JOURNAL OF COMPUTATIONAL PHYSICS 22, 451–458 (1976)

On the Computation of Correlation Functions in Molecular Dynamics Experiments

E. KESTEMONT AND J. VAN CRAEN*

Faculty of Sciences, Free University of Brussels, Brussels, Belgium

Received February 9, 1976; revised May 17, 1976

The method described by Futrelle and McGinty for the high speed computation of correlation functions, starting from the results of molecular dynamics experiments, is discussed and comparisons of computation times with the "direct" method are given. A further improvement of the method is described.

1. INTRODUCTION

The development of large computers during the last two decades has made it possible to solve numerically the equations of motion of particles subjected to given intermolecular forces, thereby opening a wide range of new possibilities in the study of molecular models [1, 2]. As the method of molecular dynamics makes available all the characteristics of the trajectory of the system in phase space, one can easily compute all its time-dependent properties and hence test the validity of the existing theoretical developments. At the present time, one can simulate not only simple systems like argon [3] and carbon oxide [4] but also more complicated ones like water [5], butane [6], and so on.

A computer experiment usually proceeds in two steps [3]. With particle geometry and forces specified, the equations of motion are solved starting from well-chosen initial conditions; the time evolution of positions, velocities, angular momentum, etc., is recorded. The second step consists in the computation of the correlation functions.

The relations between macroscopic properties of gases and liquids and correlation functions is well understood in many cases at present. It suffices, for example, to recall the link between the self-diffusion coefficient, the NMR line-width, or the dielectric permittivity and, respectively, the velocity [7], the angular momentum [8], and dipole–dipole [9] correlation functions.

Zwanzig and Ailawadi [10] pointed out that, in order to obtain reliable infor-

* Fellow of the National Foundation of Scientific Research, Belgium.

mation over the system under study, one not only has to extend the number of interacting particles as much as possible in order to approach a "macroscopic" system, but one also has to follow this system for times that are long compared to its internal characteristic times.

On the other hand, recent theoretical developments concerning the long time behavior of the velocity autocorrelation function for fluids [11, 12] ask for a better knowledge of the tail of this function.

Much work and time have been devoted to reducing the amount of computer time needed in the first step of the molecular dynamics methods, by choosing the more efficient algorithm to solve the equations of motion. Futrelle and McGinty [13] have shown how the use of fast Fourier transform techniques, developed in connection with signal processing [14, 15], can lead to a substantial computer time reduction in the determination of correlation functions.

Unfortunately, this technique seems to have been overlooked by many people working in the field of molecular dynamics. The aim of this paper is to present a detailed comparison of the "direct" and "FFT" methods for computing autocorrelation functions.

We confine ourselves to the velocity autocorrelation function although the method can be used to compute any other correlation function as well.

2. The Method

The velocity autocorrelation function is defined by

$$g(t) = C\langle \mathbf{v}_i(t) \cdot \mathbf{v}_i(t' + t)\rangle. \tag{1}$$

Here $\mathbf{v}_i(t)$ is the velocity of particle i at time t, $\langle \cdots \rangle$ denotes an equilibrium ensemble average and the normalization constant C ensures that $g(0) = 1$. According to ergodic theory the ensemble average can be replaced by a time average leading to

$$g(t) = C \lim_{T\to\infty} (1/T) \int_0^T dt'\, \mathbf{v}_i(t') \cdot \mathbf{v}_i(t' + t). \tag{2}$$

As a consequence of the convolution theorem the power spectrum of $g(t)$ is given by the squared modulus of the Fourier transform of $\mathbf{v}(t)$ [16].

If the velocities of the ensemble of NP particles are only available over N equal time intervals Δt then this last expression can be approximated by

$$g(l\,\Delta t) = (C/(N - l)) \sum_{n=0}^{N-l-1} \mathbf{v}_i(n\,\Delta t) \cdot \mathbf{v}_i[(n + l)\,\Delta t], \qquad l = 0, 1, \ldots, N - 1. \tag{3}$$

The additional weight factor $1/(N - l)$ accounts for the different number of products contributing to each $g(l\,\Delta t)$.

It should be noted that the spacing Δt used in the computation of $g(l\,\Delta t)$ does not necessarily correspond to the time increment used to integrate the equations of motion but rather to an appropriate multiple of it: choosing Δt too small would result in highly correlated contributions to $g(l\,\Delta t)$, hence increasing the computation time without improving proportionally the precision of the results [11].

The identical role played by the NP particles in an equilibrium situation enables us to average $g(l\,\Delta t)$ over the entire system instead of observing the velocity of a single particle. One then obtains

$$g(l\,\Delta t) = (C/NP(N-l)) \sum_{i=1}^{NP} \sum_{\alpha=1}^{3} \sum_{n=0}^{N-l-1} v_{i\alpha}(n\,\Delta t)\, v_{i\alpha}[(n+l)\,\Delta t], \tag{4}$$

where α labels the x, y, z components of the velocities.

In many modern molecular dynamics experiments N is chosen so that $N\,\Delta t$ is much larger than the mean free time [11] τ:

$$N\,\Delta t/\tau \simeq 100\text{–}200. \tag{5}$$

Eventually an additional average of (4) is made over a set of different initial states [11].

Expression (4) is usually computed in a straightforward way requiring a computation time proportional to N^2. A much faster method based on the convolution theorem combined with a fast Fourier transform algorithm has been described by Futrelle and McGinty [13].

It proceeds as follows. Let us focus on the contribution to $g(l\,\Delta t)$ of one particular component of the velocity of one particle, denoted by v_1,

$$g_1(l\,\Delta t) = (C/(N-l)) \sum_{n=0}^{N-l-1} v_1(n\,\Delta t)\, v_1[(n+l)\,\Delta t]. \tag{6}$$

Let us introduce the discrete Fourier transform (DFT) of the velocity, defined by

$$\tilde{v}(k\,\Delta\omega) = \sum_{n=0}^{N-1} v(n\,\Delta t) \exp(in\,\Delta t\, k\,\Delta\omega), \tag{7}$$

where $\Delta\omega = 2\pi/N\,\Delta t$; with the help of the convolution theorem one obtains, after Fourier inversion,

$$g_1(l\,\Delta t) = (C/(N-l))(1/N) \sum_{k=0}^{N-1} |\,\tilde{v}(k\,\Delta\omega)|^2 \exp(-il\,\Delta t\, k\,\Delta\omega), \tag{8}$$

where the continuation of v_1 with period $T = N\,\Delta t$ is used. However, this procedure, as it stands, introduces spurious correlations in $g(l\,\Delta t)$. This can be avoided

by simply adding a set of N zeros to the known values of $v_1(n\,\Delta t)$ and extending T to $2N\,\Delta t$; the sum in (7) now contains $2N$ terms and (8) becomes

$$g_1(l\,\Delta t) = (C/(N-l))(1/2N)\sum_{k=0}^{2N-1} |\,\tilde{v}(k\,\Delta\omega)|^2 \exp(-il\,\Delta t\,k\,\Delta\omega). \tag{9}$$

Finally taking account of the linear character of the Fourier transform, $g(l\,\Delta t)$ is simply given by summation of g_1-like terms over all particles and velocity components.

The transforms are made by a fast Fourier transform algorithm [17, 15]: the number of arithmetic operations for the whole procedure is then of order $N \log_2 N$ (see Section 4).

A further improvement over the method of Futrelle and McGinty can be obtained by noting that the FFT transform of two real velocity components can be computed simultaneously. Indeed, let $\alpha = T(v_1)$ and $\beta = T(v_2)$, respectively, be the DFT of v_1 and v_2. Introducing

$$\gamma(k\,\Delta\omega) = T(v_1 + iv_2) = \alpha(k\,\Delta\omega) + i\beta(k\,\Delta\omega) \tag{10}$$

one immediately sees that

$$\begin{aligned} |\,\gamma(k\,\Delta\omega)|^2 = {} & |\,\alpha(k\,\Delta\omega)|^2 + |\,\beta(k\,\Delta\omega)|^2 \\ & + i[\alpha(-k\,\Delta\omega)\,\beta(k\,\Delta\omega) - \alpha(k\,\Delta\omega)\,\beta(-k\,\Delta\omega)]. \end{aligned} \tag{11}$$

The last two terms on the right-hand side of Eq. (11) represent the Fourier frequency components of the cross correlations between independent velocities which for a large system in equilibrium average to zero. For smaller systems, however, their contributions to $|\,\gamma(k\,\Delta\omega)|^2$ is nonzero. Simply symmetrizing $|\,\gamma(k\,\Delta\omega)|^2$,

$$|\,\gamma_S(k\,\Delta\omega)|^2 = \tfrac{1}{2}\{|\,\gamma(k\,\Delta\omega)|^2 + |\,\gamma(-k\,\Delta\omega)|^2\}, \tag{12}$$

before taking the inverse DFT (IDFT) would lead to the right result. However, this last step is unnecessary when considering a real function $g(t)$ as can be seen by direct inspection of the IDFT of $|\,\gamma(k\,\Delta\omega)|^2$. The IDFT of the two spurious terms of (11), leads to a complex vector with a rigorously zero real part. Consequently, their contribution to the real part of $g(l\,\Delta t)$ is zero.

3. Features of the Model

The input data used for testing the program originated from a molecular dynamics experiment on an ensemble of rough spheres. This model is of particular interest as it is perhaps the simplest one showing a transfer of rotational to trans-

lational energy. It is hoped that, for such a problem as rotational relaxation, it can play the same role as the hard sphere model does in the approach of simple liquids.

This model has already been used to study the dipole–dipole autocorrelation function and the dielectric permittivity in two and three dimensions [18, 19]. Its full description as well as the connection with the rotational J diffusion model [20, 21] is out of the scope of this paper. We merely use this model as a base of comparison of the computation time for the autocorrelation function either when straightforward calculations are performed or when the Fourier transform algorithm described above is used.

For completeness we mention that the system consists of 108 particles placed in a cubic box with the usual periodic boundary conditions [19], which is followed in time over 3000 time intervals.

4. The Computer Program

The general outline of the program, which was written for a CDC 6400 computer in Fortran IV Extended, is as follows.

A table of the N required complex exponentials is constructed by a high speed, high accuracy algorithm [22].

The information about the dynamics, i.e., the velocity components of the particles, is transferred from tape or disc to the high speed central memory of the computer. Considerations on internal data treatment and available memory core lead to an optimum of the velocity components which are simultaneously read in. Note that every component has to be known over the complete time interval.

A complex vector of length $2N$ is constructed, the first N elements of which contain the entire information about two independent velocity components in their real and imaginary parts, respectively. The last N elements are filled up by zeros.

This complex vector is transformed using a FFT subroutine based on the algorithm of Cooley and Tukey [17] as described by Singleton [22].

The next step then consists in taking the squared modulus of each component $\gamma(k\,\Delta\omega)$.

The whole procedure is repeated until all velocity components of all particles have been treated.

The FFT algorithm quoted in [22] has the peculiarity of displaying the Fourier components $\gamma(k\,\Delta\omega)$ in the "bit reversal" order. A reordering of the terms is, however, not needed after each transformation. It suffices to reorder the final vector only once, just before taking the IDFT; the real part of the resulting time-dependent vector corresponds to the required correlation function after suitable normalization.

5. Discussion

The specific FFT algorithm which is used operates on vectors of dimension $N = 2^m$. We have made a comparative study of the computation times for $g(l\,\Delta t)$ for values of m ranging from 6 to 11. These results, expressed in seconds (central processor time) are given in Table I.

TABLE I

Computation Times of $g(m\Delta t)$ in Seconds (Central Processor Time) with $N = 2^m$ as the Number of Points

m	N	Straightforward computation	FFT technique
6	64	27	21
7	128	88	41
8	256	306	89
9	512	1143	182
10	1024	4412	364
11	2048	—	760

As quoted above, straightforward computation of N values of the correlation function by expression (4) requires times proportional to N^2, whereas the corresponding computation time, using a FFT technique with $N = 2^m$, is essentially proportional to mN [14, 17]. More precise values for the total number of floating point operations, when treating the three-dimensional case with NP particles are quoted in Table II. When comparing these values to the formulas established by

TABLE II

Number of Real Operations[a] in Computing $g(m\Delta t)$

	Straightforward computation	FFT technique
Real multiplications	$1.5N(N+1)NP$	$4(1.5NP+1)[(m-1)N+1]$
Real additions	$1.5N(N-1)NP$	$2(1.5NP+1)[(3m+1)N+1]$

[a] Not including the arithmetic operations needed in addressing.

Bergland [23], one should keep in mind that here $(3NP/2) + 1$ vectors of dimension $2N$ are transformed. Examination of Tables I and II shows the important gain in computer time this method pemits.

It is to be stressed, however, that one is not always interested in knowing the correlation function over long times; intervals of roughly three of four characteristic decay times should, in many applications, be sufficient. Long time study of the dynamical behavior of the system is then done only in order to increase the accuracy through better statistics. In these situations the computation time of $g(l\,\Delta t)$ in the straightforward calculation decreases drastically. Indeed if NM is the maximum number of values for which one wishes $g(l\,\Delta t)$ to evaluate, then the number of arithmetic operations reduces to $3NP$, $(N \times NM - (NM^2/2) + (NM/2))$ floating point multiplications, and $3NP$, $(N \times NM - (NM^2/2) - (NM/2))$ additions. For a typical situation, where $N = 2048$ and $NM = 200$ one is left with approximately 1.26×10^8 multiplications (and a similar number of additions) compared to 6.80×10^8 for $NM = 2048$; the actual computation takes 3480 sec; this last value is still 4.6 times larger than the time required for the complete computation of $g(l\,\Delta t)$ using the FFT technique.

For these calculations of $g(l\,\Delta t)$ with limited NM, the computation time, which goes linearly with N, will *ultimately* be shorter than the time required when using the FFT technique. Straightforward computation will be faster for $m \equiv \log_2 N$ greater than $NM/2$. If one wants to know $g(l\,\Delta t)$ over say 50 time intervals this will happen when

$$N > 2^{25} \simeq 3.4 \times 10^7.$$

In this context, what is perhaps a more important limitation is the core storage requirements for very long runs. The method as it stands implies that the complete time span of only two dynamical variables has to be held simultaneously in storage. Should this be impossible, then an auxiliary memory technique of the form described by Singleton [24] must be used.

A last aspect, important for the comparison of these two methods, is the question of errors. It appears that CDC-60 bits arithmetics leads in our test case to a maximum absolute discrepancy of 10^{-11} between the results of the two methods; this is many orders of magnitude inferior to the statistical errors [11]. So, as far as errors are concerned, the methods are equivalent.

In conclusion, one can say that the FFT method, combined with a proper choice of the interval between two "time origins," appears to be accurate and very fast; even in these applications, where the autocorrelation function is computed over a time interval which is short compared to the total time over which the dynamical variables of the system are known, the method keeps its advantageous features.

REFERENCES

1. B. ALDER AND T. WAINWRIGHT, *Phys. Rev. Lett.* **18** (1967), 988.
2. B. J. BERNE AND G. D. HARP, "Advances in Chemical Physics" (I. Prigogine and S. A. Rice, eds.), Vol. 17, Interscience, New York, 1970.
3. A. RAHMAN, *Phys. Rev. A* **136** (1964), 405.
4. G. D. HARP AND B. J. BERNE, *J. Chem. Phys.* **49** (1968), 1249.
5. A. RAHMAN AND F. H. STILLINGER, *J. Chem. Phys.* **55** (1971), 3336.
6. J. P. RYCKAERT AND A. BELLEMANS, *Chem. Phys. Lett.* **30** (1975), 123.
7. R. KUBO, *J. Phys. Soc. Japan* **12** (1957), 1570. R. KUBO, "Reports on Progress in Physics" (A. C. Stickland, ed.), Vol. 29, Inst. Phys., London, 1966.
8. R. G. GORDON, "Advances in Magnetic Resonance," (J. S. Waugh, ed.), Vol. 3, Academic Press, New York, 1968.
9. R. H. COLE, *J. Chem. Phys.* **27** (1957), 33.
10. R. ZWANZIG AND N. K. AILAWADI, *Phys. Rev.* **182** (1969), 280.
11. W. W. WOOD, "Fundamental Problems in Statistical Mechanics" (E. G. D. Cohen, ed.), Vol. 3, North-Holland, Amsterdam, 1975.
12. Y. POMEAU AND R. RÉSIBOIS, *Phys. Rep.* **19C** (1975), 64.
13. R. P. FUTRELLE AND D. J. MCGINTY, *Chem. Phys. Lett.* **12** (1971), 285.
14. See the special issue of *IEEE Trans. Audio Electroacoust.* **AU 15** (1967), 2.
15. E. ORAM BRIGHAM, "The Fast Fourier transform," Prentice-Hall, Englewood Cliffs, N.J., 1974.
16. J-P. HANSEN AND M.-L. KLEIN, *J. Phys. (Paris)* **35 L** (1971), 29.
17. J. W. COOLEY AND J. W. TUKEY, *Math. Comp.* **19** (1965), 297.
18. E. KESTEMONT AND A. BELLEMANS, *J. Computational Phys.* **7** (1971), 515.
19. E. KESTEMONT, to appear, 1976.
20. R. G. GORDON, *J. Chem. Phys.* **44** (1966), 1830.
21. R. E. D. MCCLUNG, *J. Chem. Phys.* **55** (1971), 3459.
22. R. C. SINGLETON, *Comm. ACM* **10** (1967), 647.
23. G. D. BERGLAND, *Math. Comp.* **22** (1968), 275.
24. R. S. SINGLETON, *IEEE Trans. Audio Electroacoust.* **AU 15** (1967), 91.