DARESBURY LABORATORY

INFORMATION QUARTERLY for MD & MC SIMULATIONS

An Informal Newsletter associated with Collaborative Computational Project No. 5 on Molecular Dynamics and Monte Carlo Simulations of Macroscopic systems.
Number 2 September 1981

Contents

Editor: Dr. William Smith
Science & Engineering Research Council, Daresbury Laboratory, Daresbury, Warrington WA4 4AD, England

Deputy Editor: Dr. David Heyes
Royal Holloway College, University of London, Egham TW20 0EX England

Mathematics Qualifying Exam

Fall 2010

Instructions:

- Do all problems.
- Show your work.
- Justify your answers.

Editorial

Welcome to the second issue of the CCP5 Newsletter. This issue consists almost entirely of articles written by active participants in CCP5 and contains useful comments and ideas which will hopefully be of interest to all our colleagues. It is a pleasure to thank the contributors for the articles appearing here. We should also like to remind our readers that anyone may submit an article to this newsletter, on any topic which may be of interest to our subscribers. We should like our readers, in fact, to think of the newsletter as the natural means of communicating ideas to the CCP5 participants. Please make use of it!

General News

1. D. Fincham and N. Anastasiou have made generally available their molecular dynamics program MDIONS, which enables the simulation of ionic liquids or molten salts. The program employs the usual periodic boundary conditions, a Born-Mayer-Huggins form for the short range potential functions and an Ewald sum to treat the long-range coulombic interaction.

Copies of the program are available in the Daresbury TSO datasets FK.MDIONS.ONE.CRAYJCL (standard version) and FK.MDIONS.TWO.CRAYJCL (vectorised version). CCP5 participants who are not Daresbury users may obtain copies from Dr. W. Smith at Daresbury Laboratory, who will also have the program documentation available shortly. Subscribers to the Computer Physics Communications (CPC) Program Library should note that MDIONS will be included soon.

2. The next CCP5 meeting will be on the computational aspects of "The Structure of the Interfacial Region". This meeting will take place in OXFORD (and not Southampton as was announced in the previous newsletter). The meeting will follow on from the Faraday Society meeting under the same title. The Faraday Society meeting is scheduled for 16th-17th December 1981 and the CCP5 meeting is scheduled for 17th-18th December. The speakers will include Valleau (Toronto), Woodcock (Amsterdam), Frenkel (Utrecht) and Toxvaerd (Sweden). Readers interested in attending should contact:-

Dr. D.J. Tildesley,
Department of Chemistry,
University of Southampton,
Highfield,
Southampton SO9 5NH.

3. The following CCP5 meeting will be on the subject of TRANSPORT PROPERTIES (Royal Holloway, March 1982). It is to be organised by Dr. P. Madden (Cambridge) and Dr. M. Gillan (Harwell).

NEAREST-IMAGE TRANSFORMATION ON CDC COMPUTERS

Nigel Corbin and David Fincham

The nearest-image transformation, inside the force loop, is one of the most time-consuming parts of a molecular dynamics program. Many people use the following technique to look for the nearest-image vector between a pair of particles (the co-ordinates are scaled to lie between -1 and +1)

```
RX = X(I)-X(J)
RX = RX-2*INT(RX)
```

with similar expressions for y and z components. This works well on many computers. However, for CDC machines there is a faster method which requires minimal program changes. We start the force evaluation by defining a set of integer co-ordinates

```
IX(I) = INT(X(I)*TWOTW)
```

where TWOTW = 1048576.0 = 2^{20}. This power of two is chosen so that the truncation introduces negligible errors, but integer overflows are avoided. The n.i. transformation can then be programmed using shift operations

```
IRX = IX(I)-IX(J)
IRX = IRX-SHIFT(SHIFT(IRX,-20),21)
```

Then RX = IRX/TWOTW is the correctly transformed component of the particle-particle vector. In practice we work with IRX until the cut-off has been applied, to take advantage of the greater speed of integer arithmetic.

For 108 argon atoms this method reduced the CPU time per step of the simulation from 36 ms to 29 ms.

THE EWALD SUM PROGRAM "MDIONS"

Nicholas Anastasiou and David Fincham

This note gives a brief description of the program MDIONS in the CCP5 program library. Copies of the program source and full documentation can be obtained from the program librarian, Dr. Bill Smith.

The program performs dynamic simulations (MD) on charged particles. The non-Coulomb part of the interaction is specified by means of rigid ion potentials of the Born-Mayer-Huggins form. A mixture of several different species of ion may be simulated. The Coulombic part of the interaction is handled by means of the Ewald sum technique. The routine performing the reciprocal space part of the Ewald sum is very easily adapatable to an ortho-rhombic computational box, which would be useful in the study of non-cubic crystals. The leapfrog algorithm is used to integrate the equations of motion of the ions.

As well as the usual thermodynamic averages the program calculates the mean square displacement of the ions (diffusion) and the partial r.d.f's. The partial structure factors are found by two methods. At low k values they are calculated by direct evaluation of the sums $\Sigma \exp(i\underline{k}.\underline{r})$. This method involves negligible extra work as these sums are required in the reciprocal space sum, and it complements the second calculation which works by Fourier transform of the r.d.f's. (the latter method can lead to spurious oscillations at low k-values because of the cut-off).

There are three versions of the program. The first version is written in conventional Fortran to be suitable for any computer. This has been submitted for publication in the Computer Physics Communications program library, and users of any version of the program are requested to make reference to this publication where appropriate. This version allows both the use of look-up tables and direct evaluation for the real space force calculation. A second version involves the minimal changes necessary to vectorise the force calculation on the Cray. Look-up tables are not included in this version as they would inhibit vectorisation. A third version has been optimised by Bill Smith for the Cray, partly by means of assembly language routines.

Typical execution time for the vectorised version on the Cray is 115 ms per step for 216 ions, with the r.d.f. calculation switched off. The optimised version reduces this to 100 ms.

AN ALGORITHM FOR ROTATIONAL MOTION OF RIGID MOLECULES

David Fincham

Dynamic simulations of rigid polyatomic molecules often use a quaternion method[1] to integrate the rotational motion. The quaternions provide a representation of the orientation of the molecule in which the equations of motion are free of singularities. These equations have usually been integrated using a fourth-order predictor-corrector method[2]. The purpose of this article is to suggest the use of a new and simpler algorithm to integrate these equations which gives better results than the usual algorithm and is also superior to the alternative constraint method[3].

To explain our motivation we first consider the case of centre-of-mass motion. If we write the equation of motion of the COM as two first-order equations, using an obvious notation

$$\dot{\underline{V}} = \underline{F}/m \tag{1a}$$

$$\dot{\underline{R}} = \underline{V} \tag{1b}$$

and use Taylor series expansion it is easy to derive the leapfrog algorithm

$$\underline{V}^{n+1/2} = \underline{V}^{n-1/2} + \Delta t\ \underline{F}^{n}/m \tag{2a}$$

$$\underline{R}^{n+1} = \underline{R}^{n} + \Delta t\ \underline{V}^{n+1/2} \tag{2b}$$

(This algorithm is algebraically equivalent to the Verlet algorithm but computationally superior). In the following table we compare the energy conservation of the leapfrog with a high-order predictor-corrector algorithm, the Gear 4-level formula[4]. The table shows the RMS fluctuation in total energy (arbitrary units) for an argon simulation.

Δt(fs)	10	20	30
Leapfrog	12	19	27
Gear 4-level	11	14	123

The high order algorithm performs slightly better at smaller time-

steps, but its errors increase vary rapidly at large time-steps, whereas the errors in the leapfrog increase only steadily with increasing Δt. On theoretical grounds the Gear formulae are expected to be very accurate and stable, but the theoretical analysis applies strictly only to the case of a particle moving in a fixed force field. This is not the case in a dynamic simulation where each particle moves in the fluctuating force field produced by its neighbours. The high-order algorithms predict the time derivatives of the force from its value at previous time-steps, but it would appear that this prediction is invalidated by the fluctuations in the force field in the liquid at larger time-steps, and indeed has a destabilising effect on the algorithm. This is important because we want to use as large a time-step as possible in order to sample phase space most efficiently, and the leapfrog is therefore the most suitable algorithm.

The evidence[5] indicates that in a molecular liquid the fluctuations in torque act on a very similar time-scale to the force flutuations, suggesting that a simple low-order algorithm might also be suitable for rotational motion. The basic equations governing rotational motion are the following

$$\dot{\underline{J}} = \underline{T} \tag{3a}$$

$$\underline{J}_p = \underline{\underline{A}}\, \underline{J} \tag{3b}$$

$$\omega_{pi} = J_{pi}/T_i \tag{3c}$$

$$\begin{bmatrix} \dot{\xi} \\ \dot{\eta} \\ \dot{\zeta} \\ \dot{\chi} \end{bmatrix} = 1/2 \begin{bmatrix} -\zeta & -\chi & \eta & \xi \\ \chi & -\zeta & -\xi & \eta \\ \xi & \eta & \chi & \zeta \\ -\eta & \xi & -\zeta & \chi \end{bmatrix} \begin{bmatrix} \omega_{p1} \\ \omega_{p2} \\ \omega_{p3} \\ 0 \end{bmatrix} \tag{3d}$$

Here $\underline{T}$ is the torque and $\underline{J}$ the angular momentum. We obtain the components of $\underline{J}$ in the principal axis system by means of the rotation matrix A[1].

Dividing by the principal moments of inertia I_i then gives the principal components of angular velocity, which are used to obtain the time derivatives of the quaternion parameters. We re-write this last equation symbolically as

$$\dot{\underline{q}} = \underline{\underline{Q}}\, \underline{\omega}_p \qquad (3d)$$

this being an equation in a four-dimensional space.

We see immediately that (3a) has the same form as (1a) and can be integrated by a leapfrog analogous to (2a). However, the quaternion 'velocity' depends not only on $\underline{J}$ but also on the quaternions themselves, i.e. on the orientation, both directly in (3d) and through the rotation matrix. This is a situation where a leapfrog cannot be used. To see why consider the case of the simple first-order equation

$$\dot{x} = f(x,t)$$

A leapfrog for this equation would have the form

$$x^{n+1/2} = x^{n-1/2} + \Delta t\ f(x^n, t^n)$$

$$x^{n+1} = x^n + \Delta t\ f(x^{n+1/2}, t^{n+1/2})$$

where both the 'step' and 'mid-step' co-ordinates are required since x appears in the equation for x. The problem with this algorithm is that the step and mid-step equations are only weakly coupled through the velocity term; numerical errors cause them to decouple and we get two solutions which oscillate unstably about the correct solution. The remedy is to use the following method[6]. An auxiliary equation propagates x from n to n+1/2 using a first-order Taylor expansion

$$x^{n+1/2} = x^n + \frac{1}{2}\Delta t\ f(x^n, t^n)$$

and the main equation leapfrogs from n to n+1, employing $x^{n+1/2}$ in the velocity terms

$$x^{n+1} = x^n + \Delta t\ f(x^{n+1/2}, t^{n+1/2})$$

The first-order mid-step coordinate $x^{n+1/2}$ has an auxiliary role only and is not saved. Overall, the algorithm is second-order accurate.

We thus arrive at the following algorithm for rotational motion.

Auxiliary part

1. $\underset{\sim}{J}^n = \underset{\sim}{J}^{n-1/2} + \frac{1}{2}\Delta t\ \underset{\sim}{T}^n$

2. $\underset{\sim}{J}_p^n = \underset{=}{A}^n\ \underset{\sim}{J}^n$

3. $\omega_{pi}^n = J_{pi}^n / I_i$

4. $\underset{\sim}{q}^{n+1/2} = \underset{\sim}{q}^n + \frac{1}{2}\Delta t\ \underset{=}{Q}^n\ \underset{\sim}{\omega}_p^n$

5. Use $\underset{\sim}{q}^{n+1/2}$ to calculate $\underset{=}{A}^{n+1/2}$ and $\underset{=}{Q}^{n+1/2}$

Main part

6. $\underset{\sim}{J}^{n+1/2} = \underset{\sim}{J}^{n-1/2} + \Delta t\ \underset{\sim}{T}^n$

7. $\underset{\sim}{J}_p^{n+1/2} = \underset{=}{A}^{n+1/2}\ \underset{\sim}{J}^{n+1/2}$

8. $\omega_{pi}^{n+1/2} = J_{pi}^{n+1/2} / I_i$

9. $\underset{\sim}{q}^{n+1/2} = \underset{\sim}{q}^n + \Delta t\ \underset{=}{Q}^{n+1/2}\ \underset{\sim}{\omega}_p^{n+1/2}$

10. Store $\underset{\sim}{J}^{n+1/2}$ and $\underset{\sim}{q}^{n+1}$ for the next step.

The following table compares the energy conservations of this algorithm with the fourth-order predictor-corrector, and also with the constraint method. In the latter case the 'free-flight' phase uses a leapfrog, and so we might expect similar behaviour to the new algorithm. The quantity tabulated is the RMS fluctuation in total energy expressed as a percentage

of the fluctuation in potential energy, and the system is a three-centre model for cyclopropane[7].

Δt(fs)	6	8	10	12
New algorithm	2.6	4.7	9.7	20
Constraints	2.8	2.8	12.4	37
4th order p.c.	4.0	4.2	23.6	85

We see as expected that although the fourth-order algorithm is accurate at small time-steps, its errors increase much more rapidly as the time-step increases than do those of the suggested new algorithm, which is also better in this respect than the constraint algorithm. Further advantages of the new algorithm are that it is simple, needs minimal storage (three components of $\underline{J}$ and four of $\underline{q}$) and is self-starting.

Finally, we note that diatomics are a special case. If the constraint method is used the equation for the single constraint force is a quadratic and hence exactly soluble, whereas in the polyatomic case the equations of constraint must be linearised and iterated. On the other hand, the four quaternions with the usual single constraint over-determine the problem since there are only two degrees of freedom. For these reasons we anticipate that the constraint method may be superior for diatomics.

References

1. D.J. Evans, Molec. Phys., 34, (1977) 317.
2. D.J.Evans, S. Murad, Molec. Phys., 34, (1977) 327.
3. W.F. Van Gunsteren, H.J.C. Berendsen, Molec. Phys., 34, (1977) 1311.
4. C.W. Gear, Numerical Initial Value Problems in Ordinary Differential Equations (Prentice Hall, 1971).
5. K. Singer, J.V.L. Singer, A.J. Taylor, Molec. Phys., 37, (1979) 1239.
6. D. Potter, Computational Physics (John Wiley, 1973).
7. Y. Guissani, D. Fincham, to be published.

MANY PARTICLE MOLECULAR DYNAMICS METHODS

D.M. Heyes

In a Molecular Dynamics, MD, program the number of distinct pair separations is N(N-1)/2 for a MD cell containing N molecules. Usually the range of the interaction potential is limited to several molecular diameters so that a spherical truncation can be applied when evaluating the interactions. However, in a conventional program the nearest-image transformation must be applied to each pair in the system, and their separation determined, before the "cut-off" is applied. Thus the computer time contains a component which increases quadratically with N. Consequently, an upper limit of approximately 1000 molecules is usual. Further, on a vetor processor the spherical truncation would inhibit vectorisation[1] so that the energy and forces as well must be evaluated for each pair in the system, even though when N is large most of them will be out of range.

Two methods which enable a large number of moleculaes in the MD cell to be simulated are discussed below. In these schemes the computer time increases approximately linearly with time because they use ways of pre-eliminating the more distant, non-contributing, interactions before the inter-particle separations are calculated.

The first method establishes a table of neighbours for each particle at time steps of roughly equal interval (typically every 10)[1]. This table is used at intervening time steps to determine those molecules with a chance of being within the truncation radius of each particle. Unfortunately this method has considerable memory requirements. The table of neighbours needs ~ 45 N words on using a truncation radius of 2.5σ and $r_2 = 2.94\sigma$ [see ref.(1)] on a Lennard-Jones liquid near the triple point.

In the second method the molecules are assigned to NL smaller subcells or link cells which completely fill the original MD cell[2,3]. The minimum side length of a link cell equals the truncation range of the interactions. Particles within a link cell only then need to interact with those in the first shell of link cells about it. The contents of each link cell are efficiently obtained through an array, LINK, of dimension N, which contains a consecutive series of closed chains of particle indices,

each chain being associated with a particular link cell. Each element in LINK contains the index of the next particle in the chain. The periodic boundary conditions are applied to the co-ordinates of each link cell and then automatically to their contents. The computer memory requirements of this method are rather modest. A part from array LINK(N), two more arrays of dimension NL are the only others needed to incorporate this method.

Using the Lennard-Jones MD program MDATOM[4] on the state point $\rho^* = 0.8552 = N\ \sigma^3/V$ and $kT/\varepsilon = 0.7053$ the speeds of the three methods are as follows.

Average c.p.u. time on the ULCC CDC 7600 to perform a time-step

Method	N:	108	256	864	2048	6912
Conventional method		0.036	0.163	1.573	8.460	94.173
Neighbourhood tables		-	0.093	0.310	*	*
(Γ_2 = 1.001 nm, see ref.(1))						
Link cells		-	-	0.834	2.321	6.666
(NL)				(64)	(125)	(512)

- untried, * I.c.m. memory exceeded

The conventional method is too time consuming for N larger than approximately 1000. Although the neighbourhood lists method is much faster than the conventional method, it too is not practicable for N greater than 1000 because of the considerable memory requirements. Even if this is overcome, the approximately one-in-ten time steps at which all interactions must be considered in order to create the neighbour lists, would be a prohibitive factor in its implementation. Timings for the conventional method indicate the long time required for this albeit infrequent operation.

The link cell method provides rather unspectacular gains in speed for moderately large samples of N ~ 1000 to 2000, but becomes a progressively more attractive method when the very large N values of ~ 7000 are considered.

REFERENCES

1. D. Fincham and B.J. Ralston, Comp. Phys. Commun. 23, (1981) 127-134.

2. Drs. F. von Swol and L.V. Woodcock of The Laboratory for Physical Chemistry, University of Amsterdam, The Netherlands, are thanked for their aid in programming the link cell method.

3. R.W. Hockney, in "Daresbury Laboratory: Information Quarterly for MD and MC Simulations", pp.22-23.

4. D. Fincham, Comp. Phys. Commun. 21, (1980) 247-256.