REF CORY

DARESBURY LABORATORY

# INFORMATION QUARTERLY for COMPUTER SIMULATION OF CONDENSED PHASES

An informal Newsletter associated with Collaborative Computational Project No. 5 on Molecular Dynamics, Monte Carlo and Lattice Simulations of Condensed Phases.

Number 31 OCTOBER 1989

## Contents



Editor: Dr. William Smith
Deputy Editor: Dr. Maurice Leslie

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

## General News

**CCP5 ANNUAL MEETING. CAMBRIDGE. 1989** The annual meeting of CCP5 will be held on December 17-19th 1989 in Cambridge. There will also be a meeting of the steering committee at which a new chairman will need to be appointed to replace Professor Catlow. It is also hoped to hold a round table discussion to highlight future work which will become possible with the availability of greater computing power. Invited speakers include

S. Garofalini. (Rutgers, USA)
N. Quirke (BP)
D. Tildesley (Southampton)
J. Goodfellow (Birkbeck)
P. King (Oxford)
J. Clarke (UMIST)
E. Colbourne (ICI)

A registration form is included in this newsletter.

**ARCHITECTURE AND ALGORITHMS** Information is included in this newsletter on a meeting planned for the summer of 1990.

**WORKSHOP ON PARALLEL ALGORITHMS** This workshop will be held in November at Daresbury. Further details are included in this issue of the newsletter.

**VISIT OF PROF. BINDER TO THE UK** Professor Binder will visit the UK during November. Details of his program may be found in this newsletter.

**CRAY TIME** CCP5 participants are reminded that CCP5 has an annual allocation of Cray time at Rutherford (Cray XMP-48), which is available for the development of simulation programs which are of general use to the CCP5 community. Readers who wish to use some of this allocation should write to the CCP5 Secretary, Dr. M. Leslie, TCS Division, SERC Daresbury Laboratory, Daresbury, Warrington WA4 4AD.

Contributors to the current issue.
Our thanks go to:

| | |
|---|---|
| D. Fincham | Computer Centre,<br>University of Keele,<br>Keele, Staffs ST5 5BG. |
| D. Brown<br>J. Clarke | Chemistry Department,<br>UMIST, Sackville Street<br>Manchester M60 1QD. |
| W. Smith<br>F. Müller-Plathe | Theory and Computational Science<br>Division, S.E.R.C. Daresbury<br>Laboratory, Daresbury, Warrington<br>WA4 4AD, Cheshire. |
| D. W. Noid | Chemistry Division,<br>Oak Ridge National Laboratory<br>Oak Ridge USA. |
| G. A. Pfeffer | Department of Chemistry,<br>University of Nebraska,<br>Omaha. USA. |
| B. G. Stumper | Department of Chemistry,<br>University of Tennessee,<br>Knoxville. USA. |
| S. K. Gray | Department of Chemistry,<br>Northern Illinois University,<br>DeKalb. USA. |
| M. P. Allen | H. H. Wills Physics Laboratory<br>Royal Fort, Tyndall Avenue,<br>Bristol. BS8 1TL |
| D. J. Tildesley | Department of Chemistry,<br>The University<br>Southampton. SO9 5NH |

CCP5 ANNUAL CONFERENCE 1989
GRAND CHALLENGES IN MOLECULAR SIMULATION
DOWNING COLLEGE, CAMBRIDGE. 17-19 DECEMBER 1989

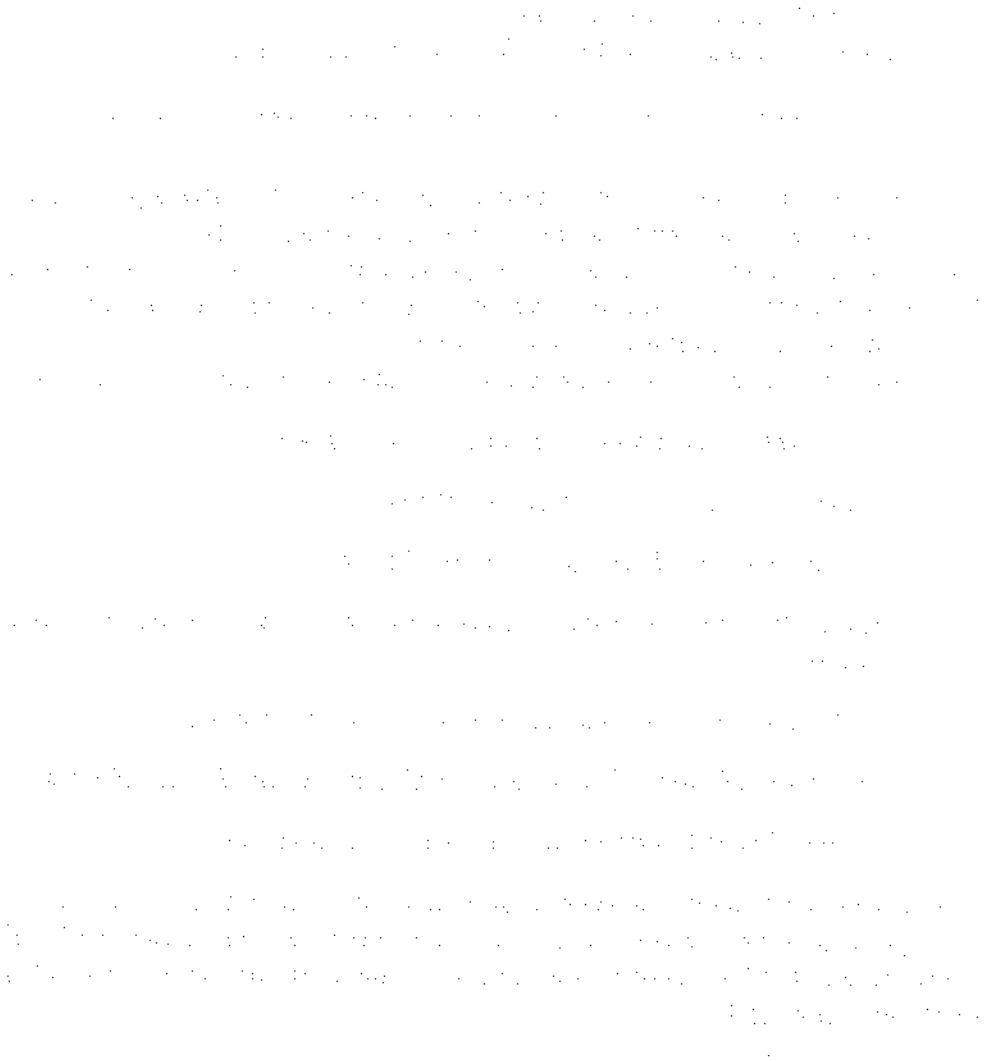
**********

The conference will be held at **Downing College, Cambridge** from 17-19 December 1989. Accommodation will be in Downing College. The subject matter of the conference will be in the general area of molecular simulation but particular attention will be given to applications which are at the limits of current theory or available computer resource.

Among the speakers who have agreed to take part in the conference are:

- D. Tildesley (Southampton) Simulation of surfaces.
- J. Clarke (UMIST) Simulation of Polymers.
- J. Goodfellow (Birkbeck) Simulation of DNA.
- P. King (Oxford) Free energy calculations of pharmaceutically important molecules.
- E. Colbourne (ICI) Modelling of heterogeneous catalysis.
- S. Garofalini (Rugters University, USA) Simulation of oxide glasses.
- N. Quirke (British Petroleum) Industrial applications.

Further contributions in the general area molecular simulation would be welcome. The meeting will start at 9. am. on Monday 18. December and finish at lunchtime on 19th. Accommodation is available on the night of the 17th. for those requiring it.



CCP5 ANNUAL MEETING - DOWNING COLLEGE, UNIVERSITY OF CAMBRIDGE

## GRAND CHALLENGES IN MOLECULAR SIMULATION

17-19 DECEMBER 1989

## REGISTRATION FORM

| | |
|---|---|
| Accommodation for 2 nights plus meals. | £83.50..... |
| Accommodation for 2 nights plus meals. | £50 ..... |
| (Students only; limited availability so first come first served.) | |
| Registration, lunch and dinner on 18th. | £38.50..... |
| Registration and lunch on 18th. only. | £23.50..... |
| Lunch on 19th. (Additional to one of above) | £7.50 ..... |

(Cheque payable to SERC CCP5)

Contributed Papers:

I do / do not wish to contribute a paper entitled

.....

I enclose an abstract of my paper

NAME .....

ADDRESS .....

.....

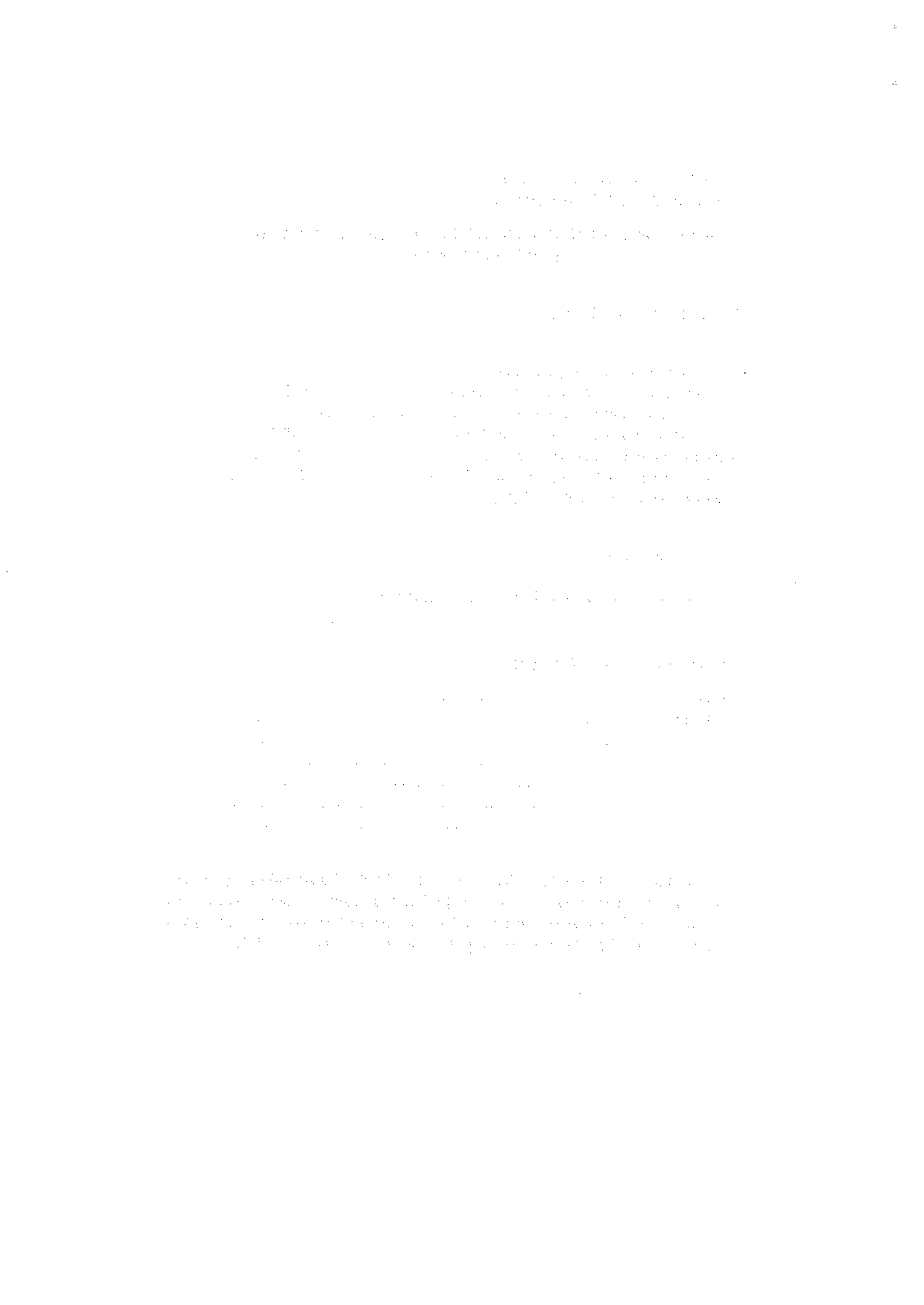
.....

.....

.....

.....

The deadline for abstracts is *November 15th 1989*. Please submit your abstract on A4 paper with a 4 cm. right hand margin. Please return with payment to M. Leslie or W. Smith, SERC Daresbury Laboratory, Warrington WA4 4AD, UK. The deadline for registration is *December 1st 1989*.



# WORKSHOP ON PARALLEL ALGORITHMS IN MOLECULAR SIMULATION

Daresbury Laboratory

8th. November 1989

The purpose of this workshop is to bring together those groups currently working on real scientific problems and using parallel processing as the method of solution. The meeting will be an informal one-day affair, in which participants will present short (15 minute) talks on their methods. There will be opportunity for discussion and speculative suggestions. The number of participants will not exceed 30.

We hope to attract participants with interests the following areas:

- Systolic loop algorithms for MD and Monte Carlo.
- Hierarchical (master-slave) algorithms.
- Algorithms for multiple, independent simulations.
- Ewald sum and other methods for Coulombic Systems.
- Algorithms for heterogeneous systems (e.g. link-cells, neighbour lists etc.).
- Algorithms for macromolecules and polymers.
- Algorithms for biological molecules.
- Vectorisation on parallel machines.
- Parallel graphics in molecular simulation.
- Energy minimisation.
- Novel algorithms.

Readers interested in taking part in this workshop should contact Dr. W. Smith or Dr. F. Müller-Plathe, TCS Division, SERC Daresbury Laboratory, Warrington WA4 4AD, or via electronic mail WL@UK.AC.DL.DLGM, or FMP@UK.AC.DL.DLGM or FMP@UK.AC.RL (Bitnet). Please specify your area of work and subject of your talk (if any).

# CCP5 Announcement
# Visit of Prof. K. Binder to the United Kingdom

12 November to 18 November 1989

CCP is pleased to announce the planned visit of Professor Kurt Binder, of the University of Mainz, to the United Kingdom in November 1989 (12th to 18th). Professor Binder is internationally respected for his contributions to simulation physics and we are delighted that he has agreed to accept our invitation.

The purpose of his visit, which is being sponsored by CCP5, is to provide U.K. colleagues, with interests in simulation, the opportunity to hear Professor Binder give topical seminars on simulation and to meet with him and discuss all aspects of simulation.

The planned itinerary for his visit is as follows:

- 13-14 November: Bristol, where he will give a seminar on "Monte Carlo Simulations of Wetting". Interested readers please contact Dr. M.P. Allen at the Department of Physics for local information (MPA @ UK.AC.BRISTOL.PVA).
- 15 November: Daresbury Laboratory, where he will lecture on "Monte Carlo Simulations in Polymer Physics." A workshop on the subject of polymer simulations is also planned for the occasion. Please contact Dr. W. Smith at Daresbury for further information (0925 603257 or WL @ UK.AC.DL.DLGM).
- 16-17 November: Edinburgh University, where he will lecture on "Recent Developments in the Theory of Finite Size Effects on Phase Transitions." Please contact Prof. Stuart Pawley or Dr. David Richards at the Physics Laboratories for further information.

We hope that many of our readers will be able to travel to these venues and take part in meetings organised.

W. Smith, CCP5 24 July 1989.

# PRELIMINARY ANNOUNCEMENT
## A CCP5 meeting on

## Architecture and Algorithms in Condensed Phase Simulations

The Collaborative Computational Project (CCP5) for the computer simulation of condensed phases is planning a meeting to be held in St Andrews, Scotland from 2nd–5th July 1990.

The meeting will consider the different types of computer architecture used in the simulation of condensed phases and protein structures and will try to highlight the advantages and disadvantages of particular architectures for a variety of problems in molecular simulation. We will also consider the algorithms for Monte Carlo, molecular mechanics, molecular dynamics and Brownian dynamics simulations and their implementation on different machines.

The meeting will include sessions on single–instruction–multiple–data computers, multiple–instruction–multiple–data computers, concurrent and pipeline processors (at the mainframes and super minicomputer level), as well as purpose–built simulation machines. There will be a session in which speakers will give detailed accounts of scientific advances resulting directly from the use of these new architectures and a session on the visualization of simulation results. A large poster session and demonstrations by a number of computer manufacturers have also been scheduled.

The meeting should be of interest to researchers in the biological and physical sciences, who use molecular simulation techniques in their work. Further details can be obtained by writing to

Dr. D.J. Tildesley
Department of Chemistry
The University
Southampton S09 5NH
U.K.

The CCP5 Program Library provides programs and documentation free of charge to academic centres upon application to Dr. W. Smith, TCS Division, S.E.R.C. Daresbury Laboratory, Daresbury, Warrington WA4 4AD, U.K.. Listings of programs are available if required but it is recommended that magnetic tapes (supplied by the applicant) be used. It may also be possible to transfer a small number of programs over the JANET network to other computer centres in the U.K.. Please note that use of inappropriate packing for magnetic tapes (e.g. padded bags) may result in them being considered unusable by Daresbury Computing Division and returned without the required software. Please ensure that these forms of packaging are not used. A list of programs available is presented in the following pages.

Readers should also note that we are authorised to supply the example programs originally published in the book "Computer Simulation of Liquids", by M.P. Allen and D.J. Tildesley (Clarendon Press, Oxford 1987). These are supplied in the same manner as the resident CCP5 programs. We are grateful to Mike Allen and Dominic Tildesley for their permission.

We should also like to remind our readers that we would welcome further contributions to the Program Library. The Library exists to provide support for the research efforts of everyone active in computer simulation and to this end we are always pleased to extend the range of software available. If any of our readers have any programs they would like to make available, please would they contact Dr. Smith.

**Please Note: For copyright reasons we are not able to supply the programs CASCADE, SYMLAT, THBFIT,THBPHON and THBREL free of charge to Universities outside the United Kingdom.**

## Program from the Book: "Computer Simulation of Liquids" by M.P. Allen and D. Tildesley, Clarendon Press, Oxford 1987.

These programs originally appeared on microfiche in the book "Computer Simulation of Liquids" by M. P. Allen and D. J. Tildesley, published by Oxford University Press, 1987. They are made freely available to members of CCP5, in the hope that they will be useful. The intention is to clarify points made in the text, rather than to provide a piece of code suitable for direct use in a research application. We ascribe no commercial value to the programs themselves. Although a few complete programs are provided, our aim has been to offer building blocks rather than black boxes. As far as we are aware, the programs work correctly, but we can accept no responsibility for the consequences of any errors, and would be grateful to hear from you if you find any. You should always check out a routine for your particular application. The programs contain some explanatory comments, and are written, in the main, in FORTRAN-77. One or two routines are written in BASIC, for use on microcomputers. In the absence of any universally agreed standard for BASIC, we have chosen a very rudimentary dialect. These programs have been run on an Acorn model B computer. Hopefully the translation of these programs into more sophisticated languages such as PASCAL or C should not be difficult.

## THE CCP5 PROGRAM LIBRARY.

| | | |
|---|---|---|
| ADMIXT | [MD,LJA/MIX,LF,TH+MSD+RDF] | W. Smith |
| CARLOS | [MC,VS+Aquo,TH] | B. Jonsson<br>S. Romano |
| CARLAN | [DA,CARLOS structure analysis] | B. Jonsson<br>S. Romano |
| CASCADE | [LS,DIL,EM,TH+STR] | M. Leslie/<br>W. Smith |
| CURDEN | [DA,Current Density Correlations] | W. Smith |
| DENCOR | [DA,Density Correlations] | W. Smith |
| HLJ1 | [MD,LJA,LF,TH+MSD+RDF] | D.M. Heyes |
| HLJ2 | [MD,LJA,LF,TH+MSD+RDF+VACF] | D.M. Heyes |
| HLJ3 | [MD,LJA,LF/LC,TH+MSD+RDF] | D.M. Heyes |
| HLJ4 | [MD,LJA,LF/CP+CT,TH+MSD+RDF] | D.M. Heyes |
| HLJ5 | [MD,LJA/SF,LF,TH+MSD+RDF] | D.M. Heyes |
| HLJ6 | [MD,LJA,TA,TH+MSD+RDF] | D.M. Heyes |
| HMDIAT | [MD,LJD,G5+Q4,TH+MSD+QC] | S.M. Thompson |
| HSTOCH | [MD/SD,VS+BA,LF+CA,TH] | W.F. van Gunsteren/<br>D.M. Heyes |
| MCN | [MC,LJA,TH] | N. Corbin |
| MCLSU | [MC,LJA,TH] | C.P. Williams/<br>S. Gupta |
| MCMOLDYN | [MD/MC,LJS+FC+AQ,<br>LF+QF/G5+QS,TH+RDF] | A. Laaksonen |
| MCRPM | [MC,RPE,TH+RDF] | D.M. Heyes |
| MDATOM | [MD,LJA,G5,TH+RDF+MSD+QC] | S.M. Thompson |
| MDATOM | [MD,LJA,LF,TH+MSD+RDF] | D. Fincham |
| MDCSPC4B | [PRMD,BHM+FC,G5+G4,TH+STF+RDF] | W. Smith |
| MDDIAT | [MD,LJD,LF+CA,TH+MSD] | D. Fincham |
| MDDIATQ | [MD,LJD+PQ,LF+CA,TH+MSD] | D. Fincham |
| MDIONS | [MD,BHM,LF,TH+MSD+RDF+STF] | D. Fincham/<br>N. Anastasiou |
| MDLIN | [MD,LJL,G5+Q4,TH+MSD+QC] | S.M. Thompson. |
| MDLINQ | [MD,LJL+PQ,G5+Q4,TH+MSD+QC] | S.M. Thompson |
| MDMANY | [MD,LJS+FC,LF+QF,TH] | D. Fincham/<br>W. Smith |
| MDMIXT | [MD,LJS/MIX,LF+QF,TH] | W. Smith |
| MDMPOL | [MD,LJS+FC/MIX,LF+QF,TH] | W. Smith/<br>D. Fincham |
| MDNACL | [MD,BHM,LF,TH+MSD+RDF] | W. Smith |
| MDPOLY | [MD,LJS,G5+Q4,TH+MSD+QC] | S.M. Thompson |
| MDMULP | [MD,LJS+PD+PQ/MIX,LF+QF,TH] | W. Smith |
| MDSGWP | [MD,LJA/SGWP,LF,TH+VACF+RDF+QC] | W. Smith/<br>K. Singer |
| MDTETRA | [MD,LJT,G5+Q4,TH+MSD+QC] | S.M. Thompson |
| MDZOID | [MD,GAU,LF+QF,TH+MSD+RDF+VACF] | W. Smith |

| | | |
|---|---|---|
| NAMELIST | [UT, Namelist emulation] | K. Refson |
| PIMCLJ | [PIMC,LJA,MC,TH+RDF+QC] | K. Singer<br>W. Smith |
| SCN | [MC,LJA,RFD,TH] | N. Corbin |
| SURF | [MD,BHM/TF/2D,LF,TH+RDF] | D.M. Heyes |
| SYMLAT | [LS,PIL,EM+SYM,TH+STR] | Harwell |
| THBFIT | [LS,PIL,EM,Potential fitting] | Harwell |
| THBPHON | [LS,PIL/3B,EM,Phonon dispersion] | Harwell |
| THBREL | [LS,PIL,EM,TH+STR] | Harwell |

**Key:**

| | | |
|---|---|---|
| Program types: | MD | Molecular dynamics |
| | MC | Monte Carlo |
| | PRMD | Parrinello-Rahman MD |
| | LS | Lattice simulations |
| | SD | Stochastic dynamics |
| | DA | Data analysis |
| | UT | Utility package |
| | PIMC | Path Integral Monte Carlo |
| System models: | LJA | Lennard-Jones atoms |
| | LJD | Lennard-Jones diatomic molecules |
| | LJL | Lennard-Jones linear molecules |
| | LJT | Lennard-Jones tetrahedral molecules |
| | LJS | Lennard-Jones site molecules |
| | RPE | Restricted primitive electrolyte |
| | BHM | Born-Huggins-Meyer ionics |
| | SGWP | Spherical gaussian wavepackets |
| | TF | Tosi-Fumi ionics |
| | VS | Variable site-site model |
| | BA | Bond angle model |
| | PD | Point dipole model |
| | PQ | Point quadrupole model |
| | MIX | Mixtures of molecules |
| | GAU | Gaussian molecule model |
| | FC | Fractional charge model |
| | PIL | Perfect ionic lattice model |
| | DIL | Defective ionic lattice model |
| | 3B | 3-body force model |
| | 2D | Two dimensional simulation |
| | SF | Shifted force potential |
| | FC | Fractional charge model |
| | AQ | Aqueous solutions |
| Algorithm: | G5 | Gear 5th order predictor-corrector |
| | Q4 | Quaternion plus 4th. order Gear P-C. |

| | | |
|---|---|---|
| | LF | Leapfrog (Verlet) |
| | QF | Fincham Quaternion algorithm |
| | QS | Sonnenschein Quaternion algorithm |
| | LC | Link-cells MD algorithm |
| | CP | Constant pressure |
| | CT | Constant temperature |
| | TA | Toxvaerd MD algorithm |
| | CA | Constraint algorithm |
| | EM | Energy minimisation |
| | SYM | Symmetry adapted algorithm |
| | RFD | Rossky-Friedman-Doll algorithm |
| Properties: | TH | Thermodynamic properties |
| | MSD | Mean-square-displacement |
| | RDF | Radial distribution function |
| | STF | Structure factor |
| | VACF | Velocity autocorrelation function |
| | QC | Quantum corrections |
| | STR | Lattice stresses |

## Programs from the Book "Computer Simulation of Liquids"

| | |
|---|---|
| F.1 | Periodic boundary conditions in various geometries |
| F.2 | 5-value Gear predictor-corrector algorithm |
| F.3 | Low-storage MD programs using leapfrog Verlet algorithm |
| F.4 | Velocity version of Verlet algorithm |
| F.5 | Quaternion parameter predictor-corrector algorithm |
| F.6 | Leapfrog algorithms for rotational motion |
| F.7 | Constraint dynamics for a nonlinear triatomic molecule |
| F.8 | Shake algorithm for constraint dynamics of a chain molecule |
| F.9 | Rattle algorithm for constraint dynamics of a chain molecule |
| F.10 | Hard sphere molecular dynamics program |
| F.11 | Constant-$NVT$ Monte Carlo for Lennard-Jones atoms |
| F.12 | Constant-$NPT$ Monte Carlo algorithm |
| F.13 | The heart of a constant $\mu VT$ Monte Carlo program |
| F.14 | Algorithm to handle indices in constant $\mu VT$ Monte Carlo |
| F.15 | Routines to randomly rotate molecules |
| F.16 | Hard dumb-bell Monte Carlo program |
| F.17 | A simple Lennard-Jones force routine |
| F.18 | Algorithm for avoiding the square root operation |
| F.19 | The Verlet neighbour list |
| F.20 | Routines to construct and use cell linked-list method |
| F.21 | Multiple timestep molecular dynamics |
| F.22 | Routines to perform the Ewald sum |
| F.23 | Routine to set up alpha fcc lattice of linear molecules |
| F.24 | Initial velocity distribution |
| F.25 | Routine to calculate translational order parameter |
| F.26 | Routines to fold/unfold trajectories in periodic boundaries |
| F.27 | Program to compute time correlation functions |
| F.28 | Constant-$NVT$ molecular dynamics - extended system method |
| F.29 | Constant-$NVT$ molecular dynamics - constraint method |
| F.30 | Constant-$NPH$ molecular dynamics - extended system method |
| F.31 | Constant-$NPT$ molecular dynamics - constraint method |
| F.32 | Cell linked-lists in sheared boundaries |
| F.33 | Brownian dynamics for a Lennard-Jones fluid |
| F.34 | An efficient clustering routine |
| F.35 | The Voronoi construction in 2d and 3d |
| F.36 | Monte Carlo simulation of hard lines in 2d |
| F.37 | Routines to calculate Fourier transforms |

# Report on the CCP5 Symposium on: "NOVEL METHODS IN MOLECULAR SIMULATION" held at Royal Holloway and Bedford New College, Egham, Surrey TW20 OEX on the 3rd and 4th of July 1989.

D.M. Heyes

September 6, 1989

This summer saw another successful CCP5 symposium held at this picturesque campus of London University. It brought together simulators working in a number of new fields of microscopic simulation. The meeting started with a talk on "Neural Networks: New Tools for the Computer Simulator", by Prof. R. Cotterill (University of Lyngby, Denmark), who reviewed the field of neural networks, in particular the potential and current problems. A seminar on "Simulations of Solids and Liquids using Cellular Automata", was then given by Dr. M. P. Allen (University of Bristol). He talked about *new* advances in producing useful dynamics out of spin lattice simulations. Both of these talks aroused much vigorous discussion. The second of our invited speakers from abroad, Dr. J. Naudts (University of Antwerp, Belgium) presented a paper on "Lattice Simulations of Percolation and Emulsions". The lattice gas cells were studied under the influence of electric field. He described some fascinating new insights into the universal properties of dynamic percolation. Another welcome visitor from abroad, Dr. G.S. Grest (Exxon, NJ, U.S.A.), reviewed some new major simulations of macromolecules. He showed strong evidence for $t^{1/2}$ and $t^{1/4}$ regimes of dynamical relaxation. Reptation and other co-operative chain motions were clearly seen from the extensive simulation results presented.

The post-prandial evening round-table discussion was centred around perceived major challenges in simulation. This produced a lively wide-ranging debate on the future of simulations of biological molecules, transport coefficients and quantum effects by direct simulation.

The second day commenced with a seminar on "Car-Parrinello Simulations", from Professor M.J. Gillan (University of Keele). The technique and some recent applications were described. Dr. R. McGreevey (University of Oxford) talked about a new $MC$ method employing scattering structure factors to produce intermolecular pair potentials in a talk entitled "Reverse Monte Carlo Simulation". Molten salt test compounds were treated in some detail. The Monte Carlo theme was continued by Dr. D.J. Tildesley (University of Southampton) who talked about "Gibbs ensemble MC" and its applications for determining the phase boundaries of multicomponent fluids. Efficiently determining the critical point of spin systems is frequently prevented by critical slowing down. Dr. D. Nicolaides (University of Bristol) in a seminar entitled, "Multi-move MC and Ising calculations", reviewed some new techniques for travelling quickly through phase space in the vicinity of the critical point. Prof. J.G. Powles (University of Kent)

lectured on "Information Theory and Fractal Methods". He talked about the shape of the trajectory of single particle motion in a simple fluid. Line shape fitting using information theory was described. "Molecular Dynamics from a New Angle", was considered by Prof. R. Cotterill (University of Lyngby, Denmark) in the second of his presentations. He introduced the possiblility of carrying out multimove $MD$ along a hypersurface which would enable real time to be spanned many orders of magnitude faster than at present. Last but not least was a presentation on "A Direct Method for Studying Reaction Rates by Equilibrium MD", by Dr. D. Brown (U.M.I.S.T., Manchester). He described a non-intrusive method for determining rate constants and mechanisms for chemical reactions by calculating concentration correlation functions.

# Two for the Price of One! The Hidden Capacity of the Complex Fast Fourier Transform

W. Smith

July 11, 1989

## Introduction

The power and versatility of the complex fast Fourier transform is well known (and indeed, has been outlined in this newsletter previously [1, 2]). The purpose of this article is to point out yet another aspect of it, which leads to even greater efficiency in computational applications. It concerns the situation in which the function to be Fourier transformed is *real*; a situation that happily arises frequently in molecular dynamics applications. To see how this can be exploited however, we must do a little mathematics. Readers interested in a more thorough account should consult reference [3].

## Basic Fourier Transform Properties

The standard form of the Fourier transform relates a function $h(t)$ with another function $H(f)$ through an integral:

$$H(f) = \int h(t)\ exp(-2\pi i f t)\ dt \tag{1}$$

and under reasonable circumstances, this transform has an inverse :

$$h(t) = \int H(f)\ exp(2\pi i f t)\ df \tag{2}$$

in which $f$ and $t$ (frequency and time - or some other suitable conjugate variables) define the *domains* of the two functions.

An important property possessed by many functions is that they may be described as *even* or *odd*. This simply means that if we change the sign of the argument, an even function returns the same value, while an odd function returns the negative of the original value. i.e.

$$h(t) = h(-t)$$

for an even function, and

$$h(t) = -h(-t)$$

for an odd function. In general functions are neither even nor odd, but can be expressed as the sum of an even and an odd function i.e.

$$h(t) = e(t) + o(t)$$

with

$$e(t) = \frac{1}{2}(h(t) + h(-t))$$

and

$$o(t) = \frac{1}{2}(h(t) - h(-t)).$$

Products of functions can also be even or odd. The product of two even functions or of two odd functions is even, while the cross product is odd. These properties are largely self evident. Equally self evident is the fact that a definite integral of an odd function over an interval $[-\alpha, +\alpha]$ *must* be zero, while the corresponding integral of an even function may be nonzero.

These simple properties are very useful when employed in conjunction with the Fourier transform. Most importantly, for the purposes of this note, it can be shown that when $h(t)$ is a real function, the Fourier transform $H(f)$, which in general is *complex*, has a real part that is *even* in $f$ and an imaginary part that is *odd* in $f$. This is seen most readily in the inverse Fourier transform (Eq. 2) of $H(f)$ back to $h(t)$. The $exp(i2\pi ft)$ term may be split into an even function $cos(2\pi ft)$ and an odd function $i\ sin(2\pi ft)$ through Euler's relation. It follows simply, from what has been said above, that the real part of $H(f)$ has to be even and the imaginary part has to be odd, if the final expression of $h(t)$ is to be real.

This property becomes even more interesting if the function $h(t)$ is constructed to be a complex sum of two *real* functions (say $a(t)$ and $b(t)$) in the following way:

$$h(t) = a(t) + i\ b(t). \tag{3}$$

It then turns out that if this function is Fourier transformed, it is possible to extract *both* the Fourier transforms of $a(t)$ and $b(t)$ from the result with very little effort. Effectively this means that we can Fourier transform the two real functions $a(t)$ and $b(t)$ *at the same time*. This useful property merits a closer examination.

## Fourier Transforming Real Functions

Let $h(t)$ be defined as in equation (3), and let its Fourier transform be $H(f)$. Let the functions $a(t)$ and $b(t)$ have Fourier transforms $A(f)$ and $B(f)$ respectively. We shall distinguish between the real and imaginary parts of the complex functions by the use of single and double dashes, *i.e.*:

$$H(f) = H'(f) + i\ H''(f)$$

where H'(f) is the real part and H"(f) the imaginary part.

Since equation (3) represents a linear combination of two functions, we can write directly:

$$H(f) = A(f) + i\ B(f).$$

(This simple equation is deceptive; $A(f)$ and $B(f)$ are complex!) Separating out the real and imaginary parts of all these functions gives:

$$H'(f) = A'(f) - B''(f) \tag{4}$$

$$H''(f) = A''(f) + B'(f). \tag{5}$$

```
      PARAMETER (N=2**integer)
      DIMENSION A(N),B(N),AI(N),BI(N)
      COMPLEX   H(N)
C
C     LOAD COMPLEX ARRAY H(N) WITH REAL A(N) AND B(N)
      DO 100 I=1,N
  100 H(I)=CMPLX(A(I),B(I))
C
C     CALL SYSTEM FAST FOURIER TRANSFORM
      CALL FFT(N,H,plus system specific parameters)
C
C     EXTRACT FOURIER TRANSFORMS OF A AND B
      A(1)=REAL(H(1))
      AI(1)=0.0
      B(1)=AIMAG(H(1))
      BI(1)=0.0
      DO 200 I=2,N
      A(I)  = 0.5*( REAL(H(I)) +  REAL(H(N+2-I)))
      AI(I) = 0.5*(AIMAG(H(I)) - AIMAG(H(N+2-I)))
      B(I)  = 0.5*(AIMAG(H(I)) + AIMAG(H(N+2-I)))
      BI(I) = 0.5*(-REAL(H(I)) +  REAL(H(N+2-I)))
  200 CONTINUE
      .  .  .  .
      etc.
```

Figure 1. FORTRAN Code for Fourier Transform of Two Real Functions Simultaneously

---

Also, for negative values of frequency we can write:

$$H'(-f) = A'(-f) - B''(-f) \tag{6}$$
$$H''(-f) = A''(-f) + B'(-f) \tag{7}$$

We can now use what we know about the real and imaginary parts of the Fourier transforms $A(f)$ and $B(f)$, namely that the real parts are even and the imaginary parts are odd, to rewrite (6) and (7) as:

$$H'(-f) = A'(f) + B''(f) \tag{8}$$
$$H''(-f) = -A''(f) + B'(f) \tag{9}$$

Lastly, combining equations (4),(5),(8) and (9), we obtain

$$\begin{aligned} A'(f) &= \frac{1}{2}(H'(f) + H'(-f)) \\ A''(f) &= \frac{1}{2}(H''(f) - H''(-f)) \\ B'(f) &= \frac{1}{2}(H''(f) + H''(-f)) \\ B''(f) &= \frac{1}{2}(-H'(f) + H'(-f)) \end{aligned} \tag{10}$$

These equations clearly show that, once the components of $H(f)$ are obtained by Fourier transforming $h(t)$, we can easily calculate the components of $A(f)$ and $B(f)$, which constitute the Fourier transforms of $a(t)$ and $b(t)$.

The realisation of this method in rude FORTRAN is particularly simple, and is presented in Figure 1. The complex array $H(N)$ is constructed from the two real arrays $A(N)$ and $B(N)$. A standard fast Fourier transform (FFT) routine (every computing system has one!) calculates the discrete Fourier transform. The real parts of the Fourier transforms of $A$ and $B$ overwrite the original arrays. The imaginary parts are placed in arrays AI(N) and BI(N) respectively.

Applications of this trick are obviously many, but a particularly nice application might be the method of Osguthorpe *et al* [4], which Fourier transforms of the trajectories of atoms in MD simulations of complex molecules. The purpose is to 'project out' particular frequencies (for example, those corresponding to particular modes of vibration) prior to displaying the trajectories in moving graphics. The effect of this technique is visually stunning and greatly reduces the 'visual chaos' effect of conventional MD movies. It is apparent here that long Fourier transforms of many trajectories are required, and any trick that reduces the computational cost is beneficial. (Indeed, it is not even necessary in this application to extract the final Fourier transforms since the data can be filtered and inverse transformed without needing to do this.)

## Convolution and Correlation

A well known application of the fast Fourier transform is to speed up the calculation of convolution and correlation integrals [2, 3]. Since in MD we are usually confronted with real data, we may ask if the above trick can be exploited here also, to permit the calculation of (say) two convolution integrals at the same time. This is indeed the case, though the algorithm is not so easily described. I shall attempt to outline the method for the calculation of a convolution integral, the corresponding treatment for correlation integrals is very similar.

The standard form of a convolution integral is

$$p(t) = \int a(\tau)b(t-\tau)\,d\tau \tag{11}$$

and the equivalent expression in the frequency domain is [3]

$$P(f) = A(f)B(f) \tag{12}$$

where $A(f)$, $B(f)$ and $P(f)$ are the Fourier transforms of $a(t)$, $b(t)$ and $p(t)$ respectively. The simplicity of this expression, coupled with the computational efficiency of the FFT is what makes the FFT method of computing convolution integrals so attractive.

Now if $a(t)$ and $b(t)$ are real functions, and we introduce two additional real functions $c(t)$ and $d(t)$, with convolution $q(t)$, we can calculate both $p(t)$ and $q(t)$ at the same time in the following manner.

Define two complex functions $g(t)$ and $h(t)$ as follows

$$g(t) = a(t) + i\,c(t)$$

$$h(t) = b(t) + i\,d(t).$$

If we Fourier transform $g(t)$ and $h(t)$ we get $G(f)$ and $H(f)$ respectively, whose real and imaginary parts we represent in the usual notation

$$G(f) = G'(f) + i\, G''(f)$$

$$H(f) = H'(f) + i\, H''(f).$$

Following the above account of the simultaneous Fourier transform of two real functions we can write immediately (c.f equation (10))

$$\begin{aligned}
A'(f) &= \frac{1}{2}(G'(f) + G'(-f)) \\
A''(f) &= \frac{1}{2}(G''(f) - G''(-f)) \\
C'(f) &= \frac{1}{2}(G''(f) + G''(-f)) \\
C''(f) &= \frac{1}{2}(-G'(f) + G'(-f)) \\
B'(f) &= \frac{1}{2}(H'(f) + H'(-f)) \\
B''(f) &= \frac{1}{2}(H''(f) - H''(-f)) \\
D'(f) &= \frac{1}{2}(H''(f) + H''(-f)) \\
D''(f) &= \frac{1}{2}(-H'(f) + H'(-f))
\end{aligned} \tag{13}$$

The function products we require in the frequency domain are

$$P(f) = A(f)B(f)$$

and

$$Q(f) = C(f)D(f).$$

The products on the right of these equations can be expanded into

$$A(f)B(f) = (A'(f)B'(f) - A''(f)B''(f)) + i\,(A'(f)B''(f) + A''(f)B'(f))$$

and

$$C(f)D(f) = (C'(f)D'(f) - C''(f)D''(f)) + i\,(C'(f)D''(f) + C''(f)D'(f))$$

from which it is obvious that

$$\begin{aligned}
P'(f) &= A'(f)B'(f) - A''(f)B''(f) \\
P''(f) &= A'(f)B''(f) + A''(f)B'(f) \\
Q'(f) &= C'(f)D'(f) - C''(f)D''(f) \\
Q''(f) &= C'(f)D''(f) + C''(f)D'(f)
\end{aligned} \tag{14}$$

Clearly, by way of the equations (13), all these components may be calculated from the components of $G(f)$ and $H(f)$.

```
      PARAMETER (N=2**Integer)
      DIMENSION A(N),B(N),C(N),D(N),P(N),Q(N)
      COMPLEX   G(N),H(N),Z(N)
      RNORM=1.0/FLOAT(N)
C     CONSTRUCT COMPLEX ARRAYS G(N) AND H(N)
      DO 100 I=1,N
      G(I)=CMPLX(A(I),C(I))
      H(I)=CMPLX(B(I),D(I))
  100 CONTINUE
C     CALCULATE FOURIER TRANSFORMS OF G(N) AND H(N)
      CALL FFT(N,G,plus system specific parameters)
      CALL FFT(N,H,plus system specific parameters)
C     CONSTRUCT COMPLEX Z(N) ARRAY
      Z1=REAL(G(I))*REAL(H(I))
      Z2=AIMAG(G(I))*AIMAG(H(I))
      Z(1)=RNORM*CMPLX(Z1,Z2)
      DO 200 I=1,N/2
      AI1=0.5*(REAL(G(I+1))+REAL(G(N+1-I)))
      AI2=0.5*(AIMAG(G(I+1))-AIMAG(G(N+1-I)))
      CI1=0.5*(AIMAG(G(I+1))+AIMAG(G(N+1-I)))
      CI2=0.5*(-REAL(G(I+1))+REAL(G(N+1-I)))
      BI1=0.5*(REAL(H(I+1))+REAL(H(N+1-I)))
      BI2=0.5*(AIMAG(H(I+1))-AIMAG(H(N+1-I)))
      DI1=0.5*(AIMAG(H(I+1))+AIMAG(H(N+1-I)))
      DI2=0.5*(-REAL(H(I+1))+REAL(H(N+1-I)))
      PI1=RNORM*(AI1*BI1-AI2*BI2)
      PI2=RNORM*(AI1*BI2+AI2*BI1)
      QI1=RNORM*(CI1*DI1-CI2*DI2)
      QI2=RNORM*(CI1*DI2+CI2*DI1)
      Z(I+1)=CMPLX(PI1-QI2,PI2+QI1)
      Z(N+1-I)=CMPLX(PI1+QI2,QI1-PI2)
  200 CONTINUE
C     INVERSE FOURIER TRANSFORM ARRAY Z(N)
      CALL FFT(N,Z,plus system specific parameters)
C     EXTRACT CONVOLUTION ARRAYS
      DO 300 I=1,N
      P(I)= REAL(Z(I))
      Q(I)=AIMAG(Z(I))
  300 CONTINUE
      .
      .
      .
      etc.
```

Figure 2. FORTRAN Code for Calculation of Two Real Convolution Integrals Simultaneously.

This however is only part of the story. The product obtained in the frequency domain must now be inverse Fourier transformed to obtain the final convolution. That is we

must obtain $p(t)$ and $q(t)$ from $P(f)$ and $Q(f)$. Since we know that $p(t)$ and $q(t)$ are real functions, it would be nice to obtain our result in an economical and straightforward way, using the lessons we have learned already. We can do this as follows.

Define a complex function $z(t)$ such that

$$z(t) = p(t) + i\, q(t) \tag{15}$$

We now say this has a Fourier transform $Z(f)$ with components $Z'(f)$ and $Z''(f)$. As before, these components are related to the Fourier transforms of $p(t)$ and $q(t)$, namely $P(f)$ and $Q(f)$:

$$\begin{aligned} P'(f) &= \frac{1}{2}(Z'(f) + Z'(-f)) \\ P''(f) &= \frac{1}{2}(Z''(f) - Z''(-f)) \\ Q'(f) &= \frac{1}{2}(Z''(f) + Z''(-f)) \\ Q''(f) &= \frac{1}{2}(-Z'(f) + Z'(-f)). \end{aligned} \tag{16}$$

We may rearrange these equations to give

$$\begin{aligned} Z'(f) &= P'(f) - Q''(f) \\ Z''(f) &= P''(f) + Q'(f) \\ Z'(-f) &= P'(f) + Q''(f) \\ Z''(-f) &= -P''(f) + Q'(f). \end{aligned} \tag{17}$$

These final equations tell us how to construct the function $Z(f)$ in the frequency domain so that on inverse Fourier transforming it to $z(t)$ we find the convolution functions $p(t)$ and $q(t)$ in the real and imaginary parts respectively, of the function $z(t)$.

This completes the description of the method. Its implementation in FORTRAN appears in Figure 2, where the variable names are the same as in the above text. The variable RNORM is a factor which correctly renormalises the functions after inverse Fourier transforming them. The reader may like to derive the corresponding algorithm for correlation functions, but beware the product of Fourier transforms in the frequency domain, since one of the functions must now be a complex conjugate.

This algorithm also has obvious applications in MD. Less well known is its application in quantum simulations [5], about which we hope to say more at a later date!

## References


[1] W. Smith, CCP5 Info. Quart. **5** 34 (1982).

[2] W. Smith, CCP5 Info. Quart. **7** 12 (1984).

[3] E.O. Brigham, *The Fast Fourier Transform and its Applications*, Prentice Hall, NJ 1988.

[4] D.J. Osguthorpe, Bath NATO Summer School, September 1988, "*Fluids, Polymers and Solids*".

[5] K. Singer and W. Smith, paper in preparation (1989).

# Benchmark Timings of an MD code on a Variety of Computers

by

David Brown[†] , Florian Müller-Plathe[‡] and Julian H. R. Clarke[†]

Within the last year we have been fortunate enough to have had access to a number of graphics workstations. By necessity these machines have to have significant computing capacity to be able to handle real time rotations etc. In order to assess the potential for molecular dynamics simulations we have evaluated the CPU performances using a FORTRAN MD code as a benchmark. MD programs do not vectorise particularly well so they provide a far more useful benchmark in this case than say the standard ones like LINPACK and Livermore Loops. The assessment has since spread rapidly to encompass a broad spectrum of computers from micros right through to 'supers'- a process aided by distributing the program to interested third parties for running on their own computing facilities. The results of all this frenzied activity is the table we present at the end. In glancing back over the 30 or so previous issues of the CCP5 Newsletter it is surprising that only one previous article [1] has presented anything similar and that was nearly seven years ago. It is perhaps timely and appropriate then that the results of our survey be disseminated through the pages of this journal.

Our philosophy in doing these benchmarks has been largely to assess the "readily available" power of these, in certain cases multiprocessor, machines. By this we mean that in all cases the code has at least been run without any changes being made to it using the FORTRAN compiler as supplied by the manufacturer (and thus, one would hope, optimized for that particular machine). We present here just the raw data and make no recommendations. Of course there are other factors involved and we leave you to draw your own conclusions!

We have assessed the effect of various optimization levels, where appropriate, which can lead to a certain amount of default vectorisation or even parallelization. **In almost all cases the compile options chosen resulted in only one processor being used.** In a few cases extra timings have been obtained for codes where changes have been made to facilitate vectorisation. No attempts have been made to parallelize code for machines such as the Silicon Graphics (SG) 4D/240, for example, which has four processors. This type of machine is quite capable of running a separate job on each processor so one could argue that its "readily available" power is larger by a factor of the number of CPUs. However, this does not affect the actual elapsed time or "wall time" of any one particular job, i.e. the time that the user perceives to have passed whilst running the compiled code given that the machine is devoted to the one task. These are the timings we are concerned with and report in this article.

The FORTRAN code we have chosen to use as a benchmark is a program entitled "Froggy" [2] which performs a molecular dynamics simulation of 108 Lennard-Jones atoms. The code has been adapted in some respects to make it self contained i.e. it requires no extra libraries, such as NAG etc., for it to run. As is always the case, the main kernel is the routine to calculate the interatomic forces. This is done in the simplest way using a double DO loop. We are fully aware of the many possible alternative ways of speeding up this process (neighbour tables, link lists, systolic loops etc.) none of these are in any way relevant to what we are trying to achieve here. The timings we report are for runs of 1000 steps of the program under conditions of constant energy at the state point ($\rho^*$=0.7,

† Chemistry Department, UMIST, Sackville Street, Manchester, M60 1QD, U.K.

‡ SERC, Daresbury Laboratory, Warrington, WA4 4AD, U.K.

$T^*$=0.3). The interaction potential was truncated at a radius of 2.8$\sigma$ and the reduced timestep used was 0.01. Apart from the forces the only other properties calculated within the double loop are the energy and the virial. In all respects "Froggy" is what might be termed a typical MD program and as such the timings reported should, therefore, be useful to other people even if they run their own code using the input parameters given.

As a benchmark MD code is a reasonable test of comparative power. We have found it a far more tangible guide to performance than say MIPs or MFLOPs. Indeed we encountered one example of a machine which was out performed by a factor of ~3 as compared to a computer of a different make but of comparable specification.

As is well known the particle trajectories generated by different types of computers, starting from the same identical initial conditions, will diverge if, as is highly likely, the arithmetic is done in different ways. On multiprocessor machines MD becomes a particularly demanding test of the abilities of nominally equivalent processors to produce exactly the same answers. We have experienced rogue chips which produce what appear to be sensible results in MD terms for the conservation of energy, say, yet are at odds with the consensus of other "identical" CPUs! In an age where parallel processing is the flavour of the month our experience has taught us that it is imperative to test each individual processing unit carefully. In this respect "Froggy" has been invaluable.

[1] D. Fincham and D. Heyes, CCP5 Newsletter, **7**, 33 (1982).

[2] The original version of the code was taken from the textbook of Allen and Tildesley, *Computer Simulation of Liquids*, Clarendon Press,Oxford (1987). Anyone wishing to run the benchmark on their own computers **and** has access to the JANet electronic mail network can contact David Brown (MCDAPDB@UK.AC.MCC.CMS) who will send you a copy of the program, an input configuration and control data set and a set of specimen results free of charge. Please do not send magnetic tapes or floppy discs or any other transfer medium without first contacting DB.

Various timings for 1000 time steps of benchmark MD code "FROGGY".

| Machine | Opt. Level / Comment | Precision | Time/s |
|---|---|---|---|
| Amdahl 5890 | O3 | 4 byte | 17 |
| | O3 | 8 byte | 20 |
| Amdahl VP1200 | scalar | 4 byte | 18.7 |
| | vector[a] | 4 byte | 14.6 |
| | vector tuned[b] | 4 byte | 2.16 |
| | vector tuned[b] | 8 byte | 2.18 |
| Amdahl VP1100 | scalar | 4 byte | 25.1 |
| | vector tuned[b] | 4 byte | 3.13 |
| | vector tuned[b] | 8 byte | 3.16 |
| Apollo DN 10000 | O | 4 byte | 62.6 |
| | O | 8 byte | 59.0 |
| Ardent Titan-1 | O2 | 4 byte | 84.1 |
| | O2 | 8 byte | 85.4 |
| Compaq 386-20 | - | 8 byte | 1821 |
| Convex C2-10 | O2 | 4 byte | 40.5 |
| | O2 | 8 byte | 36.6 |
| Cray X-MP48 | - | 8 byte | 11 |
| Cray X-MP28 | scalar; CFT compiler | 8 byte | 10.8 |
| | vector tuned; CFT compiler | 8 byte | 4.2 |
| | vector tuned; CFT77 compiler | 8 byte | 9.3 |
| FPS M64/60[c] | O3 and O4 | 8 byte | 30.3 |
| IBM 3090 | - | 4 byte | 20 |
| | - | 8 byte | 21 |
| INTEL iPSC/2 | (SRM host) scalar | 4 byte | 2111 |
| | " | 8 byte | 3894 |
| | (one node) SX | 4 byte | 491 |
| | " | 8 byte | 1005 |
| Mac II | - | 4 byte | 2114 |
| | - | 8 byte | 2116 |
| Mac SE 30 | - | 4 byte | 1520 |
| Meiko | one T800 transputer | 4 byte | 249 |
| | " | 8 byte | 340 |
| SG IRIS 4D/50 | O3 | 4 byte | 205 |
| | O3 | 8 byte | 250 |
| SG IRIS 4D/120 | O3 | 4 byte | 99 |
| | O3 | 8 byte | 129 |
| SG IRIS 4D/240 GTX | O3 | 4 byte | 59 |
| | O3 | 8 byte | 71 |
| Stellar GS1000 | O1 and O2 | 4 byte | 275.9[d] |
| | O3 | 4 byte | 281.5[d] |
| | - | 4 byte | 46.0[e] |
| Stellar GS2000 | O2 and O3 | 4 byte | 247 |
| | O3 | 8 byte | 258 |
| Sun 4/110M[f] | O3 | 4 byte | 570 |
| | O3 | 8 byte | 605 |
| Sun 3/60 | O3 | 4 byte | 2194 |
| | O3 | 8 byte | 2340 |

[a] This timing for the VP1200 was obtained by just switching on the vector option i.e. default vectorisation.

[b] One line compiler directives (*VOCL statements) were inserted to optimize the vectorisation of the inner of the force evaluation double loop. This consisted of telling the compiler the upper limit of the length of the inner loop, which variables within the inner loop are temporary (i.e. their value is not required outside of the loop) and how often the IF statement, determining the interaction range of particles, is satisfied (easily predetermined from the density and the potential truncation radius).

[c] The FPS M64/60 minimum precision is 8 byte.

[d] Timings for original code,

[e] Timings obtained by Stellar personnel on adapted code.

[f] These timings for the Sun4 were obtained on a machine containing a floating point unit known not to perform to specifications under high processing loads. Timings obtained using a revised version of the fpu were the same.

Data for the Apollo, Ardent, Convex, Cray X-MP48, IBM, Stellar GS1000 and Sun 3 were obtained by Florian Müller-Plathe (Daresbury Laboratory).
Data for Macintoshes courtesy of Martin Whittle (Chemistry Dept.,Manchester University). (Both Macs used math co-processors)
Data for Cray X-MP28 and Compaq were obtained by Hawthorne Davis (DuPont).
Data for INTEL courtesy of Bill Smith (Daresbury Laboratory).
Data for T800 transputer courtesy of Steve Liem (UMIST).
Data for the FPS and the Stellar GS2000 were obtained by Paul Mitchell (UMIST).
All the other timings were carried out by David Brown (UMIST).

# SPECTRAL ANALYSIS OF MD SIMULATIONS

D. W. Noid,* G. A. Pfeffer,† B. G. Sumpter,** and S. K. Gray‡

August 21, 1989

*Chemistry Division, Oak Ridge National Laboratory, Oak Ridge, TN 37831-6182. Research sponsored by the Division of Materials Sciences, Office of Basic Energy Sciences, U.S. Department of Energy, under contract DE-AC05-840R21400 with Martin Marietta Energy Systems, Inc.

†Department of Chemistry, University of Nebraska at Omaha, Omaha, NE 68181-0109. Partial support by the University of Nebraska at Omaha Committee on Research is acknowledged.

**Department of Chemistry, University of Tennessee, Knoxville, TN 37996-1600. Research sponsored by the Division of Materials Sciences, Office of Basic Energy Sciences, U.S. Department of Energy, under contract DE-AC05-840R21400 with Martin Marietta Energy Systems, Inc.

‡Department of Chemistry, Northern Illinois University, DeKalb, IL. Research sponsored by the National Science Foundation, CHE-8822739, and the donors of the Petroleum Research Fund administered by the American Chemical Society.

## Introduction

Spectroscopy is a valuable tool in various fields of chemistry and physics. It is especially useful for characterizing conformations of polymer chains and providing the frequency distribution for heat capacities of polymer crystals. Recently we have applied a method to calculate spectral frequencies from molecular dynamics (MD) simulation which differs from the standard Fast Fourier Transform (FFT) method.

Most calculations of spectral frequencies are based on normal coordinate analysis, which assumes a collection of harmonic oscillators as the polymer model.[1] These calculations have been successful for the description of rigid polymers or when nonlinear interactions are not important. Unfortunately, even a very simple polymer such as polyethylene can have nonlinear, resonant interactions, and the normal mode method should not be completely valid. (It is possible to add the appropriate coupling in an ad hoc way.)

In general, previous spectral calculations using molecular dynamics were not successful for large systems because dynamical chaos led to uninterpretable spectra.[2] Furthermore, the usual FFT method of producing the spectrum often required the molecular dynamics results at an unmanageable number of time steps. This is because the uncertainty in an FFT spectrum is $2\pi/T$ where T is the duration of the signal or MD simulation. To obtain frequencies reliable to 1 $cm^{-1}$ means T = 33.3 ps. Recently we have developed a new technique which renders the chaotic spectra interpretable and requires dynamic results at orders of magnitude fewer time steps, thereby reducing computational times enormously.[3-10]

## MUSIC Method

The essence of the MUSIC method[4] is that a model for the power spectrum

$$P(\omega) \propto \left| \int_{-\infty}^{\infty} e^{i\omega t} x(t)\, dt \right|^2$$

$$\propto \int_{-\infty}^{\infty} e^{i\omega t}\, P(t)\, dt$$

is assumed and, given a signal x(t) or corresponding estimate of the autocorrelation function

$$p(t) = \int_{-\infty}^{\infty} x(\tau)\, x(\tau + t)\, d\tau$$

of finite duration, an estimate of P can be obtained. In the case of obtaining dispersion curves (collective mode frequencies), x(t) is given by

$$\rho(K, t) = \sum_j e^{iKx_j}$$

where the sum is over all the atoms in the system, $x_j$ are the atomic displacements, and K is the wavevector which varies between $0 < K < 2\pi L$. A dispersion curve consists of frequencies $\omega(K)$. For a given K there might be more than one dispersion curve; e.g., one might have low frequency (acoustic) and high frequency (optical) modes.

The important feature of the MUSIC technique is the factoring of the data autocorrelation matrix R into two vector subspaces, one a signal subspace and the other a noise subspace. From a set of N samples of $\rho(K, t)$ at N time steps for $t = 0$ to $(N - 1)$ t generated from the molecular dynamics method, a complex data vector of $\tilde{X}$ dimension $p + 1$ is defined by

$$\tilde{X}_p = \begin{bmatrix} \rho(K, [m]\Delta t) \\ \rho(K, [m+1]\Delta t) \\ \rho(K, [m+2]\Delta t) \\ \vdots \\ \rho(K, [m+p]\Delta t) \end{bmatrix} \qquad (1)$$

and we will denote $x[m] = \rho(K, m\Delta t)$ to be the *mth* component of this vector. p is the dimension of the total space to be divided into signal and noise subspaces. This vector $\tilde{X}_p$ is used to generate the p*th* order Toeplitz autocorrelation matrix $\tilde{R}$ of dimension $(p + 1) \times (p + 1)$

$$\tilde{\tilde{R}} = \begin{bmatrix} r_{xx}[0] & r_{xx}[-1] & \cdots & r_{xx}[-p] \\ r_{xx}[1] & r_{xx}[0] & \cdots & r_{xx}[-p+1] \\ \vdots & \vdots & & \vdots \\ r_{xx}[p] & r_{xx}[p-1] & \cdots & r_{xx}[0] \end{bmatrix} \tag{2}$$

where an $r_{xx}$ autocorrelation sequence can be estimated from the data samples (e.g., position as a function of time) as

$$r_{xx}[m] = \frac{1}{N-m} \sum_{n=0}^{N-m-1} x[n+m]x^*[n] \ . \tag{3}$$

This matrix can be factored into a signal autocorrelation matrix $\tilde{\tilde{S}}$ and a noise autocorrelation matrix $\tilde{\tilde{W}}$.

$$\tilde{\tilde{R}} = \tilde{\tilde{S}} + \tilde{\tilde{W}} \tag{4}$$

Furthermore, we can solve for the eigenvectors and eigenvalues of $\tilde{\tilde{R}}$ such that

$$\tilde{\tilde{S}} = \sum_{i=1}^{NSIG} \lambda_i \tilde{V}_i \tilde{V}_i^T \ , \quad \text{and} \quad \tilde{\tilde{W}} = \sum_{i=NSIG+1}^{p} \lambda^\circ_i \tilde{V}_i \tilde{V}_i^T \ , \tag{5}$$

where $\tilde{V}_i$ are the eigenvectors of R associated with the NSIG largest eigenvalues $\lambda_i^\circ$ (NSIG to be chosen). Here, NSIG eigenvectors are associated with the NSIG sinusoids assumed present in the data, i.e., the major frequencies expected in the spectrum. In addition, p-NSIG eigenvectors and a set of numerically smaller eigenvalues $\lambda_i$ are associated with the noise subspace. Finally, the MUSIC frequency estimate $P_{MUSIC}(\omega)$ is then

$$P_{MUSIC}(\omega) = \frac{1}{\tilde{e}^T(\omega) \left( \sum_{k=NSIG+1}^{p} \tilde{V}_k \tilde{V}_k^T \right) \tilde{e}(\omega)} \tag{6}$$

where

$$\tilde{e}(\omega) = \begin{bmatrix} \exp(i2\pi[0]\omega) \\ \cdot \\ \cdot \\ \cdot \\ \exp(i2\pi[p]\omega t) \end{bmatrix} \qquad (7)$$

The sum in Eq. (6) is over the eigenvectors of the noise subspace. These eigenvectors will be orthogonal to signal eigenvectors, i.e., $V^T\tilde{e}(\omega) = 0$. Thus, the $P_{MUSIC}$ estimate is effectively infinite when the frequency $\omega$ is found in the signal. However, in practice, the values just become large. We have found that the relative power levels have qualitative significance but are not quantitative.[6] Given a finite duration signal, x(t), one must specify NSIG ≥ number of expected spectral features and Ip < number of data points. Optimal values of NSIG and Ip must be obtained by trial and error initially, although in practice good results are obtained by choosing Ip > 4 NSIG. There also exist functions such as the Akaike information criterion which are useful in estimating NSIG.[4] Marple has provided a convenient set of FORTRAN programs to carry out the analyses discussed above.[4]

In the first series of calculations, we have computed frequencies for an HF polymer chain[3,8] and for stressed polyethylene.[7] The demonstrated computer times vary between 1/80 to 1/1280 of the times that were previously used. Frequencies reliable to better than 1 $cm^{-1}$ were obtained with simulations of duration ~0.2 ps. An example of both quasiperiodic and chaotic spectra for an HF chain is computed with MUSIC and standard FFT method is shown in Fig. 1.

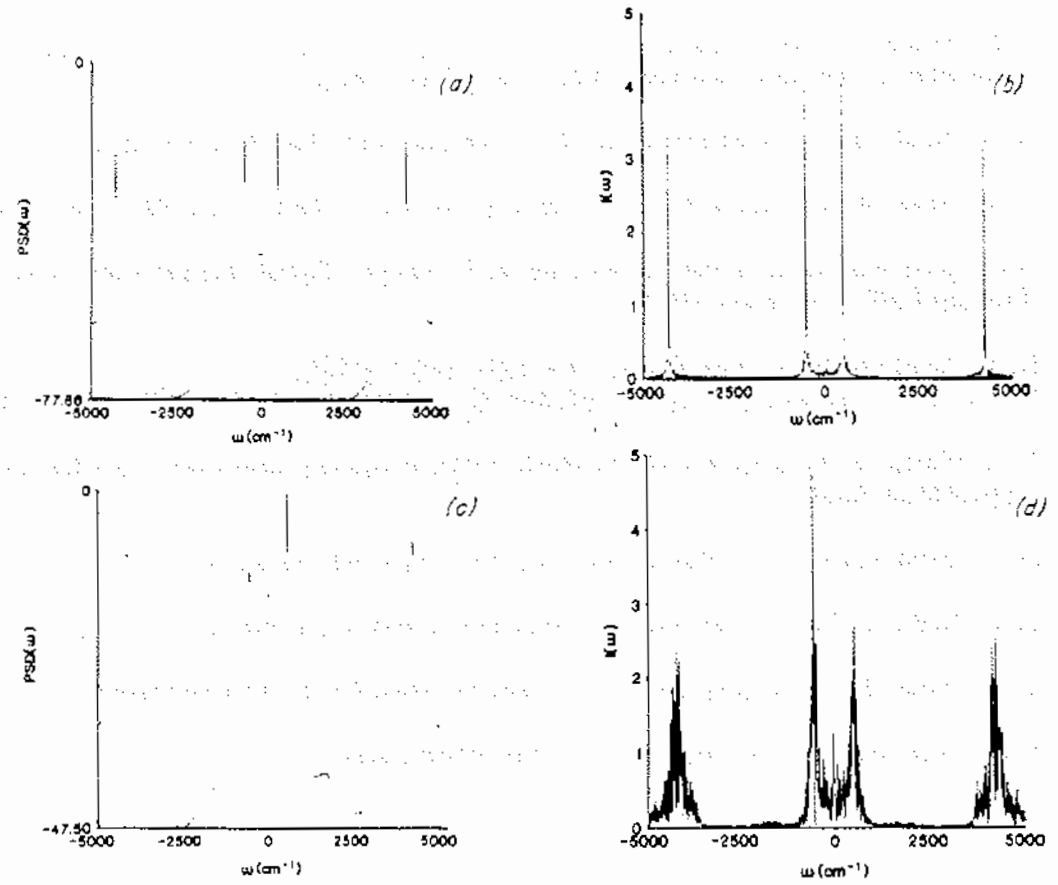



Figure 1. Spectra of quasiperiodic (a, b) and chaotic motion (c, d) using the MUSIC method in (a, c) and the FFT method in (b, d).

In related work[9] we computed the constant volume heat capacity for a crystal from the evaluation of the molecular partition function. This type of evaluation treats each of the 3N vibrations quantum mechanically, each mode having discrete energy levels equally spaced by $\hbar\omega$. Anharmonicity is included by using classical mechanics to calculate the density of states $g(\omega)$ and the frequencies $\omega$ of the modes. Thus, we calculate the constant volume heat capacity using a semiclassical approximation. This approximation is capable of determining the anharmonic contribution of the optical and acoustical modes to the constant volume heat capacity. Analysis of the temperature-dependent $g(\omega)$ spectra shows that anharmonicity tends to cause the optical modes to decrease in frequency and the acoustical modes to increase in frequency as a function of increasing temperature.[9] Using the semiclassical approximation to calculate the constant volume heat capacity, it was shown that the optical modes do not contribute to the heat capacity at low temperatures, but become the predominant factor at high temperatures. We have found that anharmonicity tends to lead to both negative and positive deviations from that of a harmonic model.

We have used MUSIC to carry out other types of spectral analysis such as simulated transition frequency estimates in few body problems and resonance energy determination in wavepacket propagation.[10] Finally, we should remark that MUSIC is certainly not the only alternative to the FFT. Many other approaches, such as maximum entropy and linear prediction, are also discussed in Marple's book.[4] We have simply found MUSIC to be reliable and accurate for the type of analyses discussed here, namely Fourier analysis of short-time MD results.

## References


[1] P. C. Painter, M. M. Coleman, and J. L. Koenig, *The Theory of Vibrational Spectroscopy and Its Application to Polymeric Materials* (Wiley, New York, 1982).

[2] D. W. Noid, M. L. Koszykowski, and R. A. Marcus, Ann. Rev. Phys. Chem. **32**, 281 (1981).

[3] D. W. Noid, B. Broocks, S. K. Gray, and S. L. Marple, J. Phys. Chem. **92**, 3386 (1988).

[4] For a good discussion of this method, see S. L. Marple, *Digital Spectral Analysis with Applications* (Prentice-Hall, New Jersey, 1987).

[5] D. W. Noid and S. K. Gray, Chem. Phys. Lett. **145**, 9 (1988).
D. W. Noid, J. E. Bloor, M. Spotswood, and M. L. Koszykowski, Chem. Phys. Lett. **154**, 391 (1989).

[6] C. E. Wozny, S. K. Gray, and D. W. Noid, "Local Frequency Estimation for Classical Trajectories" (unpublished data).

[7] D. W. Noid and G. A. Pfeffer, J. Poly. Sci. Poly. Phys. Ed. (in press).

[8] B. G. Sumpter, D. W. Noid, and B. Wunderlich, Polymer (in press).

[9] D. W. Noid, B. G. Sumpter, and B. Wunderlich, Anal. Chem. Acta (submitted).

[10] S. K. Gray and C. E. Wozny, J. Chem. Phys. (submitted).

# MOLECULAR DYNAMICS ON VECTOR AND PARALLEL COMPUTERS

An annotated bibliography by David Fincham

## Introduction

This bibliography has been produced primarily to help me in preparing lectures and review articles. However, I think a wider distribution is worthwhile, partly because it is clear that some of the authors in the field are unaware of the work of some others; and partly in the hope that it will prompt people to let me know about work that I myself have missed.

First a reminder about some basic ideas to clarify my terminology. The central computational problem in molecular dynamics is the location of interacting pairs of particles. (A particle here can also mean a small molecule, or an interaction site or group of sites on a large molecule.) Three basic strategies may be adopted, depending on the relationship between the size of the system (which depends on the range of spatial correlations to be investigated) and the range of the potential. In the *all-pairs* strategy a double loop examines each pair of particles in the system and determines their separation. Pairs separated by more than the range of the interaction are then ignored (spherical cut-off). In the *neighbour list* strategy a list is formed of pairs separated by a distance rather greater than the cut-off. Only pairs in the list need be considered during the force evaluation stage of each time step. The list is revised whenever particle movements makwe this necessary. In the *cell list* strategy the computational region is divided into cells, usually of side equal to range of the potential. At the beginning of each timestep a list is formed (usually in the form of a linked-list) indicating which particles are to be found in each cell. Then particles interact only with particles in the same or immediately neighbouring cells.

In vector processing, a *fully vector* loop is one in which all arrays are accessed in constant strides. A *gather* is a process in which one vector is re-ordered into another according to a vector of indices: *scatter* is the reverse of this. *Compression* involves selecting a subset of elements from a vector and storing them consecutively, preserving their order, in another: *expansion* is the reverse of this. A *merge* is a process by which two vectors are combined to give a single result vector by selecting elements either from one or the other.

I call a parallel computer in which each processing element obeys the same instruction stream (DAP, Connection Machine) a *processor array*. I call a parallel computer in which each procesor has its own memory and instruction stream (Intel, Ncube, Transputers) a *multicomputer*.

**Vector methods**

[1] D. Fincham and B. J. Ralston *Molecular dynamics simulation using the Cray-1 vector processing computer*
Comput. Phys. Commun. 23 (1981) 127-134

This first paper on vectorisation of molecular dynamics considered a simple (all-pairs) MD program, involving a double loop over particles, with a spherical cut-off. The IF statement for the cut-off is replaced by a call to a vector merge function which sets out-of-range interactions to zero. The scalar summation involved in accumulating the forces is achieved by storing pair forces in an auxiliary array, which is later summed by a call to the SSUM system routine.

The use of neighbour lists is also considered. In forming the lists a vectorisable loop indexes all the in-range pairs, and a separate non-vectorisable loop forms the list. In evaluating the forces, a gather picks up the required coordinates so a vectorisable loop can evaluate the forces: a scatter then distributes the stored forces back to their arrays.

Other points discussed but not explored in detail are: the use of alternative boundary conditions to cut down the number of "wasted" interactions; the conversion of the double loop over particles to a single loop to give longer vectors; and the vectorisation of the reciprocal space part of the Ewald sum.

[2] R. Vogelsang, M.Schoen, and C. Hoheisal *Vectorisation of molecular dynamics Fortran programs using the Cyber 205 vector processing computer*
Comput. Phys. Commun. 30 (1983) 235-241

The vectorisation of a neighbour-list program is considered. To give longer loops the double loop over particles and neighbours is converted into a single loop which is then broken down into blocks of manageable length. The neighbour list takes the form of a bit-vector with an entry for every particle pair in the system. When a block of interactions is being processed the appropriate part of the bit-vector is used to control compression of the particle coordinates to give a vector of pair separations for the block.

[3] F. Sullivan, R. D. Mountain and J. O'Connell *Molecular dynamics on vector computers*
J. Comput. Phys. 61 (1985) 138-153

The vectorisation of the neighbour list method is considered. To speed up the generation of the list a new method called "the method of lights" is introduced. This is based on sorting: particle data and indices are sorted into arrays in order of increasing X coordinates, and independently in order of Y and of Z coordinates. Appropriate sorting methods for the vector and scalar case are described. The sorted data are used

to locate the neighbours of each particle in a cube around it of side slightly greater than the range of the interaction (compare with cell techniques, where the cubes are fixed in space). A rather complicated series of gather and compress operations uses this information to set up a neighbour list. The neighbour list is two-dimensional: row $i$ list the neighbours (with $j > j$) of particle $i$. Since every particle has some neighbours the intial columns of this array are full. Forces are calculated in vector operations over *columns* of this array ; i.e the first vector operation picks up the first neighbour of every particle, and so on. Subsequent columns may not be full: these are handled by scalar code (presumably a compress or merge could also be used.) The usual gather/scatter is required to go between the list of indices and the coordinate and force arrays.

[4] S. Brode and R. Ahlrichs *An optimised MD program for the vector computer Cyber 205*
Comput. Phys. Commun. 42 (1986) 51-57

The most important achievement of this paper is the introduction of a reordering of the way the pairs are considered in a simple MD program. By making a copy of some of the coordinates the usual double loop over the upper triangle of pair interaction space is converted to a form in which the outer loop is shorter, and the inner loop has a fixed length equal to the number of particles. As well as giving a longer loop, this has the advantage that the force accumulation becomes a true vector operation. (And of course it is still possible to combine the double loop into an even longer single loop). To apply the spherical cut-off the compress available on the Cyber 205 is used to eliminate the out-of-range interactions, rather than simply set them to zero by means of a merge.

[5a] J. Boris *A vectorized near neighbours algorithm of order N using a Monotonic Logical Grid*
J. Comput. Phys. 66 (1986) 1-20

[5b] S.G. Lambrokos and J.P. Boris *Geometric properties of the Monotonic Lagrangian Grid algorithm for near neighbour calculations*
J. Comput. Phys. 73 (1987) 183-202

These papers introduce the Monotonic Logical (or Lagrangian) Grid which provides an alternative to cell methods for large numbers of particles. The particle data are sorted at the start of the simulation into a three dimensional structure such that the X positions increase momotonically with the first index, the Y positions with the second index, and the Z positions with the third index. One can think of each particle as being at one of the points of a distorted rectangular grid. As the particles move during the simulation, the grid distorts further. Whenever two particles interchange the order of one of their Cartesian coordinates, the particle data are exchanged in the data structure. This ensures that the grid remains monotonic. Each grid point thus remains in roughly the same region of space, and in roughly the same relationship with

neighbouring grid points, though it does not always correspond to the same particle. One can therefore find the neighbours of a given particle (grid point) by looking at a fixed set of nearby grid points. These correspond to fixed positions in the data structure and so the procedure is fully vectorisable. Because the grid is irregular, it will be necessary to pick up some particles outside the range of the cut-off in order to be sure of finding *all* particles within range. All these extra interactions are evaluated (though they could presumably be eliminated with a compress operation, at the cost of some loss of vectorisation efficiency).

[6] D.M. Heyes and W. Smith *Cray vectorised link-cell code*
Information Quarterly for Computer Simulation of Condensed Phases 26 (1987) 68-73 and 28 (1988) 63-66. (This is an informal publication, also known as the CCP5 Newsletter, available from SERC Daresbury Laboratory, Warrington, Cheshire WA4 4AD, U.K. )

This article gives code, with little explanation, for the standard cell method as implemented on the Cray. The cell indices for each particle are obtained by a simple transformation of the coordinates. This is done in a single loop over particles and is highly vectorisable. Evaluating the pair interactions within a cell, or between neighbouring cells, involves a gather loop to pick up the coordinates, a vectorisable loop over pairs in which the spherical cut-off is applied by means of a vector merge, and a scatter loop to distribute the pair forces.

[7a] D. C. Rapaport *Large-scale molecular dynamics simulation using vector and parallel computers*
Comput. Phys. Reports 9 (1988) 1-53

[7b] D.C. Rapaport *Vectorised molecular dynamics algorithms* Information Quarterly for Computer Simulation of Condensed Phases 30 (1989) 23-29

These papers ([b] is a summary of [a]) are concerned with the cell-list method in the particular case where the number of particles is very large but the range of the interaction is short, so that a cell will contain not more than two or three particles. In this case the loop over pairs of particles in the neighbouring cells would be too short for effective vectorisation. A method call "layering" is introduced. One particle from each cell is regarded as forming the first "layer" (not in physical space!), another the second layer, and so on. The inner loop of the algorithm is then over all particles in a particular layer. To simplify the application of periodic boundaries "ghost" particles are used around the edges of the box. [a] also considers parallel methods: see [18] below.

[8] M. Schoen *Structure of a simple molecular dynamics Fortran program optimised for Cray vector processing computers*
Comput. Phys. Commun. 52 (1989) 175-185

A standard neighbour list is vectorised. Interacting pairs are gathered into blocks for processing to give adequate vector lengths.

[9a] J. Mociński, J. Kitowski, Z.A. Rycerz and P.W.M. Jacobs *A vectorised algorithm on the ETA 10-P for molecular dynamics simulation of large numbers of particles confined in a long cylinder*
Comput. Phys. Commun. 54 (1989) 47-54

[9b] J. Mościński and J. Kitowski *Proposal of a vectorised MD algorithm for a very large number of particles*
Information Quarterly for Computer Simulation of Condensed Phases 28 (1988) 54-62

In [a] particle indices are sorted according to their coordinate along the cylinder axis, and neighbours found by applying a cut-off in this "index space". This is very similar to the Boris procedure [5] in one dimension, except that only the indices, not the coordinates, are sorted and therefore a gather is required to pick the latter up for vector computation of the forces. [b] proposes to extend this idea to the three-dimensional case by taking slabs of width equal to the cut off.

[10] G.S. Grest, B. Dünweg and K. Kremer *Vectorised link-cell Fortran code for molecular dynamics simulations for a large number of particles*
(preprint)

This is a hybrid method in which a cell list decomposition is used every few steps to set up a neighbour list. A compress operation is used during the force evaluation to eliminate out-of-range interactions from the list of pairs. The slowest part of the algorithm is then the scalar summation involved in accumulating the pair forces onto the total force accumulators. It is shown that this can also be vectorised with a different ordering of the neighbour list, which arises if Rapaport's layer method is used when the list is set up.

**Parallel methods**

[11a] G S Pawley and G W Thomas *Computer simulation of the plastic-to-crystalline phase transition in $SF_6$*
Phys. Rev. Lett. 48 (1982) 410-413

[11b] G S Pawley and G W Thomas *The implementation of lattice calculations on the DAP*
J. Comput. Phys. 47 (1982) 165-178

[11c] G. S. Pawley and M.T. Dove *The one-dimensional plastic phase of $SF_6$: a simulation*
Chem. Phys. Lett. 99 (1983) 45-48

[11d] M.T. Dove and G.S. Pawley *A molecular dynamics simulation study of the plastic crystalline phase of sulphur hexafluoride*
J. Phys. C: Solid State Phys. 16 (1983) 5969-5983

This series of papers was the first to consider the use of the DAP for molecular dynamics. A spatial decomposition of the problem was adopted with one molecule per processing element. Because these plastic crystals maintain their translational order each molecule remains in its own processor and the molecules with which it interacts can be found in a fixed set of nearby processors. This makes it easy to evaluate interactions with neighbours lying in a particular direction simultaneously for all molecules. These were three-dimensional simulations and [b] discusses the general problem of mapping three-dimensional problems onto two-dimensional processor arrays. There have been numerous subsequent papers from this group using the same technique for studies of various plastic crystals.

[12a] D Fincham *Molecular dynamics and graphics using the DAP*
in "Parallel Computing 83", M Feilmer, J Jonbert and U Schendel (eds.) (Elsevier, Amsterdam, 1984)

[12b] M.Neumann, O. Steinhauser and G.S. Pawley *Consistant calculation of the static and frequency dependent dielectric constant in computer simulations*
Molec. Phys. 52 (1984) 97-113

[12c] D. Fincham, N. Quirke and D.J. Tildesley *Computer simulation of molecular liquid mixtures*
J. Chem. Phys. 84 (1986) 4535-4546

[12d] D.J. Adams and G.S. Dubey *Taming the Ewald sum in the computer simulation of charged systems*
J. Comput. Phys. 72 (1987) 156

These papers used a simple all-pairs technique on the DAP. All pair interactions in the system were considered, in blocks of 64 x 64 at a time. Out-of-range

interactions are set to zero: this is very easy using the logical indexing available in DAP Fortran. In [a] the advantage of using non-cubic boundary conditions in this case is pointed out. [b] combines the method with a reaction field for the long-range forces. [c] combines the method with particle insertion for evaluation of the potential energy. [d] introduces effective pair potential approximations to the Ewald sum: since they are based on minimum image vectors *without* the use of a spherical cutoff they run at full efficiency on the DAP.

[13] N.S. Ostland and R.A. Whiteside *A machine architecture for molecular dynamics: the systolic loop*
Annals N.Y. Acad. Sci 439 (1985) 195-208

This paper shows that the "tractor tread" systolic loop algorithm, previously developed by the authors for Monte Carlo simulation, can also be used for molecular dynamics.

[14a] D. Fincham *Parallel computers and molecular simulation*
Molec. Simulation 1 (1987) 1-45

[14b] D. Fincham *Parallel computers and the simulation of solids and liquids*
Lectures given at the NATO Advanced Study Institute, Bath, 1988 (to be published)

These are primarily reviews, but I did make one or two new points. In [a] I suggested that the Monotonic Grid method might be suitable for liquid simulations on the DAP: this was subsequently done by M. Allen (unpublished) and I expanded on the point in [b]. I also suggested that the Brode-Ahlrichs vectorised all-pairs method could be used as a processor array method or as a multicomputer method. In the latter case it becomes a systolic loop method, later to be known as SLD. I also introduced a new systolic loop method, called "pass the parcel" in [a] but later known as SLS.

[15] F. Bruge, V. Martorana and S.L. Fornili *Concurrent molecular dynamics simulation of ST2 water on a Transputer array*
Molec. Simulation 1 (1988) 309-320

This implementation uses a mapping of particles to processors called "boustrophedonic". The processors are connected in a ring and the method is similar in principle to the systolic loop methods. The program is written in Occam.

[16] A.R.C. Raine, D. Fincham and W. Smith *Systolic loop methods for molecular dynamics simulation using multiple Transputers*
Comput. Phys. Commun. 55 (1989) 13-30

This paper is a systematic examination of systolic loop methods. The statistics of load balancing, and the communication overheads, are examined. It is shown that the methods can be reorganised so that computation can proceed in parallel with communication. On Transputers SLS is the best method. Providing the ratio of particle to processor numbers is greater than about 10 the methods achieve 100 % efficiency.

[17] H. Grubmüller, H. Heller and K. Schulten *Molecular dynamics simulation on a parallel computer* Molec. Simulation (to appear)

These authors have built a Transputer-based machine and written an Occam program for the molecular dynamics of biopolymers (based on CHARMM protocols). The method is similar to the SLD systolic loop.

[18] see [7] above

Here the author considers parallel methods based on a spatial decomposition. The computational box is divided into slabs of equal size, each associated with a processor. The processors are connected in a ring. Each processor evaluates interactions between particles inside its slab: interactions between particles in neighbouring slabs are calculated in both processors after exchange of the relevant particle coordinates. Because of the ring topology all processors are equivalent and periodic boundaries are easily taken into account. The ratio of communication to computation depends on the ratio between the cut off distance (controlling the amount of data to be exchanged between processors ) and the thickness of the slab. As particles move during the simulation they will eventually cross the boundaries between slabs. Some bookkeeping work is then necessary to maintain the data in each processor in an organised state.

[19] H.G. Petersen and J.W. Perram *Molecular dynamics on transputer arrays. I. Algorithm design, programming issues, timing experiments and scaling projections* Molec. Phys. 67 (1989) 849-860

This paper also considers a slab-based decomposition on a ring of processors. A standard spatial decomposition using a linked list is used within a slab. The interest is in modest numbers of particles but large numbers of processors so that the slab thickness may be less than the cut-off. Then particles need to interact with others in several nearby processors, and coordinates and force accumulators circulate by the required number of steps in a manner very similar to the systolic loop methods.

[20] G.S. Pawley, C.F. Baillie, E. Tenenbaum amd W. Celmaster *The BBN Butterfly used to simulate a molecular liquid* Parallel Computing 11 (1989) 321-329

The Butterfly is a parallel computer in which processors and memory modules are connected via a switching network so that in effect all the memory is shared. A neighbour list program is implemented. The evaluation of interactions between a particular molecule "i" and its set of neighbours "j" from the list is treated as an independent task: the operating system distributes these tasks to free processors. As is usual the list only contains entries for j>i. The tasks get progressively smaller and this leads to good load balancing. Conversely, more than one task will calculate forces for a particular particle. To minimise memory contention a task first accumulates "j" forces into a private array, before adding into global memory under control of a lock. Nevertheless efficiency drops for large numbers of processors.

## Conclusions

An article in these pages once stated that 'vectorisation can often be accomplished in an afternoon of undemanding work'. It is certainly true that existing programs can often be adapted by a judicious use of gather/scatter and compress/expand to achieve a speed-up from scalar code by a factor of four or so. However, although these procedures may have some hardware support, they can not be regarded as fully vector operations, and other problems may also remain. In the all-pairs case the standard loop ordering leaves a scalar summation for the force on particle "i", as well as a shortening inner loop [1]. A standard neighbour list involves an inevitable gather/scatter, and also a short inner loop (over neighbours of a particular particle), though it is possible to get round this by converting the double loop to a loop over pairs in various ways [2,3,8]. Cell-list methods involve gather/scatter, and also a short inner loop, over particles in a neighbouring box [6]. Again, there are ways around this problem, as in the hybrid methods combining cell decomposition with a neighbour list [3,10].

More significant gains in performance are possible, but require new algorithms which are intrinsically more vectorisable. In my opinion there are two really significant papers in the vectorisation of molecular dynamics. For moderate number of particles the simple loop re-ordering of Brode and Ahlrichs [4] makes the all-pairs problem fully vector, removing any scalar summation. For larger systems the Monotonic Grid of Boris [5] provides a spatial decomposition with a regular data structure which gives a fully vector force loop of length equal to the number of particles. It is interesting to note that these papers did not appear until five years after [1]. The layer method [7] is an alternative approach to efficient vectorisation with a spatial decomposition and it would be interesting to see it benchmarked against the Monotonic Grid.

Turning to processor arrays, a straightforward all-pairs method [12] works well on moderate sized systems. (An attempt was made in the early days by Berendsen's group to introduce a neighbour list in the form of a logical mask and then use compress and expand type operations on it: though possible this does not lead to a very efficient implementation.) On larger systems a spatial decomposition with one particle per processor is used for lattices [11] and can be applied to liquids in Monotonic Grid form [14].

On multicomputers systolic loop methods are fairly obvious and have been invented independently a number of times [13,14,15,16,17 and others], though not always in the most efficient form. (The really clever idea in systolic loops is the original tractor-tread for Monte-Carlo since it allows independent particle moves to be made on a parallel computer, but that is another story.) They have the advantage that a distributed neighbour list may be incorporated [16]. For large systems a spatial decomposition with processors controlling a particular region of space [18,19] is also

an obvious idea which works well as long as the system is really large so that the ratio of calculation to communication is reasonably high.

Any vector or parallel method suffers from the fact that one requires neighbours in a spherical region about a particular particle, and the efficiency of finding these is always improved if the geometry of the cell or sub-cell is as close to spherical as possible. I made this point in [1], but very few subsequent papers have adopted this approach.

# Quaternion quickie

*M. P. Allen, H. H. Wills Physics Laboratory*
*Royal Fort, Tyndall Avenue, BRISTOL BS8 1TL*

I read a couple of nice articles on quaternions in the latest *Molecular Simulation* [1, 2]. This, and a conversation with a colleague, prompted me to write this note. It deals with a little technical point, and another minor, but interesting, historical one.

Suppose that I have carried out a simulation of a system of rigid molecules using constraints, or some other method based on atomic positions or bond direction cosines. Sometimes, I want to convert these positions, or equivalently the $3 \times 3$ rotation matrix for each molecule, into quaternions. The converse operation, expressing the rotation matrix as a function of the quaternions, is part of any quaternion-based simulation program. Assuming that I have adopted the most symmetrical definition of the quaternion parameters $(q_0, q_1, q_2, q_3)$, the rotation matrix **A** is

$$\begin{pmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{pmatrix} = \begin{pmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1 q_2 - q_0 q_3) & 2(q_1 q_3 + q_0 q_2) \\ 2(q_1 q_2 + q_0 q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2 q_3 - q_0 q_1) \\ 2(q_1 q_3 - q_0 q_2) & 2(q_2 q_3 + q_0 q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{pmatrix}.$$

Inverting this, to obtain $(q_0, q_1, q_2, q_3)$ in terms of the elements of **A**, seems trivial at first sight. Let the trace of the matrix be $T = A_{11} + A_{22} + A_{33}$, and recall that the quaternions are normalized, $q_0^2 + q_1^2 + q_2^2 + q_3^2 = 1$. Then we can see that the 'scalar' element of the quaternion can easily be obtained from the trace

$$q_0 = \sqrt{(T+1)/4} \tag{1}$$

while the off-diagonal terms will yield the rest of the elements

$$q_k = (A_{ji} - A_{ij})/4q_0. \tag{2}$$

Here $i, j, k$ are in cyclic 1,2,3 order. Usually all will be well, but disaster will strike whenever $q_0 \to 0$, i.e. whenever the molecule is upside down. At this limit we get $0/0$, and close to it we will experience great numerical imprecision. With many molecules in our simulation, this is a very real danger: just as real as the divergence in the Euler angle equations of motion which prompted Denis Evans [3] to adopt quaternions in the first place.

The solution to the problem is simple, straightforward, and has appeared in the literature [4]. I'll just reproduce it here. Things look most pretty if we define $A_{00} = T$. Then we have

$$\begin{aligned} 4q_0^2 &= 1 + 2A_{00} - T \\ 4q_1^2 &= 1 + 2A_{11} - T \\ 4q_2^2 &= 1 + 2A_{22} - T \\ 4q_3^2 &= 1 + 2A_{33} - T \end{aligned} \tag{3}$$

and

$$\begin{aligned} 4q_0 q_1 &= A_{32} - A_{23} & \qquad 4q_2 q_3 &= A_{32} + A_{23} \\ 4q_0 q_2 &= A_{13} - A_{31} & \qquad 4q_3 q_1 &= A_{13} + A_{31} \\ 4q_0 q_3 &= A_{21} - A_{12} & \qquad 4q_1 q_2 &= A_{21} + A_{12}. \end{aligned} \tag{4}$$

Ten equations, four unknowns. The method is as follows. Choose, from equations (3) the component $q_i$ with the largest absolute magnitude. Calculate it from (3), giving it (arbitrarily) a positive sign. We are allowed to do this because two quaternions, differing only by reversing the signs of all the components, are equivalent representations of a given rotation. Amongst equations (4) there are three involving this element $q_i$. These may be used to determine the other elements (with the correct relative signs) exactly as in equation (2). The difference is that we are dividing by $q_i$, which is guaranteed to be far from zero. In fact it is easy to see that $\frac{1}{2} \leq |q_i| \leq 1$, from the normalization condition. No worries.

Now this may seem obvious, but actually several other schemes for doing this have been proposed, including the crude one of eqns (1,2) and various other improvements on it [5]. The one described here is, I think, the most symmetrical and foolproof. Another point in its favour is that it can be programmed so as to generate shorter IBM 370 code than its rivals. This is apparently important, when it is to be used in the guidance computer of a spacecraft.

Spacecraft? Yes indeed, as you can see by looking at the reference list, I have been reading further afield than usual. The guidance systems of Skylab, and the space shuttle, use (used in the former case) quaternion parameters rather than Euler angles, for exactly the same reasons that we adopt them in our simulations: avoidance of the Euler angle singularity, and economy of code. It seems that NASA knew all about the technique, long before it was adopted in the molecular simulation community. They were using the Runge-Kutta algorithm with quaternions in 1969 [6], and I've even seen a reference to an intriguingly-titled 1958 paper [7], although I must confess I've not yet managed to lay my hands on it.

Now this is not in any way intended to diminish Denis Evans' original work in the field. After all, the crucial equation of motion (the link between the angular velocity vector and the time derivatives of the quaternion parameters) appears in Whittaker's text on classical mechanics [8], referred to by Evans, thus setting the basis of the technique firmly in the mists of antiquity. Nobody, to my knowledge, had suggested using it for molecules before, and this was the big contribution. The space scientists, as it happened, had come across the problem earlier, and had arrived at the same solution. The point of this note is that there may be a few other technical, computational, points that we could pick up from the space science literature. Quite complex problems of linked rigid bodies, for example, are handled in the standard texts [9].

My thanks go to my colleague, Brian Pollard, who directed me to the spacecraft literature cited on the next page.

# References


[1] R. M. Lynden-Bell and A. J. Stone, "Reorientational correlation functions, quaternions and Wigner rotation matrices", Molec. Simulation **3** 271–281 (1989).

[2] G. R. Kneller and A. Geiger, "A method to calculate the g-coefficients of the molecular pair correlation function from molecular dynamics simulations", Molec. Simulation **3** 283–300 (1989).

[3] D. J. Evans, "On the representation of orientation space", Molec. Phys. **34** 317–325 (1977); D. J. Evans and S. Murad, "Singularity free algorithm for molecular dynamics simulation of rigid polyatomics" Molec. Phys. **34** 327–331 (1977).

[4] S. W. Shepperd, "Quaternion from rotation matrix", J. Guidance and Control **1** 223-224 (1978).

[5] C. Grubin, "Derivation of the quaternion scheme via the Euler axis and angle", J. Spacecraft and Rockets **7** 1261–1263 (1970); C. C. Rupp, "Equations for calculating initial values of the four parameters", George C Marshall Space Flight Center, Huntsville Ala., Memo R-ASTR-NGA (1968); A. C. Hendley, "Quaternions for control of space vehicles", Proc. Inst. Navigation, National Space Meeting on Space Shuttle - Space Station - Nuclear Shuttle Navigation, Huntsville, Ala. (1971); A. R. Klumpp, "Singularity-free extraction of a quaternion from a direction-cosine matrix", J. Spacecraft and Rockets **13** 754–755 (1976); Spurrier, "Comment on 'Singularity-free extraction of a quaternion from a direction-cosine matrix' ", J. Spacecraft and Rockets **15** 255 (1978).

[6] A. C. Fang and B. G. Zimmerman, "Digital simulation of rotational kinematics", NASA TN D-5302, Washington DC (1969).

[7] A. C. Robinson, "On the use of quaternions in simulation of rigid body motion", WADC Tech. Rept. 58-17 (1958). Apparently WADC stands for Wright Air Development Center, Wright-Patterson Air Force Base, Ohio.

[8] E. T. Whittaker, "A treatise on the analytical dynamics of particles and rigid bodies" (Cambridge University Press, 1960; but 1st edition 1904, I think).

[9] P. C. Hughes, "Spacecraft attitude dynamics" (Wiley, 1986). Incidentally, from this reference I learned that a rigid body suspended in gimbals, for which the Euler angles correspond exactly to the individual degrees of freedom, will show the physical effects of the singularity in the equations of motion. The phenomenon is called *gimbal lock*; it is avoided by using 4-gimbal suspension systems rather than 3. For genuine free rotors, of course, the singularity is confined to the mathematics.

# Error in "Computer Simulation of Liquids"

*M. P. Allen, H. H. Wills Physics Laboratory*
*Royal Fort, Tyndall Avenue, BRISTOL BS8 1TL*
*D. J. Tildesley, Department of Chemistry*
*The University, SOUTHAMPTON SO9 5NH*

We have become aware of a small error in our book "Computer Simulation of Liquids" [1]. It is small in the $\mathcal{O}(1/N)$ sense, but that is not to say that it might not be significant in appropriate circumstances. To prevent the mistake becoming (more) widespread, we are mentioning it here.

In section 4.5 (pp 123–126) we discuss isothermal-isobaric Monte Carlo. If the configuration of the system is written in terms of the box volume $V$ and box-scaled coordinates s, then averages of a property $\mathcal{A}$ may be written

$$< \mathcal{A} >_{NPT} \propto \int_0^\infty \mathrm{d}V \int \mathrm{ds}\ \exp(-\beta \mathcal{V}) \exp(-\beta P V) V^N\ \mathcal{A}.$$

Here $T$ is the temperature, $P$ the pressure, $N$ the number of atoms, and $\mathcal{V}$ the potential energy function. As usual $\beta = 1/k_B T$, $k_B$ being Boltzmann's constant. The appropriate ensemble may be simulated by sampling $V$ and the scaled coordinates uniformly, and using the weight function

$$\exp(-\beta \mathcal{V}) \exp(-\beta P V) V^N$$

or equivalently using the function

$$\delta H = (\mathcal{V}_{\mathrm{new}} - \mathcal{V}_{\mathrm{old}}) + P(V_{\mathrm{new}} - V_{\mathrm{old}}) - N k_B T \ln(V_{\mathrm{new}}/V_{\mathrm{old}})$$

in a pseudo-Boltzmann factor $\exp(-\beta \delta H)$ for assessing acceptance or rejection of trial moves in the usual Metropolis scheme.

However, in equation (4.31) of our book, and the surrounding text, we describe uniform sampling of the box *length* $L$, not the volume. If $V = L^3$, these are not equivalent. If we wish to sample $L$ we must first change variables to give

$$< \mathcal{A} >_{NPT} \propto \int_0^\infty \mathrm{d}L \int \mathrm{ds}\ \exp(-\beta \mathcal{V}) \exp(-\beta P L^3) L^{3N+2}\ \mathcal{A}$$

so the weight function can be written

$$\exp(-\beta \mathcal{V}) \exp(-\beta P L^3) L^{3N+2}$$

and the function $\delta H$ becomes,

$$\begin{aligned} \delta H &= (\mathcal{V}_{\mathrm{new}} - \mathcal{V}_{\mathrm{old}}) + P((L^3)_{\mathrm{new}} - (L^3)_{\mathrm{old}}) - (3N+2) k_B T \ln(L_{\mathrm{new}}/L_{\mathrm{old}}) \\ &= (\mathcal{V}_{\mathrm{new}} - \mathcal{V}_{\mathrm{old}}) + P(V_{\mathrm{new}} - V_{\mathrm{old}}) - (N + \tfrac{2}{3}) k_B T \ln(V_{\mathrm{new}}/V_{\mathrm{old}}). \end{aligned}$$

The factor $N k_B T$ has become $(N + \frac{2}{3}) k_B T$. Unfortunately, in the text, we gave the $V$-sampling version rather than this one.

In Wood's original article [2] he describes uniform sampling of the *inverse* box length. Setting $\lambda = L^{-1}$ it is easy to derive his weight function

$$\exp(-\beta \mathcal{V}) \exp(-\beta P \lambda^{-3}) \lambda^{-(3N+4)}$$

and $\delta H$

$$\begin{aligned}\delta H &= (\mathcal{V}_{\rm new} - \mathcal{V}_{\rm old}) + P((\lambda^{-3})_{\rm new} - (\lambda^{-3})_{\rm old}) + (3N+4)k_B T \ln(\lambda_{\rm new}/\lambda_{\rm old}) \\ &= (\mathcal{V}_{\rm new} - \mathcal{V}_{\rm old}) + P(V_{\rm new} - V_{\rm old}) - (N+\tfrac{1}{3})k_B T \ln(V_{\rm new}/V_{\rm old}).\end{aligned}$$

Finally, Eppenga and Frenkel [3] consider sampling $\ln V$ rather than $V$. In this case the weight is (expressed in terms of $V$)

$$\exp(-\beta \mathcal{V}) \exp(-\beta P V) V^{N+1}$$

so

$$\delta H = (\mathcal{V}_{\rm new} - \mathcal{V}_{\rm old}) + P(V_{\rm new} - V_{\rm old}) - (N+1)k_B T \ln(V_{\rm new}/V_{\rm old}).$$

So, in summary, it is easy to derive the correct weight function for the sampling scheme chosen, and we hope we have got it right above, but we didn't do it in the book.

Mike Stapleton first drew our attention to this mistake; our thanks go to him, and to one or two other colleagues who also noticed it. We are not aware of any other significant errors in the book, but of course if anyone else has spotted one, please let us know.

# References

[1] M. P. Allen and D. J. Tildesley, "Computer Simulation of Liquids" (Oxford University Press 1987; paperback edition 1989).

[2] W. W. Wood, "Monte Carlo studies of simple liquid models", in *Physics of Simple Liquids* (Ed: H. N. V. Temperley, J. S. Rowlinson and G. S. Rushbrooke) pp 115-230 (North Holland 1968).

[3] R. Eppenga and D. Frenkel, "Monte Carlo study of the isotropic and nematic phases of infinitely thin hard platelets", Molec. Phys. **52**, 1303--1334 (1984).

# CCP5 Literature Survey 1988 - Addendum

W. Smith

July 17, 1989

Bishop, M. and Saltiel, C.J.
Polymer shapes in two-, four- and five-dimensions
J. Chem. Phys. 88 3976 (1988).

Bishop, M.
The WCA reference system for four- and five-dimensional Lennard-Jones fluids.
J. Chem. Phys. 88 5779 (1988).

Bishop, M. and Saltiel, C.J.
Application of the pivot algorithm for investigating the shapes of two- and three-dimensional lattice polymers.
J. Chem. Phys. 88 6594 (1988).

Bishop, M. and Saltiel, C.J.
Universal properties of linear and ring polymers.
J. Chem. Phys. 89 1159 (1988).

Bishop, M.
The collapse transition for two-dimensional linear and ring polymers.
J. Chem. Phys. 89 1719 (1988).

Cagin, T. and Ray, J.R.
Isothermal molecular dynamics ensembles.
Phys. Rev. A 37 4510 (1988).

Cagin, T. and Ray, J.R.
Fundamental treatment of molecular dynamics ensembles
Phys. Rev. A 37 247 (1988).

Cagin, T. and Ray, J.R.
Elastic constants of sodium from molecular dynamics.
Phys. Rev. B 37 699 (1988).

Cagin, T. and Ray, J.R.
Third order elastic constants from molecular dynamics: Theory and an example calculation.
Phys. Rev. B 38 7940 (1988).

Frenkel, D., Kil, A.J., van de Doef, B.M.,and Zijlstra R.J.J.,
Monte Carlo simulation of carrier number noise spectra in the integral quantum Hall regime.
J. Phys. C21 L177 (1988).

Frenkel, D., Lekkerkerker, H.N.W., and Stroobants, A.,
Thermodynamic stability of a smectic phase in a system of hard rods.
Nature 332: 822 (1988).

Frenkel, D.,
Structure of hard-core liquid crystals.
J. Phys. Chem. 92 3280 (1988).

Hayoun, M., Meyer, M. and Turq, P.
Molecular dynamics study of self diffusion in a liquid-liquid interface.
Chem. Phys. Letters 147 203 (1988).

Huang, J., Meyer, M. and Pontikis, V.,
Adequacy of Al and Cu pseudopotentials for the computer simulation of edge locations.
Scripta Metallurgica 22 463 (1988).

Kluge, M.D. and Ray J.R.
Elastic constants and density of states of a molecular dynamics model of amorphous silicon.
Phys. Rev. B 37 4132 (1988)

Kranendonk, W.G.T. and Frenkel, D,
Simulation of the adhesive-hard-sphere model.
Mol.Phys. 64:403 (1988).

Ladd A.J.C., Frenkel D. and Colvin M.E.,
An application of lattice-gas cellular automata to the study of Brownian motion.
Phys. Rev. Lett. 60 975 (1988)

Mansour, K.A., Murad, S. and Powles J.G.
A computer simulation study for model liquid Ammonia - time correlation functions.
Molec. Phys. 65 785 (1988).

Meyer, M., Mareschal, M. and Hayoun, M.
Computer modelling of a liquid-liquid interface.
J. Chem. Phys. 89 1067 (1988).

Polian, A., Loubeyre P., and Boccara N. eds.,
*Introduction to Computer Simulation. Proceedings of the Les Houches Workshop on 'Simple Molecular Systems at Very High Density'*,
Plenum, New York,(1988), p.411.

Powles, J.G., Williams M.L. and Evans W.A.B.
Does a microscopically inhomogeneous polar liquid have a dielectric constant?
J. Phys C 21 1639 (1988).

Powles, J.G., Rickayzen, G. and Williams M.L.
The density profile of a fluid confined to a slit.
Molec. Phys. 64 33 (1988).

Ray J.R.
Elastic constants and statistical ensembles in molecular dynamics.
Comput. Phys. reports. 8 109 (1988).

Ray, J.R., and Rahman, A.
Molecular dynamics methods to study structural phase transformations in solids.
Physica B 150 250 (1988).

van Waveren, M., Bishop, M. and Michels J.P.J.
Brownian dynamics study of the relaxation behaviour of linear and ring polymers.
J. Chem. Phys. 88 1326 (1988).