercial applications. We have shown that it is the relative sizes of the constituent atoms which is of primary importance in determining the above properties. The relative intensity of particle interactions plays a secondary role although of course it does affect the values of some of the thermodynamic properties. The glass transition in these systems is thus shown to be primarily a packing phenomenon. Some particularly interesting effects are observed when the size ratio of the component particles is about Ø.4, at which 'interstitial packing' of the smaller particles in the liquid lattice of larger particles becomes significant. For an equimolar mixture, for instance, the component diffusion coefficients show quite different behaviour as a function of density, an effect which tends to 'smear out' the glass transformation range (7).

References

1. M.Whittle and J.H.R.Clarke, submitted for publication
2. D.Brown and J.H.R.Clarke, article in preparation
3. L.V.Woodcock, F. van Swol, S.M.Hayne and J.H.R.Clarke, article in preparation
4. C.A.Angell, J.H.R.Clarke and L.V.Woodcock, Adv. Chem. Physics, 1981,48,397
5. J.H.R.Clarke, J.C.S. Faraday 11, 1979,75,1371
6. J.H.R.Clarke, J.F.Maguire and L.V.Woodcock, Faraday Discussions of the Chemical Society,198Ø,69,273
7. L.Copeland and J.H.R.Clarke, to be published

Correlation Functions and the Fast Fourier Transform

W. Smith

In a previous issue of the CCP5 newsletter[1] I indicated how the fast Fourier transform (FFT) may be used to improve the efficiency of calculating correlation functions. This present note is intended to provide a demonstration of the capability of the FFT in this application.

To recap the previous article; the discrete correlation function c(k) calculated from the discrete functions u(k) and h(k) is obtained conventionally via a 'sum over origins':

$$c(k) = \frac{1}{(N-k)} \sum_{n=0}^{N-k-1} u^*(n)h(n+k) \qquad (1)$$

where: k=0,...,N-1

This is a straightforward procedure which requires of the order N^2 floating point operations to compute the correlation function of array length N.

The FFT method requires that the discrete functions u(k) and h(k) be Fourier transformed and the correlation function calculated as an array product in the frequency domain. Subsequent Fourier inversion of this product yields the desired correlation function. This procedure may be summarised by the equation:

$$c(k) = \frac{1}{(N-k)(2N)} \sum_{j=0}^{2N-1} U^*(j)H(j)\exp(i2\Pi jk/(2N)) \qquad (2)$$

where: k=0,...,N-1

It should be noted that the FFT method requires the arrays u(k) and h(k) be doubled in length (by appending N zero elements) to prevent spurious correlations. Even so, by this route, the number of floating point operations is more of the order (2N) $\log_2$ (2N) and should produce the correlation functions much more efficiently once N becomes large enough.

To compare these two methods I have calculated a typical density auto-correlation function on the CRAY-1 computer by the two methods outlined above using realistic data. The initial data array was of length 32768 (complex) elements. The two programs used in the comparison are presented in Listing 1 (conventional method) and Listing 2 (FFT method). The programs print out the CPU time elapsed at comparable stages of the calculation. These times are presented in Table 1. The correlation functions calculated are presented in Table 2 (conventional method) and Table 3 (FFT method).

Before discussing the results, some comments on the two programs used are in order. Firstly, it should be noted that the conventional method (Listing 1), in calculating the correlation function, does not use every data element as a new 'origin', but instead uses every fourth element. Secondly, the correlation is only evaluated out to an element separation of 511, giving a discrete correlation function 512 elements in length. This might be called 'the reduced conventional method'. Such alterations are quite usual in the conventional method and greatly reduce the computational effort. (It is quite easy to show that the number of floating point operations required drops from approximately $8N^2$, where N is the number of data elements; to approximately $8N*N_t/N_g$ where N_t(<<N) is the chosen length of the correlation function and N_g is the gap between the origins. In this application the computational effort is thus reduced by a factor approaching 256). The FFT method does not reduce the volume of data processing and calculates a full correlation function 32768 elements in length. For convenience however only 512 elements are considered for comparison.

Comparison of the correlation functions produced by the two methods (Table 2 and Table 3) reveals that they are in close agreement. The normalised (i.e. c(0) = 1.0) correlation functions differ only in the fifth decimal place for the 512 elements shown. The observed differences may be due to one or both of two factors. Firstly, the reduced amount of data processed by the conventional method may be expected to give slightly different results from a full evaluation (even so the good agreement is impressive). Secondly it is possible that the FFT method needs 'fine-tuning' with an appropriate window function to produce exact results.

The most impressive aspect of the FFT method is its speed. Table 1 shows the CUP time elapsed at various stages of the calculation. The conventional method performs the correlation calculation in 3.91544 seconds. This, however, represents only 1/256 approximately, of the full correlation produced by the FFT method. Thus a more comparable figure would be 1000 seconds approximately. The FFT method requires 14.98147 seconds to calculate the full correlation. This indicates that in the full application, the FFT method may be expected to be about 67 times faster than the conventional method. In this respect the FFT method is greatly superior to the conventional method.

A further matter of importance, in view of the availability of the CRAY-1 array processor, is the vectorisability of the two techniques. In its most basic form the conventional method is not vectorisable because of its use of irregular array addressing[2] within the central DO-LOOP of the correlation function calculation. This must be seen as a great disadvantage, because if the non-vectorised FFT routine BFFT2 is replaced by the vectorised, CRAY-specific routine CFFT2[3] a further astonishing improvement in speed is observed (Table 1). The correlation now requires only 0.19704 seconds for a full correlation function. It is thus in the region of 1340 times faster than the conventional method! Even if the reduced conventional method is used the FFT method is still 20 times faster.

The superiority of the FFT method is thus obvious in this application, but is it always so? The relative merits of the two methods hinge on the desirability of producing the fully comprehensive correlation function. It is doubtful that such a comprehensive function will always be required, though when it is, the FFT method is the obvious choice. In other circumstances, however, the conventional method can have distinct advantages.

To begin with, consider the case where more than one correlation function is to be calculated. It is interesting to note that while the conventional method is not intrinsically vectorisable, it can be made so in this circumstance by replacing the principal correlation loop:

```
   DO 40 L=1,LOR
   M=IND(L)
   IND(L)=M+1
   NUM(M)=NUM(M)+1
   CORT(M)=CORT(M)+G(N)*CORO(L)
40 CONTINUE
```

with the nested loop:-

```
   DO 40 L=1,LOR
   M=IND(L)
   IND(L)=M+1
   NUM(M)=NUM(M)+1
   DO 40 K=1,NCOR
   CORT(M,K)=CORT(M,K)+G(N,K)*CORO(L,K)
40 CONTINUE
```

In which the central loop, now vectorised, calculates the Mth element for a set of NCOR correlation functions. By this method it is possible for the conventional method to improve its performance against the FFT method, which must necessarily process only one (entire) correlation function at a time. If the reduced form of the conventional method is used (and the results of this appear to be adequate for most purposes) it is possible for the method to compete satisfactorily with the FFT method.

Secondly, the conventional method has the advantage over the FFT method in terms of computer memory requirement. The FFT method is most efficient when it is processing the maximum amount of data and thus is a prodigious user of computer memory. This coupled with the fact that FFT algorithms generally require the processed arrays to be of length 2^N, means that a given application may quickly exceed the available memory. There are of course strategies to overcome this difficulty, such as using overlapping Fourier transforms to produce shorter correlation functions (see for example the program given in Listing 3) but it would appear that the computed correlation function still needs to be quite long before it becomes preferable to the reduced conventional method on a normal computer.

A further point worth making is that the FFT method requires all the data to be processed to be present in the computer memory before processing can begin. This means that the program may have to spend some time waiting for this data to be read into the memory, unless more sophisticated I/O procedures are adopted. The conventional method does not suffer this drawback; processing can begin at once without being seriously delayed by the data input requirement.

In conclusion then, it would appear that the FFT method is vastly superior to the conventional method when a comprehensive correlation function is required. However the conventional method does appear to be quite adaptable and by chosing to calculate shorter correlation functions (via the reduced method) there are circumstances where the conventional method seems preferable.

TABLE 1

Accumulative CPU Time at Various Stages of Correlation Function Calculation

Time after end of:-	Conventional Method	FFT Method	Vectorised FFT Method	Vectorised Overlapping FFT Method
START	0.10691	0.32184	0.27056	0.14824
READ	0.10925	0.32430	0.27291	0.15063
INITIALISATION	0.10937	0.57947	0.29983	0.15198
CORRELATION	4.02481	15.56094	0.49687	0.63491
NORMALISATION	4.02497	15.56145	0.49738	0.63542
PRINT	4.04693	15.58345	0.51932	0.65744

LISTING 1.   Evaluation of Correlation Function by Conventional Method.

```
      COMPLEX G(32768),CORT(512),CORO(512)
      DIMENSION IND(512),NUM(512)
      NGAP=4
      NORG=128
      WRITE(6,10)
10    FORMAT(1H1)
      ITIM=0
      CALL SECOND(TIME)
      WRITE(6,20)ITIM,TIME
20    FORMAT(1H0,'TIME AT STAGE',I3,' = ',F20.10)
C***********************************************************************
      ITIM=1
      READ(7)G
      CALL SECOND(TIME)
      WRITE(6,20)ITIM,TIME
C***********************************************************************
      ITIM=2
      LOR=0
      MOR=0
      DO 25 L=1,512
      NUM(L)=0
      CORT(L)=(0.0,0.0)
25    CONTINUE
      CALL SECOND(TIME)
      WRITE(6,20)ITIM,TIME
C***********************************************************************
      ITIM=3
      DO 50 N=1,32768
      IF(MOD(N-1,NGAP).GT.0)GO TO 30
      LOR=MIN0(LOR+1,NORG)
      MOR=MOD(MOR,NORG)+1
      IND(MOR)=1
      CORO(MOR)=CONJG(G(N))
30    DO 40 L=1,LOR
      M=IND(L)
      IND(L)=M+1
      NUM(M)=NUM(M)+1
      CORT(M)=CORT(M)+G(N)*CORO(L)
40    CONTINUE
50    CONTINUE
      CALL SECOND(TIME)
      WRITE(6,20)ITIM,TIME
C***********************************************************************
      ITIM=4
      RNORM=FLOAT(NUM(1))/REAL(CORT(1))
      DO 60 L=1,512
60    CORT(L)=RNORM*CORT(L)/FLOAT(NUM(L))
      CALL SECOND(TIME)
      WRITE(6,20)ITIM,TIME
C***********************************************************************
      ITIM=5
      DO 190 I=1,128,2
190   WRITE(6,200)(I+J-2,CORT(I+J-1),J=1,512,128)
200   FORMAT(4(1H ,I5,2F13.5))
      CALL SECOND(TIME)
      WRITE(6,20)ITIM,TIME
      STOP
      END
```

LISTING 2.   Evaluation of Correlation Function by FFT Method.

```
      COMPLEX G(33000),G2(32536),W(163840),CORT(512)
      WRITE(6,10)
   10 FORMAT(1H1)
      ITIM=0
      CALL SECOND(TIME)
      WRITE(6,20)ITIM,TIME
   20 FORMAT(1H0,'TIME AT STAGE',I3,' = ',F20.10)
C**********************************************************************
      ITIM=1
      READ(7)G
      CALL SECOND(TIME)
      WRITE(6,20)ITIM,TIME
C**********************************************************************
      ITIM=2
      IND=1
      ISC=-1
      CALL BFFT2(IND,ISC,65536,G,W,G)
      IND=0
      DO 26 L=32769,65536
      G(L)=(0.0,0.0)
   26 CONTINUE
      CALL SECOND(TIME)
      WRITE(6,20)ITIM,TIME
C**********************************************************************
      ITIM=3
      CALL BFFT2(IND,ISG,65536,G,W,G)
      DO 50 L=1,65536
   50 G(L)=G(L)*CONJG(G(L))
      CALL BFFT2(IND,ISG,65536,G,W,G)
      CALL SECOND(TIME)
      WRITE(6,20)ITIM,TIME
C**********************************************************************
      ITIM=4
      RNORM=32768.0/REAL(G(1))
      DO 60 L=1,512
   60 CORT(L)=RNORM*CONJG(G(L))/FLOAT(32769-L)
      CALL SECOND(TIME)
      WRITE(6,20)ITIM,TIME
C**********************************************************************
      ITIM=5
      DO 190 I=1,128,2
  190 WRITE(6,200)(I+J-2,CORT(I+J-1),J=1,512,128)
  200 FORMAT(4(1H ,I5,2F13.5))
      CALL SECOND(TIME)
      WRITE(6,20)ITIM,TIME
      STOP
      END
```

TABLE 2. Correlation Function Calculated by Conventional Method.

0	1.00000	0.00000	128	0.23785	-0.00651	256	0.10461	0.01195	384	0.03032	0.03986
2	0.99861	-0.00057	130	0.24314	-0.00614	258	0.10536	0.01259	386	0.02898	0.04026
4	0.99449	-0.00112	132	0.24801	-0.00566	260	0.10610	0.01330	388	0.02791	0.04067
6	0.98765	-0.00166	134	0.25258	-0.00522	262	0.10703	0.01404	390	0.02682	0.04114
8	0.97820	-0.00217	136	0.25664	-0.00467	264	0.10793	0.01486	392	0.02599	0.04160
10	0.96619	-0.00266	138	0.26034	-0.00418	266	0.10901	0.01568	394	0.02513	0.04212
12	0.95178	-0.00310	140	0.26348	-0.00359	268	0.11001	0.01660	396	0.02452	0.04263
14	0.93505	-0.00353	142	0.26620	-0.00305	270	0.11116	0.01748	398	0.02387	0.04320
16	0.91622	-0.00390	144	0.26834	-0.00242	272	0.11221	0.01848	400	0.02345	0.04374
18	0.89537	-0.00425	146	0.27003	-0.00187	274	0.11337	0.01940	402	0.02298	0.04434
20	0.87275	-0.00452	148	0.27112	-0.00122	276	0.11440	0.02045	404	0.02273	0.04491
22	0.84847	-0.00480	150	0.27174	-0.00065	278	0.11551	0.02139	406	0.02241	0.04552
24	0.82280	-0.00498	152	0.27177	0.00000	280	0.11648	0.02246	408	0.02231	0.04610
26	0.79583	-0.00517	154	0.27133	0.00058	282	0.11748	0.02339	410	0.02211	0.04670
28	0.76786	-0.00526	156	0.27032	0.00122	284	0.11832	0.02447	412	0.02213	0.04727
30	0.73897	-0.00538	158	0.26885	0.00178	286	0.11916	0.02537	414	0.02204	0.04784
32	0.70948	-0.00540	160	0.26686	0.00241	288	0.11981	0.02643	416	0.02215	0.04837
34	0.67945	-0.00544	162	0.26443	0.00295	290	0.12044	0.02728	418	0.02215	0.04889
36	0.64918	-0.00541	164	0.26153	0.00354	292	0.12087	0.02830	420	0.02233	0.04936
38	0.61874	-0.00540	166	0.25822	0.00405	294	0.12124	0.02909	422	0.02238	0.04979
40	0.58841	-0.00532	168	0.25450	0.00460	296	0.12140	0.03004	424	0.02262	0.05018
42	0.55825	-0.00527	170	0.25042	0.00508	298	0.12148	0.03075	426	0.02270	0.05050
44	0.52853	-0.00517	172	0.24600	0.00558	300	0.12134	0.03163	428	0.02297	0.05079
46	0.49929	-0.00510	174	0.24127	0.00600	302	0.12110	0.03226	430	0.02307	0.05097
48	0.47076	-0.00499	176	0.23628	0.00644	304	0.12063	0.03305	432	0.02334	0.05112
50	0.44297	-0.00492	178	0.23103	0.00681	306	0.12006	0.03358	434	0.02344	0.05115
52	0.41615	-0.00483	180	0.22560	0.00719	308	0.11926	0.03429	436	0.02369	0.05114
54	0.39029	-0.00477	182	0.21998	0.00751	310	0.11833	0.03471	438	0.02377	0.05098
56	0.36560	-0.00470	184	0.21426	0.00782	312	0.11719	0.03533	440	0.02398	0.05079
58	0.34205	-0.00467	186	0.20840	0.00808	314	0.11592	0.03566	442	0.02401	0.05043
60	0.31981	-0.00465	188	0.20253	0.00832	316	0.11443	0.03617	444	0.02415	0.05004
62	0.29886	-0.00465	190	0.19658	0.00853	318	0.11282	0.03641	446	0.02412	0.04946
64	0.27933	-0.00469	192	0.19069	0.00871	320	0.11100	0.03683	448	0.02418	0.04884
66	0.26117	-0.00474	194	0.18478	0.00866	322	0.10906	0.03698	450	0.02407	0.04802
68	0.24449	-0.00483	196	0.17900	0.00898	324	0.10694	0.03731	452	0.02403	0.04715
70	0.22924	-0.00493	198	0.17327	0.00909	326	0.10469	0.03739	454	0.02382	0.04608
72	0.21549	-0.00508	200	0.16772	0.00915	328	0.10229	0.03763	456	0.02366	0.04496
74	0.20317	-0.00523	202	0.16226	0.00921	330	0.09978	0.03765	458	0.02334	0.04362
76	0.19233	-0.00544	204	0.15705	0.00921	332	0.09714	0.03782	460	0.02307	0.04223
78	0.18290	-0.00562	206	0.15196	0.00923	334	0.09440	0.03779	462	0.02263	0.04062
80	0.17489	-0.00587	208	0.14715	0.00919	336	0.09156	0.03790	464	0.02222	0.03896
82	0.16822	-0.00608	210	0.14251	0.00917	338	0.08865	0.03784	466	0.02164	0.03708
84	0.16290	-0.00635	212	0.13817	0.00908	340	0.08568	0.03790	468	0.02110	0.03516
86	0.15882	-0.00658	214	0.13403	0.00904	342	0.08264	0.03781	470	0.02038	0.03303
88	0.15596	-0.00685	216	0.13019	0.00893	344	0.07959	0.03784	472	0.01969	0.03085
90	0.15425	-0.00708	218	0.12658	0.00887	346	0.07648	0.03775	474	0.01883	0.02849
92	0.15361	-0.00733	220	0.12328	0.00875	348	0.07341	0.03775	476	0.01798	0.02609
94	0.15399	-0.00754	222	0.12021	0.00869	350	0.07030	0.03767	478	0.01696	0.02353
96	0.15529	-0.00776	224	0.11744	0.00858	352	0.06727	0.03767	480	0.01596	0.02095
98	0.15746	-0.00794	226	0.11492	0.00854	354	0.06421	0.03762	482	0.01478	0.01823
100	0.16038	-0.00810	228	0.11268	0.00846	356	0.06129	0.03763	484	0.01363	0.01550
102	0.16402	-0.00823	230	0.11069	0.00846	358	0.05833	0.03760	486	0.01228	0.01268
104	0.16824	-0.00833	232	0.10894	0.00843	360	0.05555	0.03763	488	0.01097	0.00986
106	0.17303	-0.00840	234	0.10746	0.00849	362	0.05275	0.03766	490	0.00947	0.00698
108	0.17821	-0.00843	236	0.10620	0.00853	364	0.05016	0.03772	492	0.00801	0.00414
110	0.18381	-0.00843	238	0.10519	0.00866	366	0.04756	0.03780	494	0.00635	0.00127
112	0.18963	-0.00837	240	0.10437	0.00879	368	0.04519	0.03790	496	0.00475	-0.00153
114	0.19572	-0.00830	242	0.10380	0.00902	370	0.04281	0.03804	498	0.00296	-0.00433
116	0.20188	-0.00816	244	0.10338	0.00925	372	0.04069	0.03820	500	0.00123	-0.00703
118	0.20815	-0.00801	246	0.10322	0.00957	374	0.03856	0.03840	502	-0.00068	-0.00969
120	0.21434	-0.00777	248	0.10316	0.00992	376	0.03670	0.03862	504	-0.00251	-0.01224
122	0.22052	-0.00754	250	0.10334	0.01035	378	0.03484	0.03889	506	-0.00451	-0.01470
124	0.22649	-0.00722	252	0.10360	0.01082	380	0.03325	0.03917	508	-0.00643	-0.01703
126	0.23234	-0.00692	254	0.10409	0.01135	382	0.03164	0.03951	510	-0.00850	-0.01924

TABLE 3. Correlation Function Calculated by Fast Fourier Transform Method.

0	1.00000	0.00000	128	0.23786	-0.00662	256	0.10461	0.01194	384	0.03025	0.03984
2	0.99862	-0.00056	130	0.24312	-0.00621	258	0.10530	0.01259	386	0.02898	0.04023
4	0.99449	-0.00112	132	0.24803	-0.00576	260	0.10609	0.01329	388	0.02784	0.04065
6	0.98766	-0.00165	134	0.25256	-0.00528	262	0.10698	0.01404	390	0.02683	0.04110
8	0.97820	-0.00216	136	0.25667	-0.00477	264	0.10793	0.01486	392	0.02592	0.04158
10	0.96619	-0.00265	138	0.26033	-0.00423	266	0.10895	0.01569	394	0.02514	0.04209
12	0.95177	-0.00310	140	0.26352	-0.00367	268	0.11001	0.01658	396	0.02445	0.04262
14	0.93506	-0.00352	142	0.26621	-0.00310	270	0.11111	0.01750	398	0.02387	0.04317
16	0.91620	-0.00390	144	0.26838	-0.00250	272	0.11221	0.01845	400	0.02338	0.04374
18	0.89537	-0.00423	146	0.27004	-0.00190	274	0.11332	0.01942	402	0.02298	0.04432
20	0.87274	-0.00453	148	0.27117	-0.00129	276	0.11441	0.02041	404	0.02266	0.04491
22	0.84848	-0.00478	150	0.27176	-0.00068	278	0.11548	0.02142	406	0.02241	0.04551
24	0.82278	-0.00499	152	0.27182	-0.00006	280	0.11649	0.02242	408	0.02223	0.04611
26	0.79583	-0.00516	154	0.27135	0.00056	282	0.11745	0.02343	410	0.02212	0.04670
28	0.76784	-0.00529	156	0.27037	0.00117	284	0.11834	0.02443	412	0.02206	0.04728
30	0.73898	-0.00538	158	0.26889	0.00177	286	0.11914	0.02541	414	0.02205	0.04785
32	0.70946	-0.00543	160	0.26691	0.00236	288	0.11984	0.02638	416	0.02208	0.04839
34	0.67945	-0.00545	162	0.26447	0.00294	290	0.12042	0.02733	418	0.02215	0.04890
36	0.64916	-0.00545	164	0.26157	0.00350	292	0.12089	0.02825	420	0.02226	0.04938
38	0.61875	-0.00542	166	0.25826	0.00405	294	0.12123	0.02914	422	0.02239	0.04982
40	0.58839	-0.00537	168	0.25454	0.00457	296	0.12142	0.02999	424	0.02254	0.05021
42	0.55826	-0.00530	170	0.25046	0.00507	298	0.12147	0.03081	426	0.02271	0.05054
44	0.52852	-0.00523	172	0.24604	0.00555	300	0.12137	0.03158	428	0.02289	0.05081
46	0.49930	-0.00514	174	0.24131	0.00600	302	0.12110	0.03231	430	0.02308	0.05102
48	0.47074	-0.00506	176	0.23631	0.00642	304	0.12067	0.03300	432	0.02326	0.05115
50	0.44299	-0.00498	178	0.23107	0.00682	306	0.12006	0.03364	434	0.02345	0.05120
52	0.41614	-0.00490	180	0.22563	0.00718	308	0.11929	0.03423	436	0.02362	0.05116
54	0.39031	-0.00484	182	0.22001	0.00751	310	0.11834	0.03477	438	0.02377	0.05104
56	0.36559	-0.00478	184	0.21427	0.00781	312	0.11722	0.03527	440	0.02391	0.05082
58	0.34206	-0.00475	186	0.20843	0.00808	314	0.11593	0.03571	442	0.02401	0.05049
60	0.31980	-0.00473	188	0.20253	0.00832	316	0.11446	0.03611	444	0.02409	0.05006
62	0.29887	-0.00474	190	0.19660	0.00853	318	0.11282	0.03646	446	0.02412	0.04952
64	0.27932	-0.00478	192	0.19068	0.00871	320	0.11103	0.03677	448	0.02412	0.04886
66	0.26118	-0.00484	194	0.18480	0.00887	322	0.10907	0.03703	450	0.02407	0.04808
68	0.24448	-0.00493	196	0.17899	0.00899	324	0.10696	0.03726	452	0.02397	0.04718
70	0.22925	-0.00504	198	0.17328	0.00909	326	0.10470	0.03744	454	0.02381	0.04614
72	0.21548	-0.00519	200	0.16770	0.00916	328	0.10231	0.03759	456	0.02360	0.04498
74	0.20317	-0.00535	202	0.16227	0.00921	330	0.09978	0.03770	458	0.02334	0.04368
76	0.19232	-0.00554	204	0.15702	0.00923	332	0.09715	0.03778	460	0.02301	0.04225
78	0.18289	-0.00575	206	0.15196	0.00923	334	0.09440	0.03784	462	0.02262	0.04068
80	0.17487	-0.00598	208	0.14712	0.00921	336	0.09157	0.03787	464	0.02216	0.03898
82	0.16821	-0.00621	210	0.14251	0.00917	338	0.08866	0.03788	466	0.02164	0.03714
84	0.16287	-0.00646	212	0.13814	0.00911	340	0.08568	0.03787	468	0.02104	0.03517
86	0.15880	-0.00671	214	0.13402	0.00903	342	0.08265	0.03785	470	0.02038	0.03308
88	0.15594	-0.00697	216	0.13016	0.00895	344	0.07958	0.03781	472	0.01964	0.03086
90	0.15422	-0.00722	218	0.12657	0.00886	346	0.07649	0.03778	474	0.01882	0.02853
92	0.15359	-0.00745	220	0.12324	0.00877	348	0.07340	0.03773	476	0.01793	0.02609
94	0.15396	-0.00768	222	0.12019	0.00868	350	0.07031	0.03769	478	0.01696	0.02356
96	0.15526	-0.00789	224	0.11741	0.00860	352	0.06725	0.03766	480	0.61591	0.02094
98	0.15742	-0.00808	226	0.11489	0.00853	354	0.06423	0.03763	482	0.01479	0.01824
100	0.16036	-0.00824	228	0.11264	0.00848	356	0.06125	0.03761	484	0.01358	0.01549
102	0.16398	-0.00837	230	0.11065	0.00845	358	0.05834	0.03761	486	0.01229	0.01268
104	0.16822	-0.00847	232	0.10891	0.00845	360	0.05551	0.03762	488	0.01092	0.00984
106	0.17298	-0.00853	234	0.10742	0.00848	362	0.05276	0.03765	490	0.00948	0.00697
108	0.17819	-0.00856	236	0.10617	0.00855	364	0.05011	0.03771	492	0.00796	0.00411
110	0.18376	-0.00855	238	0.10515	0.00866	366	0.04757	0.03779	494	0.00636	0.00125
112	0.18962	-0.00850	240	0.10434	0.00881	368	0.04513	0.03789	496	0.00470	-0.00158
114	0.19568	-0.00841	242	0.10375	0.00901	370	0.04282	0.03802	498	0.00297	-0.00436
116	0.20186	-0.00828	244	0.10336	0.00927	372	0.04063	0.03818	500	0.00118	-0.00709
118	0.20811	-0.00811	246	0.10316	0.00957	374	0.03857	0.03838	502	-0.00066	-0.00974
120	0.21434	-0.00789	248	0.10314	0.00993	376	0.03664	0.03860	504	-0.00256	-0.01230
122	0.22049	-0.00764	250	0.10329	0.01035	378	0.03484	0.03886	506	-0.00450	-0.01476
124	0.22650	-0.00734	252	0.10359	0.01082	380	0.03318	0.03915	508	-0.00647	-0.01710
126	0.23231	-0.00700	254	0.10403	0.01135	382	0.03165	0.03948	510	-0.00848	-0.01931

LISTING 3. Evaluation of Correlation Function by Overlapping FFT Method.

```
      COMPLEX G(33000),CORT(512),GL(512,4),GS(512,2),WL(5120),WS(2560)
      WRITE(6,10)
10    FORMAT(1H1)
      ITIM=0
      CALL SECOND(TIME)
      WRITE(6,20)ITIM,TIME
20    FORMAT(1H0,'TIME AT STAGE',I3,' = ',F20.10)
C***********************************************************************
      ITIM=1
      READ(7)G
      CALL SECOND(TIME)
      WRITE(6,20)ITIM,TIME
C***********************************************************************
      ITIM=2
      IND=1
      ISG=-1
      CALL CFFT2(IND,ISG,2048,GL,WL,GL)
      CALL CFFT2(IND,ISG,1024,GS,WS,GS)
      IND=0
      DO 25 L=1,512
      GL(L,1)=G(L)
      CORT(L)=(0.0,0.0)
25    CONTINUE
      CALL SECOND(TIME)
      WRITE(6,20)ITIM,TIME
C**********************************************************************
      ITIM=3
      G1=1.0/2048.0
      G2=1.0/1024.0
      IBLK=512
30    DO 35 L=1,512
      GS(L,1)=G(L+IBLK)
      GL(L,2)=G(L+IBLK)
35    CONTINUE
      DO 36 L=1,512
      GS(L,2)=(0.0,0.0)
      GL(L,3)=(0.0,0.0)
      GL(L,4)=(0.0,0.0)
36    CONTINUE
      CALL CFFT2(IND,ISG,2048,GL,WL,GL)
      CALL CFFT2(IND,ISG,1024,GS,WS,GS)
      DO 37 L=1,512
      GS(L,1)=GS(L,1)*CONJG(GS(L,1))
      GS(L,2)=GS(L,2)*CONJG(GS(L,2))
      GL(L,1)=GL(L,1)*CONJG(GL(L,1))
      GL(L,2)=GL(L,2)*CONJG(GL(L,2))
      GL(L,3)=GL(L,3)*CONJG(GL(L,3))
      GL(L,4)=GL(L,4)*CONJG(CL(L,4))
37    CONTINUE
      CALL CFFT2(IND,ISG,2048,GL,WL,GL)
      CALL CFFT2(IND,ISG,1024,GS,WS,GS)
      DO 38 L=1,512
      CORT(L)=CORT(L)+G1*CL(L,1)-G2*GS(L,1)
      GL(L,1)=G(L+IBLK)
38    CONTINUE
      IBLK=IBLK+512
      IF(IBLK.LT.32768)GO TO 30
      DO 40 L=1,512
40    CORT(L)=CORT(L)+G2*GS(L,1)
      CALL SECOND(TIME)
      WRITE(6,20)ITIM,TIME
C***********************************************************************
```

```
      ITIM=4
      RNORM=32768.0/REAL(CORT(1))
      DO 60 L=1,512
60    CORT(L)=RNORM*CONJG(CORT(L))/FLOAT(32769-L)
      CALL SECOND(TIME)
      WRITE(6,20)ITIM,TIME
C***********************************************************************
      ITIM=5
      DO 190 I=1,128,2
190   WRITE(6,200)(I+J-2,CORT(I+J-1),J=1,512,128)
200   FORMAT(4(1H ,I5,2F13.5))
      CALL SECOND(TIME)
      WRITE(6,20)ITIM,TIME
      STOP
      END
```

References

1. W. Smith, "An Introduction to the Discrete Fourier Transform" in CCP5 Information Quarterly No. 5. June 1982 (and references cited therein).

2. D. Fincham, B.J. Ralston, "Molecular Dynamics Simulation Using the CRAY-1 Vector Processing Computer". C.P.C. (1981). 23 127.

3. W.P. Petersen, "Complex Fast Fourier Transform Binary Radio Subroutine (CFFT2)". CRAY-1 Document 2240203.

Virtual trajectories method for obtaining shear viscosity by Molecular Dynamics: The expansion approach

by

D. Fincham and D.M. Heyes.

1. Introduction.

In this note we report the methodology for obtaining shear viscosity by the method of differential trajectories[1]. A sequence of sheared liquids is generated (in the limit of infinitesimal shear rate) by deviating from equilibrium trajectories through the application of a constant shear rate.

There are two ways of performing this. In the first method sheared and unsheared trajectories are formed separately, starting from the same point in phase space. By subtracting the shear stresses at comparable times from this point one eliminates irrelevant fluctuations not pertinent to the property being investigated[2]. In the second method we create this difference in shear stress without having to return to the same point in phase space. In fact, by expansion of coordinates, the difference between the actual and virtual trajectories is propagated without leaving the equilibrium system. Note that the virtual trajectories themselves are not generated as is involved in the first approach. This has advantages in computational execution speed and numerical accuracy. The second method is, consequently, outlined in the next section.

2. Basic Equations.

The equations of motion for the disturbed and undisturbed trajectories through phase space are as follows. For the equilibrium state by Newton's second law of motion,

$$m_i \ddot{\underset{\sim}{r}}_i = \underset{\sim}{F}_i(\underset{\sim}{r}^N), \tag{1}$$

where $\underset{\sim}{r}^N$ represents the set of vectors of the N-particle MD cell. $\underset{\sim}{r}_i$ is the position of molecule i. $\underset{\sim}{F}_i$ is the force on i and m_i is the mass of molecule i.

In contrast, the perturbed state evolves according to the following equation of motion. Imagine we apply the perturbation at time t=0. Then,

$$\begin{aligned} m_i(\ddot{\underset{\sim}{r}}_i + \delta\ddot{\underset{\sim}{r}}_i) &= \underset{\sim}{F}_i(\underset{\sim}{r}^N + \delta\underset{\sim}{r}^N) + \underset{\sim}{F}_i^{ext} \\ &= \underset{\sim}{F}_i(\underset{\sim}{r}^N) + \delta\underset{\sim}{F}_i + \underset{\sim}{F}_i^{ext}, \end{aligned} \tag{2}$$

where $\underset{\sim}{F}_i^{ext}$ is some external force on i, $\delta\underset{\sim}{r}^N$ are the deviant trajectories of the N molecules (at some time t > 0) and $\delta\underset{\sim}{F}_i$ is the consequent change in internal force on i at a later time, resulting from the changed motions caused by $\underset{\sim}{F}_i^{ext}$. Now, subtracting and taking components,

$$m_i \, \delta\ddot{r}_{i\alpha} = \delta F_{i\alpha} + F_{i\alpha}^{ext}. \tag{3}$$

If we note that,

$$F_{i\alpha} = \sum_{j \neq i} g(r_{ij}) \, r_{ij\alpha} \tag{4}$$

where $\underset{\sim}{r}_{ij} = \underset{\sim}{r}_i - \underset{\sim}{r}_j$ and that the force on i is simply a function of the r_{ij} components then (to first order) the change in force due to the altered state is,

$$\delta F_{i\alpha} = \sum_{j\neq i} \sum_{\beta} \frac{\partial}{\partial r_{ij\beta}} \{g(r_{ij}) r_{ij\alpha}\} \, \delta r_{ij\beta} \; . \tag{5}$$

The sum over x, y and z components is denoted by $\sum_{\beta}$. But as,

$$\frac{\partial}{\partial r_{ij\beta}} \{g(r_{ij}) r_{ij\alpha}\} = \frac{dg_{ij}}{dr_{ij}} \cdot \frac{\partial r_{ij}}{\partial r_{ij\beta}} \cdot r_{ij\alpha} + g(r_{ij})\delta_{\alpha\beta}, \tag{6}$$

$$r_{ij} = [r_{ijx}^{\;2} + r_{ijy}^{\;2} + r_{ijz}^{\;2}]^{\frac{1}{2}} \tag{7}$$

and

$$\frac{\partial r_{ij}}{\partial r_{ij\beta}} = \frac{1}{2} \, [r_{ijx}^{\;2} + r_{ijy}^{\;2} + r_{ijz}^{\;2}]^{\frac{1}{2}} . 2r_{ij\beta}$$

$$= \frac{r_{ij\beta}}{r_{ij}} \; , \tag{8}$$

then we have,

$$\delta F_{i\alpha} = \sum_{j\neq i} \{\sum_{\beta} \frac{1}{r_{ij}} \frac{dg}{dr_{ij}} r_{ij\alpha} \, r_{ij\beta} \, \delta r_{ij\beta} + g(r_{ij}) \delta r_{ij\alpha}\}, \tag{9}$$

Note that all the functions of $\underset{\sim}{r}_{ij}$ are derived from the equilibrium state. For the Lennard-Jones system then,

$$g(r) = (24\varepsilon) \frac{1}{r^2} \{2(\frac{\sigma}{r})^{12} - (\frac{\sigma}{r})^{6}\}$$

$$= (24\varepsilon) \; \{2\sigma^{12} r^{-14} - \sigma^6 \, r^{-8}\}, \tag{10}$$

and

$$\frac{1}{r} \frac{dg}{dr} = (\frac{24\varepsilon}{r}) \; \{-28\sigma^{12} r^{-15} + 8\sigma^6 r^{-9}\}$$

$$= (24\varepsilon) \frac{1}{r^4} \{-28 \; (\frac{\sigma}{r})^{12} + 8 \; (\frac{\sigma}{r})^{6}\} \quad . \tag{11}$$

Any property of the difference state can now be determined. Let us consider the stress tensor (the off diagonal component of which is used to calculate η, the shear viscosity).

The expression for the equilibrium stress tensor, $\underset{\approx}{\tau}$, is

$$\underset{\approx}{\tau} = \frac{1}{V}\{\sum_{i} m_i \dot{\underset{\sim}{r}}_i \dot{\underset{\sim}{r}}_i + \sum_{i<j}\sum \underset{\sim}{r}_{ij}\, \underset{\sim}{F}_{ij}\}. \qquad (12)$$

Similarly for the disturbed state we have,

$$\underset{\approx}{\tau} + \delta\underset{\approx}{\tau} = \frac{1}{V}\{\sum_{i} m_i(\dot{\underset{\sim}{r}}_i + \delta\dot{\underset{\sim}{r}}_i)(\dot{\underset{\sim}{r}}_i + \delta\dot{\underset{\sim}{r}}_i)$$

$$+ \sum_{i<j}^{N}\sum^{N} (\underset{\sim}{r}_{ij} + \delta\underset{\sim}{r}_{ij})(\underset{\sim}{F}_{ij} + \delta\underset{\sim}{F}_{ij})\}, \qquad (13)$$

which leads to,

$$\delta\underset{\approx}{\tau} \approx \frac{1}{V}\{m_i(\dot{\underset{\sim}{r}}_i\delta\dot{\underset{\sim}{r}}_i + \delta\dot{\underset{\sim}{r}}_i\dot{\underset{\sim}{r}}_i) +$$

$$\sum_{i<j}^{N}\sum^{N} (\delta\underset{\sim}{F}_{ij}\underset{\sim}{r}_{ij} + \underset{\sim}{F}_{ij}\delta\underset{\sim}{r}_{ij})\}\ , \qquad (14)$$

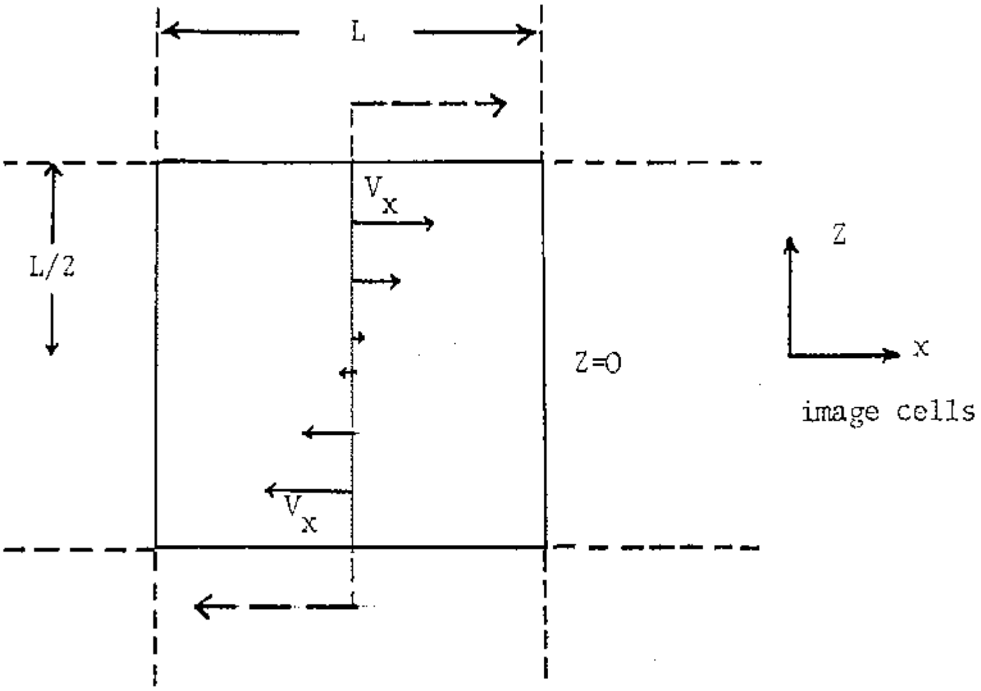
on ignoring terms in $\delta\underset{\sim}{r}_{ij}^{2}$ and $\delta\underset{\sim}{r}_{ij}\ \delta\underset{\sim}{F}_{ij}$.

3. Maintenance of a Velocity gradient.

We want a linear velocity gradient,

$$\frac{d}{dz}\langle v_x\rangle = \dot{\varepsilon}\ , \qquad (15)$$

where $\dot{\varepsilon}$ is a constant. This is pictorially represented below for a two dimensional cross-section through the MD cell.



Initially this is set-up by making,

$$\delta\dot{x}_i = z_i\dot{\varepsilon} \ . \tag{16}$$

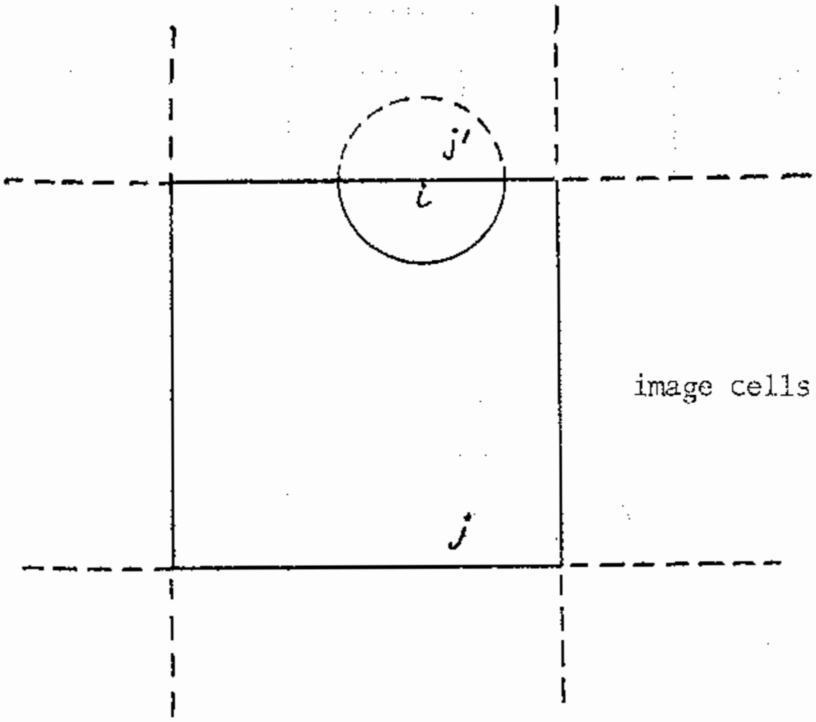
Here $\dot{\varepsilon}$ can be arbitrarily small. In order to maintain this velocity profile when particles cross the xy boundaries then,

$$\delta\dot{x}_i \rightarrow \delta\dot{x}_i - L\dot{\varepsilon} \ , \tag{17}$$

if it passes beyond $z_i = L/2$ and comes back at $z = -L/2$. Conversely if it crosses at $z = -L/2$ then,

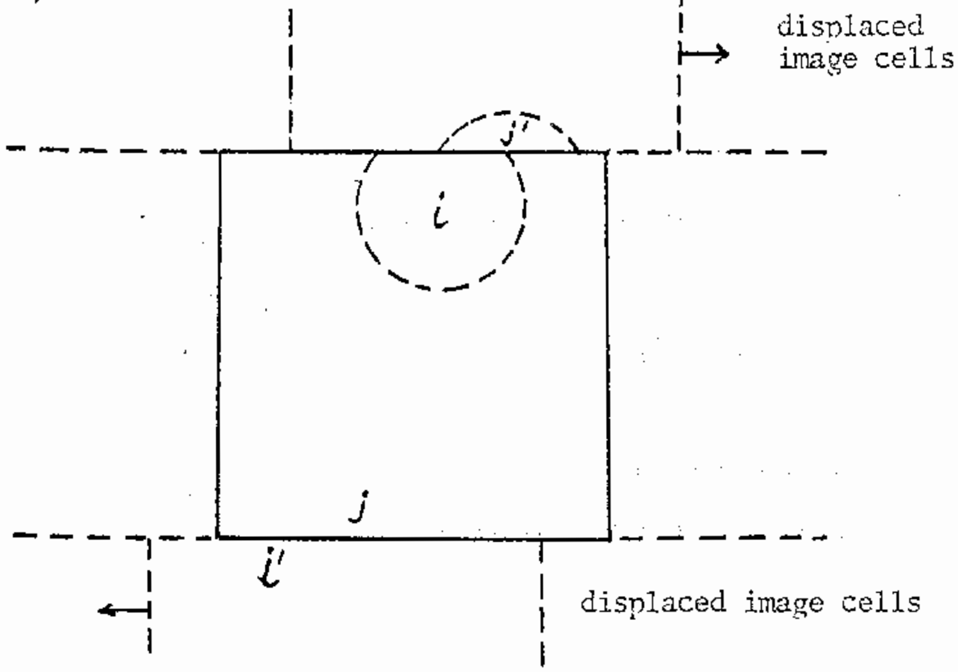
$$\delta\dot{x}_i \rightarrow \delta\dot{x}_i + L\dot{\varepsilon} \ . \tag{18}$$

We also need to change the xy co-ordinates as follows. Initially, molecule i has the image environment:



where the prime denotes an image particle.

After time t the sphere of interactions about molecule i and which crosses a cell boundary becomes split. This is a requirement of laminar shear flow which, obviously, has to continue smoothly beyond the real MD cell walls.



Hence for the above interactions,

$$\delta x_{ij'} = \delta x_{ij} + L\dot{\varepsilon}t. \qquad (19)$$

The sign in eqn. (19) would be changed if the interactions were across the opposite face of the MD cell. Also when the particle i crosses the z boundary at z = L/2 then,

$$\delta x_i \rightarrow \delta x_i - L\dot{\varepsilon}t \ . \qquad (20)$$

In the limit of long time (and hopefully before "noise" sets in) the shear viscosity, η, for this simple monatomic liquid is given by a plateau in the function,

$$\eta = - \delta\tau_{xz} \ (t \rightarrow \infty)/\dot{\varepsilon}, \qquad (21)$$

for shear flow in the xz plane.

It is interesting to speculate upon the origin of the increasing uncertainty in $\delta\tau_{xz}$ with time. Test runs on a liquid state Lennard-Jones system using a truncated potential were made. It could be envisaged that there are problems associated with impulses on molecule i created by particles crossing the boundary of the truncation sphere around molecule i. One way of remedying this is to use a shifted force potential. Calculations were performed on one in which both the potential and force go to zero at the cut-off distance for interactions. Indistinguishable results from those of the truncated Lennard-Jones potential were obtained. Also halving the time step from the usual value did not reduce this noise. One must conclude, therefore, that the noise is probably physical rather than computational in origin.

The deviant shear stress response is mainly a function of the differential positions and is governed by the potential surface on which they travel. As one usually can only sample a limited number (approximately 40) of such independent host trajectories it is perhaps not surprising that the individual stress responses will follow different paths. The individual track histories differ from each other more with time because the differential positions and velocities at time t are the compounded effect of such previous differential co-ordinates and velocities. Therefore the "noise" observed at long times ($t \sim$ 1-2 psec) in the difference in trajectories method is probably a natural property of the system - rather than a computational artefact.

References.

1. K. Singer, J.V.L. Singer and D. Fincham, Mol.Phys., 40, 515 (1980).
2. D.M. Heyes, J.J. Kim, C.J. Montrose and T.A. Litovitz, J.Chem.Phys., 73, 3987 (1980).

Molecular Dynamics Timings on a Selection of Computers

by

D. Fincham and D.M. Heyes

We are now in a period of great change in the computing environment. The previous generation of CDC machines (e.g. 6600 and 7600), which have been so valuable to the MD and MC simulator in the last decade, are soon to be replaced at UMRCC and ULCC by a Cyber 205 and a CRAY-1S, respectively. These are called vector machines and will require some recoding of your MD or MC program to take advantage of their computer architecture. These changes can lead to factors of three or four in increased speed of execution over unvectorised codes and is strongly recommended. It can often be accomplished in an afternoon's undemanding work.

An alternative approach has been adopted in the ICL 2980/DAP computer configuration. This is a truly "parallel" machine and consists of 4096 independent 1 bit CPU's. The MD technique lends itself well to this machine, being a problem which involves doing the same operation independently on many array elements. One can also choose the level of arithmetic precision. Mathematical operations are performed in the software and not in the hardware as in some other machines. DAP's are now sited at Edinburgh University and Queen Mary College, London, and a machine at NPL provides a bureau service.

Alongside the developments in such large "number crunching" computers the last few years has seen the introduction of College affordable mini-computers (e.g. Digital's VAX 11/780) and user affordable micro-computers such as the ICL PERQ. We are entering an era of distributed interactive computing. Dedicated single user computers with superb graphics will be inter-connected through a local area network and consequently offer some of the mainframe resouces.

These machines, which although sometimes orders of magnitude slower than the larger super computers, can be dedicated to a single user for many hours. Thus they provide comparable computing power within the same real time to the simulator. An exciting development which should be available in 1984 is the ICL PERQ/DAP combination which will provide super computer power in your office at a reasonable cost.

A change over from one computer to another always requires an unfruitful adjustment period. The purpose of this report is to provide the simulator with an estimate of how long his or her job will take to run a variety of available computer systems. We hope to provide the answer to: Is it worth my spending time setting up a problem on a particular computer (which I may have access to)?

We have considered the simple Molecular Dynamics, MD, system of spherically symmetric molecules enclosed in a cubic MD cell. The popular Lennard-Jones, LJ, interaction potential was used[1],

$$\phi(r) = 4\varepsilon((\sigma/r)^{12} - (\sigma/r)^{6}), \qquad (1)$$

where ε and σ are the well depth and a typical distance parameter, respectively. The truncation radius for all interactions, r_c, was 2.5σ. No pair radial distribution function was evaluated as it is a property of user specific interest and can increase the computer time by 25%, depending on how it is evaluated. Here we have confined the calculation to the potential energy and virial repulsive and attractive potential components, and also to the forces[1]; these being the usual minimum requirements. Two densities (and temperatures) were considered. The near triple point state of reduced number density: $\rho^* = 0.8442$ (or $\rho = 0.8442\ \sigma^{-3}$) and reduced temperature $T^* = 0.722$ (or $T = 0.722\varepsilon/k_B$) was considered. Another, low density, state was investigated: $\rho^* = 0.5$ and $T^* = 1.5$. We will first consider the cpu times per time increment, Δt, using conventional MD i.e., going through all pair interactions each step. Table 1 compares seven computers with N (the number of molecules in the MD cell) = 108,256 and 864.

Both the VAX11/780 and Amdahl have a 32 bit single precision, SP, word length. Double precision, DP, was also investigated. It is unlikely whether the inaccuracies involved in SP warrant the fourfold increase in computer time associated with DP on the VAX. The errors introduced in the equations of motion by the Verlet algorithm[1] itself are sufficient to eliminate any advantages of DP towards more accurate trajectories.

The Amdahl (based at ULCC) is half way between a 6600 and 7600 (60 bit word SP) in speed. A vectorised LJ program is the fastest of all on the CRAY-1S and is equivalent to "5 or 6 CDC 7600's". Using the same program but turning off the vectorisation optimisation produces a machine which is about twice as fast as the CDC 7600, i.e. the floating point hardware is better.

In Tables 2 and 3 a comparison between the conventional and neighbourhood table, N.T., method is made[2]. The N.T. scheme is a technique for speeding the execution time of an MD program by using a not too excessive amount of cpu memory. At intervals a table of neighbour indices is created. All particle pairs which are separated by a distance less than r_{co}, where $r_{co} > r_c$, are included in this table. On the DAP a logical mask marking in-range interactions is used instead of an actual list of neighbour indices. Since the DAP processors are single bit this is most economical in computations and storage.

This table is used for the succeeding time steps to select those pairs which have a chance of being within the cut-off sphere of radius r_c. The N.T. are updated whenever there is a chance of a pair which were formally separated by $r > r_{co}$ now being separated by $r < r_c$, through the intervening particle motion. Tables 2 and 3 reveal that it is favourable for the N.T. to be recreated fairly frequently so that the tables are quite small in size ($r_{co}-r_c = 0.32\sigma$ is a popular choice) Obviously there will come a point with further reductions in r_{co} ($\rightarrow r_c$) when the computer time will start to rise again.

The N.T. method gains most for larger N (when there are more interactions to be ignored) and for scalar machines (which are all of those considered except the CRAY-1S and the DAP). The N.T. method accesses pairs in the MD force loop which are not sequentially related, in terms of their array indices. This prevents vectorisation. It can be made partially vectorisable by so-called "gather and scatter" procedures[2]. But still the gain in speed is only approximately 25% at most for N = 256. This compares unfavourably with the gains of order 100% on the scalar machines for this number of particles. Of course, running the CRAY-1S in scalar mode will restore this advantage of N.T. over conventional MD. But it is still faster to maintain the vectorisation optimisation for overall N.T. speed of execution as Tables 2 and 3 reveal.

References.

1. J.P. Hansen and I.R. McDonald, Theory of Simple Liquids (Academic Press, New York, 1976).

2. D. Fincham and B.J. Ralston, Computer Physics Communication 23, (1981) 127: "Molecular Dynamics Simulation using the CRAY-1 Vector Processing Computer".

TABLE 1.

CPU times per time step on a variety of Computers by conventional Molecular Dynamics.

Key: a) quantities in brackets denote double precision arithmetic, otherwise single precision arithmetic was used.

b) V. the program was run in vector mode(CFT,ON=V.) on the CRAY-1S.

c) S. the program was run in scalar mode (CFT,OFF=V.) on the CRAY-1S. The VAX used does not have a floating point accelerator.

Computer:	ICL	ICL	DIGITAL	AMDAHL	CDC	CDC	CRAY-1S	
	PERQ	2980	VAX		6600	7600	V.	S.
		DAP	11/780					
$\rho^*=0.5$ $T^*=1.5$								
N								
108		0.05	0.88 (3.0)	(0.09)	0.19	0.04	0.007	0.02
256		0.11	3.9 (12)	(0.45)	0.93	0.18	0.03	0.12
864		1.09	37 (124)	(4.5)	11	1.8	0.28	1.3
$\rho^*=0.8442$ $T^*=0.722$								
N								
108	5*	0.05	0.95 (3.6)	(0.10)	0.23	0.04	0.007	0.02
256	20*	0.11	4.4 (13)	(0.48)	1.0	0.19	0.03	0.12
864		1.09	39 (126)	(4.7)	11	1.8	0.28	1.3

* using POS operating system; estimated three times faster using UNIX operating system.

TABLE 2.

Neighbourhood Tables (N.T.) on the VAX 11/780 and CRAY-1S.

As for Table 1, V and S stand for vector and scalar optimisation modes on the CRAY-1S.

C denotes conventional MD.

$\Delta r = r_{co} - r_c$, the thickness of the boundary shell between interaction and N.T. cutoffs.

The time step was $\Delta t^* = 0.005\sigma(m/\varepsilon)^{\frac{1}{2}}$.

SP denotes single precision arithmetic.

DP denotes double precision arithmetic.

State point: $\rho^* = 0.5$, $T^* = 1.5$.

δt is the computer time per time increment.

N_e is the average number of entrants in the N.T.

N_f is the average number of time steps in-between the recreation of the neighbour lists.

Computer	Method	Mode of Arithmetic	N	$\frac{\Delta r}{\sigma}$	N_e	N_f	$\frac{\delta t}{\text{sec}}$
VAX	C	SP	256				3.9
VAX	N.T.	SP	256	0.323	6000	10	1.41
VAX	N.T.	SP	256	0.587	7000	18	1.44
VAX	C	DP	256				12
VAX	N.T.	DP	256	0.323	6000	10	4.8
VAX	N.T.	DP	256	0.587	8000	18	5.4
CRAY	C	SP/S	256				0.12
CRAY	N.T.	SP/S	256	0.323	7000	10	0.05
CRAY	N.T.	SP/S	256	0.587	9000	20	0.05
CRAY	C	SP/V	256				0.029
CRAY	N.T.	SP/V	256	0.323	7000	10	0.021
CRAY	N.T.	SP/V	256	0.587	10000	26	0.024
VAX	C	SP	864				37
VAX	N.T.	SP	864	0.323	20000	20	8
CRAY	C	SP/S	864				1.3
CRAY	N.T.	SP/S	864	0.323	20000	16	0.22
CRAY	N.T.	SP/S	864	0.734	34000	28	0.23
CRAY	C	SP/V	864				0.28
CRAY	N.T.	SP/V	864	0.323	20000	16	0.09
CRAY	N.T.	SP/V	864	0.734	34000	24	0.10

TABLE 3.

Description as for Table 2 except: $\rho^* = 0.8442$, $T^* = 0.722$

Computer	Method	Mode of Arithmetic	N	$\frac{\Delta r}{\sigma}$	N_e	N_ℓ	$\frac{\delta t}{\text{sec}}$
VAX	C	SP	256				4.4
VAX	N.T.	SP	256	0.323	10000	12	2.5
VAX	N.T.	SP	256	0.734	17000	30	3.2
VAX	C	DP	256				13
VAX	N.T.	DP	256	0.323	10000	12	6.9
VAX	N.T.	DP	256	0.734	17000	30	9.5
CRAY	C	SP/S	256				0.12
CRAY	N.T.	SP/S	256	0.323	11000	18	0.06
CRAY	N.T.	SP/S	256	0.734	17000	50	0.08
CRAY	C	SP/V	256				0.029
CRAY	N.T.	SP/V	256	0.323	11000	20	0.027
CRAY	N.T.	SP/V	256	0.734	17000	50	0.038
VAX	C	SP	864				39
VAX	N.T.	SP	864	0.323	34000	14	10
CRAY	C	SP/S	864				1.3
CRAY	N.T.	SP/S	864	0.323	34000	20	0.25
CRAY	N.T.	SP/S	864	0.734	60000	30	0.33
CRAY	C	SP/V	864				0.28
CRAY	N.T.	SP/V	864	0.323	34000	20	0.11
CRAY	N.T.	SP/V	864	0.734	60000	30	0.15

TABLE 4.

The fastest CPU times per time step, δt/sec, on a variety of computers.

Key: C denotes conventional MD.

V denotes vector optimisation on the CRAY-1S.

S denotes scalar operation mode on the CRAY-1S.

The N = 256 and 864 calculations were performed using neighbourhood tables (N.T.) with $\Delta r = 0.323\sigma$.

{...} denotes conventional method using PASCAL. All other programs were written in FORTRAN IV.

Computer	ICL	ICL	DIGITAL	AMDAHL	CDC	CDC	CRAY-1S	
	PERQ	2980	VAX		6600	7600	V.	S.
		/DAP	11/780					
$\rho^* = 0.5$ $T^* = 1.5$								
N/Method								
108/C:	{5}*	0.05	0.88 (3.0)	(0.09)	0.19	0.04	0.007	0.02
256/N.T.:	{20}*	((0.11))	1.4 (4.8)	(0.18)	0.4	0.08	0.021	0.05
864/N.T.:		((1.09))	8	(0.95)		0.4	0.09	0.22
$\rho^* = 0.8442$ $T^* = 0.722$								
108/C:	{5}*	0.05	0.95 (3.6)	(0.10)	0.23	0.04	0.007	0.02
256/N.T.:	{20}*	((0.11))	2.5 (6.9)	(0.25)	0.55	0.10	0.027	0.06
864/N.T.:		((1.09))	9.8	(1.1)			0.11	0.25

* using POS operating system; estimated three times faster using the UNIX operating system.

(( )) denote C