Efficient on the fly calculation of time correlation functions in computer simulations

Jorge Ramírez,[1,a)] Sathish K. Sukumaran,[2] Bart Vorselaars,[3] and Alexei E. Likhtman[3]
[1]*Departamento de Ingeniería Química, ETSI Industriales, Technical University of Madrid, José Gutiérrez Abascal 2, 28006 Madrid, Spain*
[2]*Graduate School of Science and Engineering, Yamagata University, 4-3-16 Jonan, Yonezawa, Yamagata 992-8510, Japan*
[3]*Department of Mathematics, University of Reading, Whiteknights, P.O. Box 220, Reading RG6 6AX, United Kingdom*

(Received 8 April 2010; accepted 31 August 2010; published online 15 October 2010)

Time correlation functions yield profound information about the dynamics of a physical system and hence are frequently calculated in computer simulations. For systems whose dynamics span a wide range of time, currently used methods require significant computer time and memory. In this paper, we discuss the multiple-tau correlator method for the efficient calculation of accurate time correlation functions on the fly during computer simulations. The multiple-tau correlator is efficacious in terms of computational requirements and can be tuned to the desired level of accuracy. Further, we derive estimates for the error arising from the use of the multiple-tau correlator and extend it for use in the calculation of mean-square particle displacements and dynamic structure factors. The method described here, in hardware implementation, is routinely used in light scattering experiments but has not yet found widespread use in computer simulations. © *2010 American Institute of Physics*. [doi:10.1063/1.3491098]

I. INTRODUCTION

According to the fluctuation-dissipation theorem, the instantaneous equilibrium fluctuations, of long wavelength and low frequency, of a dynamical function $f(t)$ around its average, $\Delta f(t)=f(t)-\langle f\rangle$, relax to the equilibrium value (the average), following the same laws as macroscopic small disturbances (for instance, see Ref. 1). Upon closer inspection, the instantaneous value of $\Delta f(t)$ appears chaotic. However, interesting, nonchaotic information about the dynamics of the system can be extracted by considering the autocorrelations of these fluctuations at different times, $\langle \Delta f(t)\Delta f(0)\rangle$. The sign $\langle\ \rangle$ here represents an average over all possible initial conditions of the system. Invoking the ergodic hypothesis, the ensemble average can be replaced by the time average (a definition is provided later).

Time correlation functions (TCFs) reveal relaxation properties of physical systems and consequently are often calculated in computer simulations.[2,3] According to theory, TCFs calculated at equilibrium are directly related to the linear response functions and using the Green–Kubo relations yields the transport coefficients as integrals over the equilibrium TCF. However, the calculation of TCF consumes significant CPU time and memory resources, especially if the fluctuations decay over many decades in time. In this paper, after a brief review of the calculation of time correlation functions mainly to establish notation, we present an algorithm that allows accurate calculation of TCF on the fly during computer simulations, without excessive additional memory and computational cost. The algorithm is based on a well known multiple-tau correlator method frequently used in dynamic light scattering experiments.[4–6] The main idea behind the method is the use of averages for data storage. For the calculation of the time correlation function between two points at a given time lag, one can average the data over some interval around the two points without significant loss of information, if the size of the averaging interval is sufficiently smaller than the distance between the points. This averaging process considerably reduces the amount of storage required to calculate the full TCF. However, if the simulation spans a wide range of time scales, to accurately calculate the TCF, we need to vary the averaging size. This idea of varying the averaging time is naturally included in the multiple-tau correlator method that we analyze and extend in this paper. The algorithm in its smoothing version has already been used successfully in a series of papers by the same authors.[7–12] The aim of this paper is to draw the attention of the wider computer simulation community to this efficient method for calculating TCF and to analyze the error due to the use of this algorithm. In addition, we show how the method can be easily adapted for the calculation of both the mean-square particle displacement and the single chain dynamic structure factor during computer simulations.

II. METHODS

A. Standard calculation of time correlation functions

In general, given a simulation trajectory of duration T, the time autocorrelation of a dynamical function $f(t)$ can be expressed as

[a)]Electronic mail: jorge.ramirez@upm.es.

$$F(\tau) = \frac{1}{T-\tau}\int_0^{T-\tau} f(t+\tau)f(t)dt, \tag{1}$$

where τ is called the lag time. In a simulation with $N-1$ steps of length Δt and trajectory length T, such that $T=(N-1)\Delta t$, we can discretize Eq. (1) as

$$F_j = \frac{1}{N-j}\sum_{i=0}^{N-j-1} f_i f_{i+j}, \tag{2}$$

where $F_j \equiv F(j\Delta t)$ and $f_i \equiv f(i\Delta t)$. Due to the typically large value of N (currently simulations with $N \approx 10^7-10^8$ are common and this value is only expected to increase in the future), it is usually not possible to store all the values of f_i in memory. This is especially true if one has to calculate a large number of correlation functions, for example, several correlation functions per particle (typically 10^4-10^6). Therefore, these correlations are usually calculated by postprocessing the data collected during the simulation and stored in a disk file. The number of operations required to calculate a single value of the time correlation function F_j scales linearly with N. Sometimes the TCF can contain useful information over a very wide range of time scales, and the correlation function may have a low but still relevant value at late times (for a particular example from polymer dynamics, we refer the reader to Ref. 10). To improve accuracy and statistical efficiency, it is necessary to include as many of the available values of f_i as possible while calculating the TCF. Typically, the desired number of points in the TCF F_j is known, say M. M will depend on the expected features of the TCF, and the points F_j will be either linearly or logarithmically distributed in time. Thus, the total number of operations grows as NM. This number can be reduced significantly if the values of the function f_i are either collected only every γ steps or only the average over γ steps is stored. However, this will make information about the correlation function unavailable for $t < \gamma\Delta t$ and will lead to errors for $t \approx \gamma\Delta t$ [an example of this can be observed in the stress relaxation from molecular dynamics (MD) simulations of polymers[10,12]]. The main problem is that all N values of f_i should ideally be in memory at the time of calculation of the TCF, which is not usually possible due to the large value of N. Considering that the data need to be read from a disk, possibly several times, the calculation of a time correlation function can be excruciatingly slow. In addition, it is common to calculate more than one correlation function from a single simulation, so the computational and memory load may be several times higher. For example, in MD simulations it is frequently useful to analyze the mean-square displacement of every single particle, so a TCF is needed for each particle in the system, and this number may be well over 10^5. It would be very useful to have a method that deals with the array f_i and calculates the TCF F_j on the fly and therefore obviates the need for repeatedly accessing the disk.

It is also germane to comment on the accuracy of the time correlation functions calculated with Eqs. (1) and (2). According to the ergodic hypothesis, Eq. (1) is exact only in the limit $T \to \infty$. The finiteness of the simulation run introduces a difficult to avoid inaccuracy in the calculated value of the TCF. A second source of inaccuracy comes from the size of the discrete time-step Δt used in the computer simulation and the time-stepping algorithm used to integrate the equations of motion. However, the results are usually satisfactory as long as both Δt and T are far from the characteristic time scale of the dynamical processes of interest τ_0, i.e., as long as $\Delta t \ll \tau_0 \ll T$. In this work, we suggest a numerical algorithm to calculate, in as efficient and accurate way as possible, the TCF from a simulation where the time-step size, the time-stepping algorithm, and the simulation length have been fixed.

B. Time correlation of a time-averaged function

Suppose we have N observations of the function $f(t)$ at equidistant points in time $t=0,\Delta t,\ldots,(N-1)\Delta t$, which we will denote by $f_i = f(i\Delta t)$. Then, the time correlation function at the time $\tau = j\Delta t$ can be calculated by means of Eq. (2). Now let us consider the correlation function between the function f averaged over k neighboring points,

$$\bar{f}_{i,k} = \frac{1}{k}\sum_{j=0}^{k-1} f_{i+j}, \tag{3}$$

$$\begin{aligned}\bar{F}_{j,k} &= \frac{1}{N-j-k+1}\sum_{i=0}^{N-j-k} \bar{f}_{i,k}\bar{f}_{i+j,k} \\ &= \frac{1}{k^2(N-j-k+1)}\sum_{i=0}^{N-j-k}\sum_{p=0}^{k-1}\sum_{q=0}^{k-1} f_{i+p}f_{i+j+q}.\end{aligned} \tag{4}$$

In the limit of large $N \gg j$, one can neglect small differences in the upper limits and obtain

$$\bar{F}_{j,k} \approx \frac{1}{k^2}\sum_{p=0}^{k-1}\sum_{q=0}^{k-1} F_{j+q-p} = \frac{1}{k^2}\sum_{p=-k+1}^{k-1}(k-|p|)F_{j+p}. \tag{5}$$

This simple formula shows that calculating the correlation function between quantities averaged over k points is equivalent to a smoothing of the correlation function over the interval $\pm(k-1)$ with the triangular smoothing function $(k-|p|)$. In the limit of large k the sums can be replaced by integrals

$$\bar{F}(\tau,\lambda) = \frac{1}{\lambda^2}\int_{-\lambda}^{\lambda}(\lambda-|t|)F(\tau+t)dt,$$

where $\lambda = k\Delta t$ is the averaging time.

In the limit $\lambda \ll \tau$ the correlation function inside the integral sign can be expanded in Taylor series,

$$F(\tau+t) \approx F(\tau) + F'(\tau)t + F''(\tau)t^2/2 + \ldots .$$

Using this expansion and the integral above, we can get an estimate for the relative error introduced by averaging over an interval λ,

$$\Delta(\tau,\lambda) \equiv \frac{F(\tau)-\bar{F}(\tau,\lambda)}{F(\tau)} \approx \frac{\lambda^2}{12}\frac{F''(\tau)}{F(\tau)}. \tag{6}$$

In other words, the approximation is good if the second derivative of the correlation function is not too large, and the error is quadratic in the averaging time λ. The typical correlation functions can be locally approximated by either a power-law or an exponential, so it is interesting to calculate the error for these two ideal cases.

In case of a power-law relaxation $F(\tau)=c\tau^{-\alpha}$, where α is the negative slope of $F(\tau)$ on a log-log scale, we obtain from Eq. (6)

$$\Delta(\tau,\lambda)=\frac{\alpha(\alpha+1)}{12}\left(\frac{\lambda}{\tau}\right)^2, \quad \text{power-law relaxation.} \tag{7}$$

If λ is a small fraction of τ, this is uniformly small providing α is not too big.

In the case of an exponential relaxation $F(\tau)=c\exp(-\tau/\mu)$, where μ is the relaxation or terminal time, we obtain

$$\Delta(\tau,\lambda)=\frac{1}{12}\left(\frac{\lambda}{\mu}\right)^2, \quad \text{exponential relaxation.} \tag{8}$$

If the averaging time λ is proportional to the lag time τ, i.e., $\lambda=\beta\tau$ with $\beta<1$, this error can become arbitrary large when $\tau>\mu/\beta$. However, the signal at the times much larger than the terminal time μ becomes exponentially small, and it is never realistic to get any signal for $\tau>6\mu$ or so because of statistical error.

C. Mean-square displacement

Apart from correlation functions, one can use the same idea of preaveraging for the calculation of the average mean-square displacement of a selected particle with position $r(t)$,

$$g_j=\frac{1}{N-j}\sum_{i=0}^{N-j-1}\langle(r_{i+j}-r_i)^2\rangle.$$

Similarly to the previous section, let us define the time-averaged position of the particle and its mean-square displacement,

$$\bar{r}_{i,k}=\frac{1}{k}\sum_{j=0}^{k-1}r_{i+j}, \tag{9}$$

$$\begin{aligned}\bar{g}_{j,k}&=\frac{1}{N-j-k+1}\sum_{i=0}^{N-j-k}(\bar{r}_{i+j,k}-\bar{r}_{i,k})^2\\&=\frac{1}{k^2(N-j-k+1)}\sum_{p=0}^{k-1}\sum_{q=0}^{k-1}\sum_{i=0}^{N-j-k}(r_{i+j+p}-r_{i+p})\\&\quad\times(r_{i+j+q}-r_{i+q}).\end{aligned} \tag{10}$$

To simplify this expression, let us consider an auxiliary function,

$$w_j=2\sum_{i=0}^{N-j-1}r_{i+j}r_i=\sum_{i=0}^{N-j-1}r_{i+j}^2+\sum_{i=0}^{N-j-1}r_i^2-\sum_{i=0}^{N-j-1}(r_{i+j}-r_i)^2.$$

The first two terms in the last expression differ by

$$\sum_{i=0}^{N-j-1}r_{i+j}^2-\sum_{i=0}^{N-j-1}r_i^2=\sum_{i=N-j+1}^{N-1}r_i^2-\sum_{i=0}^{j-1}r_i^2.$$

If say a simple random walk starts from 0, the first term will be of order $j\langle\Delta r^2\rangle$ (here Δr is the stepsize of the random walk), whereas other terms in the previous equation are of order of $N\langle\Delta r^2\rangle$. Thus, if $j\ll N$, this difference can be neglected,

$$\sum_{i=0}^{N-j-1}r_{i+j}^2\approx\sum_{i=0}^{N-j-1}r_i^2\approx\sum_{i=0}^{N-1}r_i^2.$$

Similarly, we can neglect differences in the upper limit of all other sums,

$$2\sum_{i=0}^{N-j-1}r_{i+j}r_i\approx 2\sum_{i=0}^{N-j-1-p}r_{i+j+p}r_{i+p}=w_i.$$

Thus, Eq. (10) can be rewritten as

$$\begin{aligned}\bar{g}_{j,k}&=\frac{1}{2k^2}\sum_{p=0}^{k-1}\sum_{q=0}^{k-1}(w_{q-p}-w_{j+p-q}-w_{j+q-p}+w_{p-q})\\&=\frac{1}{k^2}\sum_{p=-k+1}^{k-1}(k-|p|)g_{j+p}-\frac{1}{k^2}\sum_{p=-k+1}^{k-1}(k-|p|)g_p,\end{aligned} \tag{11}$$

where we took into account time reversibility, $g_{-p}=g_p$. The first sum of this equation is exactly the same as in Eq. (5) for the correlation function. However, in addition to it there is an extra correction term, which depends on the mean-square displacement at smaller time scales. Thus, when calculating the mean-square displacement from averaged positions, each correlation needs to be corrected by this term. This correction becomes especially important in case the mean-square displacement achieves an (almost) constant value as a function of time.

Yet a different way to calculate the mean-square displacement is by not resorting to averaging. In this case, one sacrifices statistical accuracy for systematic accuracy—this will be discussed in Sec. II F.

D. Algorithm proposed by Frenkel

Frenkel[3] noted the importance of having a scheme for measuring time correlation functions which allows for adjustable sampling frequencies. In a quest for an algorithm to calculate time correlation functions with a number of operations that scales linearly with N and modest memory requirements, Frenkel derived a scheme based on time averages. Frenkel's approach is explained in detail in his book.[3] However, for the sake of completeness, we summarize it using the notation of this paper.

We can define a hierarchical structure of block averages of the function f in the following manner:

$$f_i^{(l,m)}=\frac{1}{m}\sum_{j=(i-1)m+1}^{im}f_j^{(l-1,m)}, \tag{12}$$

with $f_i^{(0,m)}=f_i$. Equation (12) is a recursive relation between block averages of levels l and $l-1$, where m determines the size of the average [$f_i^{(l,m)}$ is equivalent to $\bar{f}_{i,k}$ of Eq. (4) with $k=m^l$]. This way, the past values of the function are stored with a resolution that decreases with the lag time (i.e., the time between current time and the time when the function was calculated: see the left picture of Fig. 1 for a schematic depiction of the data structure in Frenkel's correlator) and the

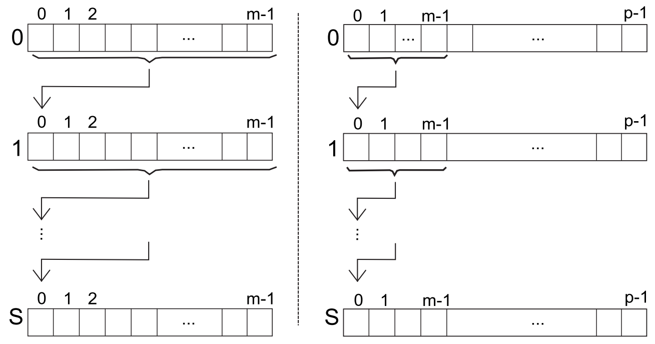

FIG. 1. Schematic view of the data structure the correlator proposed by Frenkel (left) and the multiple-tau correlator reviewed in this paper (right). Data arrays at different levels are represented, and the arrow symbolically represents the averaging and transfer of data between different levels of the correlator.

time correlation function can be calculated using the averaged values.

The required storage at level l is m per correlation function. The total required storage for a simulation of length $T = m^l \Delta t$ (or $N = m^l$) is lm, whereas the requirements in the case of the conventional approach introduced in Sec. II A would be m^l. As explained by Frenkel,[3] at each time-step we need to update $f^{(0,m)}$ and correlate it with all entries in the $f^{(0,m)}$ array, which requires m operations. The next block average has to be updated and correlated once every m time-steps, the third once every m^2 steps, and so on. The required number of operations to calculate the whole correlation function in a simulation of length N is thus

$$O_F = Nm\left(1 + \frac{1}{m} + \frac{1}{m^2} + \cdots + \frac{1}{m^i}\right) < N\frac{m^2}{m-1}. \qquad (13)$$

At level i, related to the array $f^{(i,m)}$, the averaging time λ_i and the minimum lag time between points are the same and equal to m^i time-steps (see Table I, for details). As seen in Eqs. (7) and (8) of Sec. II B, the error would be $\alpha(\alpha+1)/12$ for a power law relaxation and of order of 1/12 in the region around the terminal time for an exponential relaxation, which is undesirably large. This means that, although the correlator proposed by Frenkel is very good in the sense of the number of necessary operations and memory usage, it is not in the sense of the error size. In order to improve this aspect, it would be desirable to have an algorithm where both the averaging time and the lag time between averaged values can be controlled independently.

E. Multiple-tau correlator

In this section we review and analyze the multiple-tau time correlator that was introduced in the framework of dynamic light scattering.[4,5] In practical terms, it involves a small modification with respect to the algorithm of Frenkel that allows an independent control of the averaging time and the lag time. The data structure is schematically presented in Fig. 1.

Similar to Frenkel's correlator, we define block averages of the function f according to Eq. (12). However, at level l, the correlator array has now length $p > m$ instead of m (typical values are $m=2$ and $p=16$). Again, the resolution of the stored function decreases with time. The required storage at level l is p per correlation function. For a more detailed explanation of the data structure of the multiple-tau correlator, we refer the reader to Appendix A. The minimum and maximum lag times at level l are pm^{l-1} and $(p-1)m^l$, respectively, in number of time-steps (see Table I). The total required storage for a simulation of length $T=(p-1)m^l\Delta t$ is lp, whereas in the conventional approach it would be $(p-1)m^l$ (see Sec. II A).

At each time-step, we update level 0 and correlate it with all p entries in the $f^{(0,m)}$ array, a procedure that needs p operations. Level 1 has to be updated and correlated every m steps, but not all the correlations need to be calculated. The first relevant lag time at level 1 is at position p/m in the array $f^{(1,m)}$, all other distances smaller than p/m are already calculated in the previous correlator level. Therefore, the minimum lag time that needs to be correlated at level 1 is $p\Delta t$, and the number of correlations that need to be calculated is $p(1-1/m)$. The same argument applies to all levels $l > 1$. The required number of operations is thus

$$O_M = Np + Np\left(1 - \frac{1}{m}\right)\left(\frac{1}{m} + \frac{1}{m^2} + \cdots + \frac{1}{m^l}\right) < N\frac{p(m+1)}{m}, \qquad (14)$$

which is about p/m times larger than Eq. (13) and also linear in N. About Np of these operations are employed in the calculation of the first p lag times [a fraction $m/(m+1)$ of the total number of operations]. If the accuracy of early time correlations is not an issue, one can simplify or even eliminate the calculation of the correlations at level 0 and gain some computational resources. However, the main advantage

TABLE I. Summary of relevant features that describe Frenkel's and multiple-tau correlators: number of correlation points per level N_i, minimum and maximum lag time, $\tau_{\min,i}$ and $\tau_{\max,i}$, and averaging time λ_i, in number of time-steps, at each block average level i.

	Frenkel (m)				Multiple-tau (p,m)			
Level i	N_i	$\tau_{\min,i}$	$\tau_{\max,i}$	λ_i	NP_i	$\tau_{\min,i}$	$\tau_{\max,i}$	λ_i
0	m	0	$(m-1)$	1	p	0	$(p-1)$	1
1	$m-1$	m	$(m-1)m$	m	$p(1-\frac{1}{m})$	p	$(p-1)m$	m
2	$m-1$	m^2	$(m-1)m^2$	m^2	$p(1-\frac{1}{m})$	pm	$(p-1)m^2$	m^2
$\vdots$							$\vdots$	
l	$m-1$	m^l	$(m-1)m^l$	m^l	$p(1-\frac{1}{m})$	pm^{l-1}	$(p-1)m^l$	m^l

of using the multiple-tau correlator does not lie only in the use of computational resources but in the combined effects of adjustable accuracy of the time correlation function and the optimal use of memory. As will be shown in Appendix A, the storage needs for the accurate calculation of the time correlation function of some variable in a realistic molecular dynamics simulation using the multimple-tau correlator may be of just 2000 numbers. This fact permits the calculation of the time correlation functions of many different quantities during a simulation, including per molecule and even per atom quantities, an operation that would be very costly using standard methods.

The main difference between the multiple-tau time correlator and Frenkel's correlator lies in the ratio between the averaging time and the minimum lag time at each level of the correlator, which controls the size of the error in standard cases, as seen in Sec. II B. In Frenkel's correlator this ratio is equal to 1, whereas in the multiple-tau correlator is equal to m/p. This means that we can tune the relative error introduced by the multiple-tau time correlator by carefully selecting m and p. The smaller the ratio m/p, the smaller the relative error will be.

F. Nonaveraged algorithm

For some observables, such as the mean-square displacement, the systematic error due to averaging can become very large, which can be resolved using corrections. However, in general, the derivation of these corrections can become quite tedious (e.g., for higher moments of the displacement). Moreover, they also do not correct the smoothing effect [i.e., the first sum of Eq. (11)]. Another way of resolving this issue is not to submit *average* values to the correlator.

In practice this is implemented as follows. Normally, every value ω is added to the accumulator A and the value A/m is pushed to the next correlator level (as discussed in Appendix A). Instead we only add one out of m values to the accumulator A, although the counter C is still increased for each element ω. Once $C=m$, the value of the accumulator is send to the next level. This corresponds to giving 100% weight to the first element instead of $1/m$ weight to each of the m elements.

As there is no smoothing involved, there are no systematic errors. The obvious drawback of this method is that for every next correlator level only a fraction $1/m$ of the data is used, the rest of the data is discarded. This will lead to an increase in the statistical error. Nevertheless, as we will see, in some situations this increase in the statistical error is almost negligible with the added benefit of not having to resort to correction methods.

III. CASE STUDIES

In this section, we present two real examples of the use of the multiple-tau correlator, both to illustrate its use and to analyze the error. First, we apply it to the calculation of the stress relaxation modulus and the orientation tensor relaxation from MD simulations of chain molecules. Then, we analyze the use of the multiple-tau correlator in Brownian

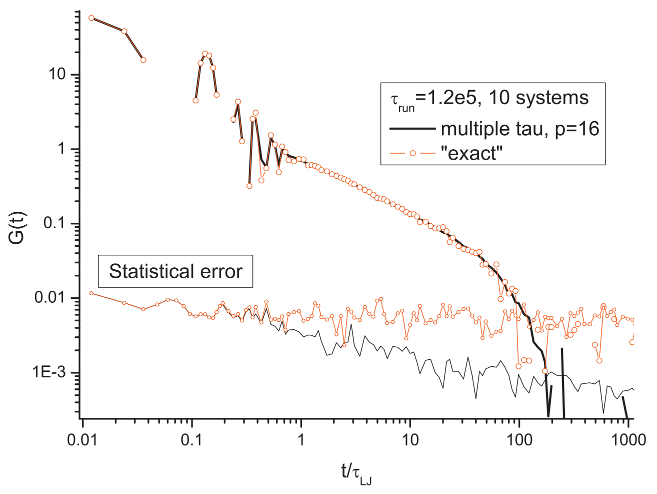

FIG. 2. Stress autocorrelation function averaged over ten runs from the Kremer–Grest model, $N=10$, calculating the exact time correlation function (see the text) and using the multiple-tau correlator with $m=2$ and $p=16$. The bottom curves show the statistical error (the absolute error-bars) for the two methods. The gaps at early time are due to the use of a logarithmic scale and the negative value of G(t).

dynamics (BD) simulations of the well known Rouse model,[13] for which analytical expressions of correlations can be calculated.

A. Stress from molecular dynamics

The stress autocorrelation function $G(t)$ is notoriously difficult to calculate in molecular dynamics due to huge fluctuations at early time (caused by bond vibrations). In order to analyze different possibilities of computing $G(t)$, we run molecular dynamics of the standard Kremer–Grest model described elsewhere.[10,14] We use a particle density $\rho=0.85$, chain length $N=10$, and a cubic box of 30 chains. To improve accuracy, we average $\langle\sigma_{xy}(t)\sigma_{xy}(0)\rangle$ over all possible ways to select a pair of perpendicular axis x and y. The result is

$$G(t) = \frac{V}{5k_BT}[\langle\sigma_{xy}(t)\sigma_{xy}(0)\rangle + \langle\sigma_{yz}(t)\sigma_{yz}(0)\rangle + \langle\sigma_{zx}(t)\sigma_{zx}(0)\rangle] + \frac{V}{30k_BT}[\langle N_{xy}(t)N_{xy}(0)\rangle + \langle N_{xz}(t)N_{xz}(0)\rangle + \langle N_{yz}(t)N_{yz}(0)\rangle], \quad (15)$$

where $N_{\alpha\beta}=\sigma_{\alpha\alpha}-\sigma_{\beta\beta}$.

There are two sources of errors in the calculated $G(t)$: statistical errors due to a finite simulation time and systematic errors introduced by the use of averaged correlators. In order to separate them, we perform 10 (Fig. 2) and 100 simulations (Figs. 3–6) of length $T=1.2\times10^5\tau_{LJ}$, which corresponds to 2400 terminal relaxation times μ (which is $50\tau_{LJ}$ for our system, τ_{LJ} being the Lennard-Jones unit of time). In each run we calculate the "exact" correlations according to Eq. (2) and the correlation using the suggested correlators with different values of p.

Figure 2 shows $G(t)$ averaged over $N_{ex}=10$ runs, obtained with the two methods. It is apparent visually that, apart from saving memory, the use of the multiple-tau cor-

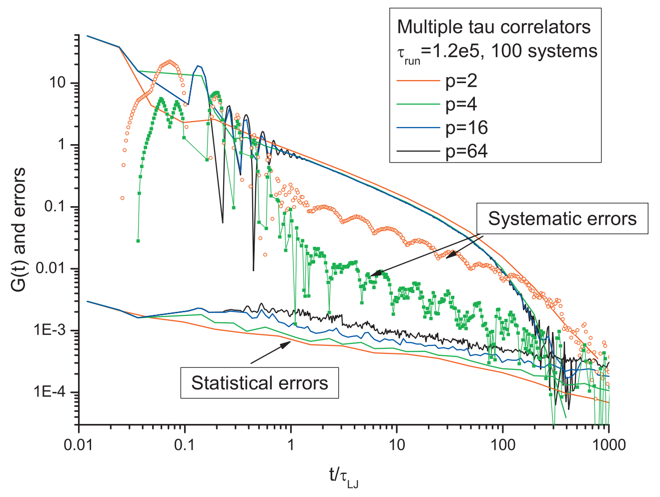

FIG. 3. Stress autocorrelation function averaged over 100 runs obtained using multiple-tau correlators with $p=2,4,16,64$, statistical error (lines at the bottom) and systematic error for $p=2$ and 4. Systematic error for higher p is below statistical error (not shown).

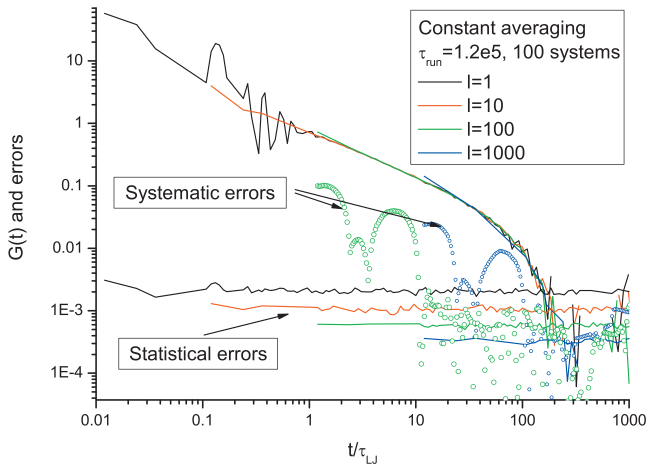

FIG. 4. Stress autocorrelation function averaged over 100 runs obtained with fixed preaveraging over $l=1,10,100,1000$ points. Statistical errors are shown by horizontal curves, whereas systematic errors for 1=100 and 1 =1000 are shown by symbols.

relator seems to yield smoother data in the terminal region. In order to quantify this, we perform statistical analysis and define the statistical error as the mean deviation from the average, divided by the square root of the number of independent experiments N_{ex},

$$\Delta_{\text{stat}}(t) = \frac{1}{\sqrt{N_{\text{ex}}}}\sqrt{\frac{1}{N_{\text{ex}}}\sum_{k=1}^{N_{\text{ex}}} G_k^2(t) - \left(\frac{1}{N_{\text{ex}}}\sum_{k=1}^{N_{\text{ex}}} G_k(t)\right)^2}. \quad (16)$$

Figure 2 shows that statistical errors from the multiple-tau correlator data are smaller than those from Eq. (2). To illustrate this in more detail, we plot results for $G(t)$ and its statistical error $\Delta_{\text{stat}}(t)$ obtained by using correlators with different p and $m=2$ for $N_{\text{ex}}=100$ simulations in Fig. 3. The point where $G(t)$ intersects with $\Delta(t)$ corresponds to 100% error. The systematic error is estimated by the difference of $G(t)$ between small p and the largest $p=64$, for which the systematic error is negligible. These systematic errors are plotted by lines with symbols for $p=2$ and $p=4$. We see that for the low p, the systematic error is significant, whereas for $p=8$ and above, it falls below the statistical error for $t>\tau_{\text{LJ}}$. From Eq. (6) we expect it to decrease as $1/p^2$, which is consistent with the data. We note that usually in realistic simulations statistical errors are larger than in our case since we considered 100 very long runs in terms of the terminal time. Thus, any $p\geq 8$ can be safely used. A useful finding is that the statistical error is increasing with increasing p, i.e., smoothing actually decreases the statistical error by about a factor of 4, which is equivalent to running 16 times more experiments.

An alternative method often used in simulations is to save the stress tensor averaged over a fixed number of time-steps into a file, and then postprocess these files using the exact correlation definition, Eq. (2). This corresponds to fixed k in Eq. (4) as opposed to choosing l proportional to time as in the multiple-tau correlator. The analysis of the same data using this method is shown in Fig. 4. The case $l=1$ corresponds to no averaging, i.e., using exact Eq. (2). We see that this produces a statistical error of $\Delta(t)\approx 2\times 10^{-3}$, which is independent of time and crosses $G(t)$ at $t=150\tau_{LJ}$, i.e., at about three relaxation times. Therefore, even after running 100 experiments for 2400 longest relaxation times μ each exact correlation function is still too noisy to produce an accurate estimate of the terminal region.

Note that the reason for the statistical error appearing to be independent on the lag time t on a logarithmic scale is that the number of independent samples is about $(T-t)/\mu$ rendering a relative error of approximately $\sqrt{\mu/(T-t)}$, which is in zeroth order independent of t.

Averaging over 100 time-steps reduces the statistical error to $\Delta=0.6\times 10^{-3}$, which is still worse than $\Delta_{\text{stat}}(100)\approx 0.3\times 10^{-3}$ obtained by using the multiple-tau correlator with $p=16$ for times around the terminal time. Using a larger fixed averaging time l is not practical since the range of the accurately obtained function will be very narrow.

In order to check that our results are not limited to the stress relaxation function, we use the same simulations to obtain the orientation tensor autocorrelation function. We define the orientation tensor as

$$O_{\alpha\beta}(t) = \frac{1}{N_{\text{bonds}}}\sum_{\text{all bonds}} u_\alpha u_\beta,$$

where u is a bond vector, and the sum is performed over all bonds in the system. The autocorrelation function $S(t)$ is then defined similar to Eq. (15). Figures 5 and 6 show the same analysis as Figs. 3 and 4, but for the orientation autocorrelation function. One can see that averaging here has less effect on the statistical accuracy (i.e., averaging over 1000 points reduces the error by a factor of 1.6 instead of factor of 5 in the $G(t)$ calculations). This is explained by the absence of large oscillations in $S(t)$ at early time as compared to $G(t)$. However, the relative statistical error without preaveraging is smaller, and thus the systematic error plays a slightly bigger role: the $p=8$ correlator still has noticeable error of about 1%, but use of $p=16$ is safely below the statistical noise apart from the little region around $t\approx 0.3\tau_{\text{LJ}}$, where the second derivative is large. Even in this region it is below 0.5%. In view of the previous results, we can safely conclude that

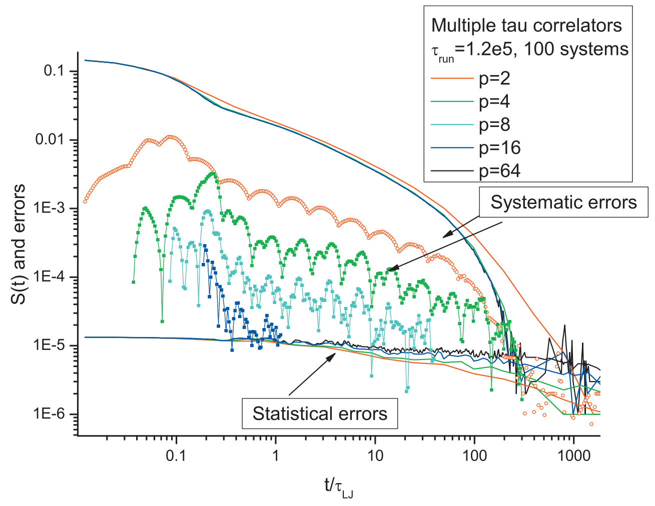

FIG. 5. Similar to Fig. 3 but for the orientation tensor autocorrelation function.

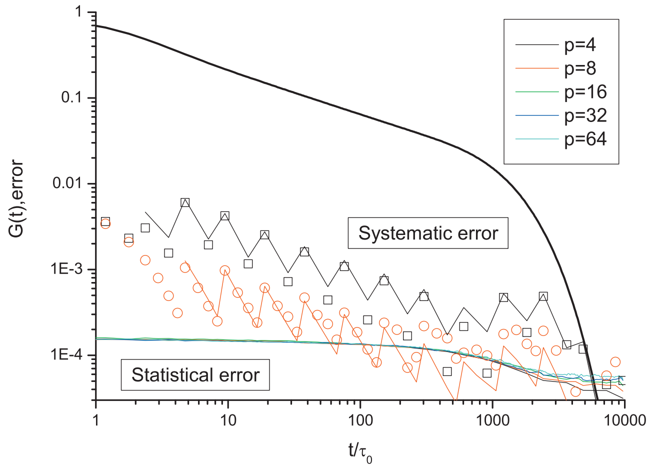

FIG. 7. Relaxation modulus, calculated from the stress-stress autocorrelation function of Rouse chains of length $N=20$ with the chain ends fixed at the origin. The bold line shows the analytical solution obtained from normal mode analysis; the thin lines at the bottom show the error bars of the multiple-tau correlators with $m=2$ and $p=4,8,16,32,64$ calculated from Brownian dynamics simulations; the symbols show the systematic error from the simulations, in good agreement with the predictions of Eq. (5) shown in lines.

the best choice is $p=16$. A larger p value is worse from the statistical point of view, whereas smaller values result in systematic error.

B. Brownian dynamics simulations of Rouse chains

Brownian dynamics simulations of single chain models are frequently used for comparison with experimental data or to test new physical ideas. Apart from being much simpler to calculate and yielding smoother results than molecular dynamics simulations, they are very useful to test numerical methods because they can be compared in simple cases against known analytical results. One of our goals in this paper is to test the accuracy of the error estimates produced in Sec. II. In particular, we intend to check the error corrections derived for the standard time correlation function [Eq. (5)] and the mean-square displacement [Eq. (11)]. Moreover, we want to see what the influence on the statistical error is when the nonsmoothing algorithm, as described in Sec. II F, is applied.

For these purposes, we run Brownian dynamics simulations of a linear Rouse chain of $N=20$ bonds, with the position of the first bead fixed at the origin. As it is well known, the relaxation dynamics of the Rouse model can be determined completely by recurring to normal mode analysis. In particular, the expressions for the relaxation modulus or the mean-square displacement of the center of mass or of any given monomer can be easily obtained. In contrast to the stress or orientation relaxation functions, averaged mean-square displacements are monotonically increasing functions. By fixing the position of one of the chain ends, mean-square displacements become asymptotically bounded at late times. We have identified this as a critical case for the use of the multiple-tau correlator, so this particular version of the Rouse model has been selected for the present section.

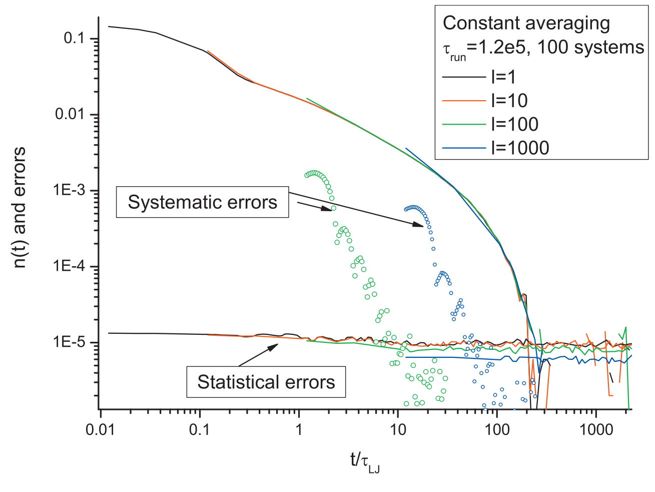

FIG. 6. Similar to Fig. 4 but for the orientation tensor autocorrelation function.

Let us focus first on the stress relaxation. In Fig. 7, the analytical calculation of the relaxation modulus of a Rouse chain with an end fixed is shown as a bold black line. It clearly shows the expected Rouse features: $-1/2$ power law regime ended by a terminal exponential relaxation at the terminal time τ_R, which is proportional to the square of the chain length. For simplicity, in this and the following plots, we use units where $k_BT=1$, the friction coefficient $\zeta=1$, and the averaged squared bond length $b^2=1$; in these units, the longest relaxation or Rouse time is $\tau_R \approx 4N^2\tau_0$, with $\tau_0 = 1/3\pi^2$. As we did in Sec. II for molecular dynamics simulation, we are going to analyze the two contributions to the relaxation modulus calculated with the multiple-tau correlator: the statistical error due to finite length of the statistical ensemble and other computational aspects of the simulation (integration algorithm and time-step size); and the systematic error coming from the use of the multiple-tau correlator. In Fig. 7, we present the results from a simulation of 1000 chains run for 1000 terminal times, where $m=2$ is fixed and p is changed over a wide range. The statistical errors, calculated using Eq. (16), are shown as lines at the bottom. The systematic error is calculated as the difference between the numerical value of $G(t)$ obtained from the

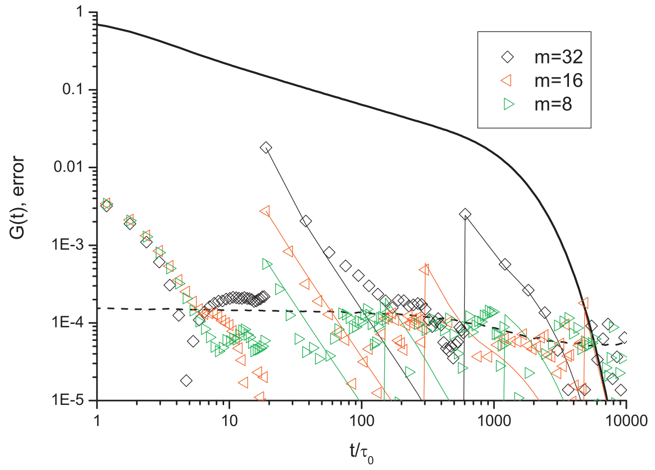

FIG. 8. Same as Fig. 7 with $p=32$ and $m=8,16,32$. The error bars (dashed line) are the same for all correlators. Again, symbols and lines show the systematic error from the simulations and the predictions of Eq. (5), respectively.

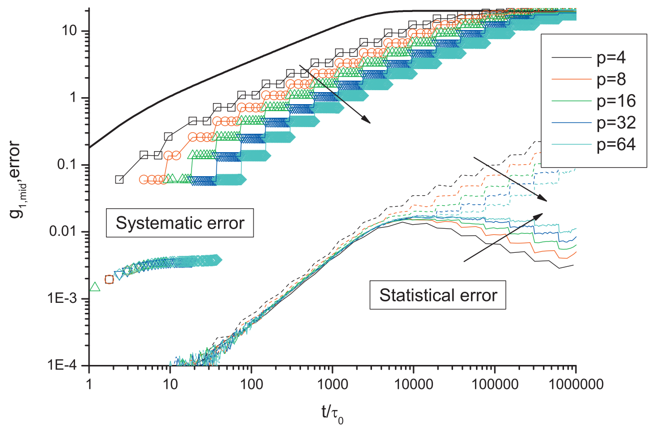

FIG. 9. Mean-squared displacement of the middle monomer ($i=10$) of the Rouse simulations shown in previous figures. The bold line shows the analytical solution; the thin lines at the bottom show the error bars of the multiple-tau correlators with $m=2$ and $p=4,8,16,32,64$; the symbols show the systematic error from the simulations, in good agreement with the predictions of Eq. (11) shown with lines. The dashed lines at the bottom show the statistical error when no averaging is applied (in which case the systematic error equals zero). Arrows point toward increasing p.

multiple-tau correlator for each p (not shown on the plot) and the analytical value (bold line), and is shown as symbols, along with the predictions of Eq. (5), shown as lines. The agreement between predicted and calculated systematic errors is excellent, except for the first few data points. For these initial points, which correspond to the first level in the multiple-tau correlator, Eq. (5) predicts no error. Obviously, the simulation yields errors of computational origin also for those points. As seen in molecular dynamics, the systematic error is significant for low values of p but gets below the level of the statistical error for $p>8$. It must be noted that our simulations are rather precise. In practice, due to limited computational availability, Brownian dynamics simulations are calculated on smaller ensembles and over a shorter lengths. We once again see that the statistical error is smaller for lower values of p, but the effect is clearly less important than in molecular dynamics.

For the sake of completeness, we repeat the same analysis for a fixed value of $p=32$ and changing values of m. The results are shown in Fig. 8. In this case, the statistical errors are almost the same for all values of m investigated, and therefore they are shown as a single dashed line. The systematic error is significant [from Eq. (6), we expect it to grow as m^2], but falls below the level of the statistical error for $m<8$. In fact, as can be easily extracted from Table I, for a given lag time t, the averaging time in the multiple-tau correlator is bounded from above by mt/p. In the power law relaxation region of $G(t)$, according to Eq. (7) where $\alpha=1/2$, we expect the relative systematic error to be bounded by $(m/4p)^2$. We can use this estimate to chose the proper ratio p/m so that the error is always below the desired level. For example, if $p/m>8$, the relative systematic error will be smaller than 0.1% in the power law region. A similar type of analysis can be done for the exponential terminal relaxation region. As a general conclusion, a choice of $p/m\geq 8$ seems most appropriate. It is clear that a very large value of m may compromise the resolution of the results. From a balance between storage needs and precision, we conclude that the preferred choice of parameters is $p=16$ and $m=2$.

We turn our attention now to the mean-squared displacement (MSD) of the middle monomer, or $g_{1,\text{mid}}$, of our end-attached Rouse chain of length $N=20$. In Fig. 9, the analytical solution of $g_{1,\text{mid}}$ is shown as a bold black line. As expected, it shows a clear 1/2 slope roughly from τ_0 to τ_R followed by a flat region at later times whose value depends simply on the size of the chain. The calculations with the multiple-tau correlator from the simulations have the usual two errors. The statistical errors, shown as thin colored lines at the bottom of Fig. 9, have the same trend as before: for fixed m, the size of the error increases with p. However, the size of the statistical error in this case is more than three orders of magnitude below the value of $g_{1,\text{mid}}$. This observation will be used later on. The main contribution to the error clearly comes from the systematic error, shown as symbols in Fig. 9. The agreement with the prediction of Eq. (11), shown as lines, is again excellent. The systematic error has a clear steplike shape in which the different levels of the multiple-tau correlator can be easily differentiated. It can be seen that the size of the systematic error grows monotonically, eventually getting equal to $g_{1,\text{mid}}$ in the terminal region, which is unacceptably large. It is therefore important to eliminate or reduce as much as possible the systematic error from the results of the multiple-tau correlator.

We will employ two methods to achieve this. One method will be to completely eliminate the systematic error in the MSD, but it sacrifices some statistical accuracy, as discussed in Sec. II F. The other method will employ a correction, which lead to a considerable decrease in the systematic error, without affecting the statistical accuracy.

In Fig. 9 the statistical error for the nonsmoothing version of the algorithm is shown as dashed lines. We see that before the saturation of the MSD, the increase in the statistical error is nearly negligible as compared to the smoothened version in which all simulation data are used. The reason for this seems to be connected to the fact that the fluctuations in the MSD increase with lag time, so that ne-

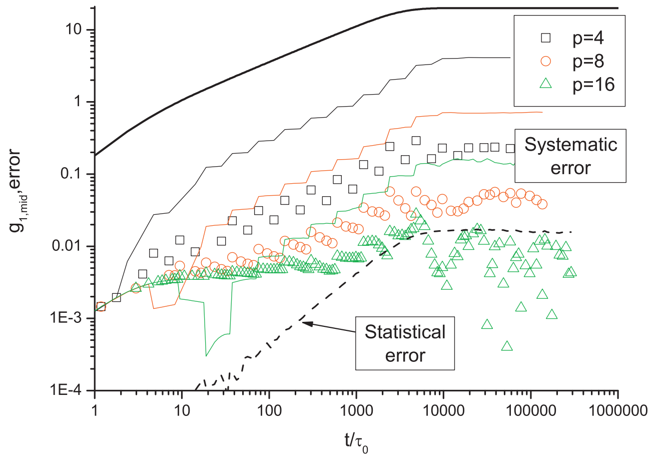

FIG. 10. Same as Fig. 9 but adding the second term of the correction of Eq. (11). The error bars (dashed line) are the same for all correlators. The systematic errors for $p=4,8,16$ after correction with Eq. (17) are shown with symbols; the lines show the systematic errors using a simpler correction detailed in the text.

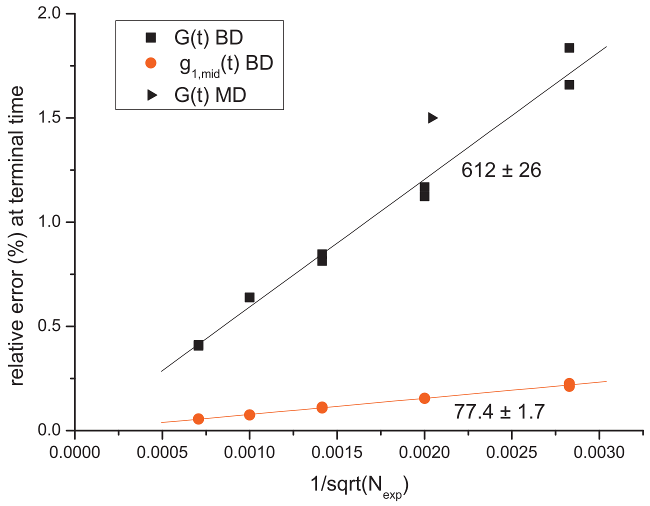

FIG. 11. Error bar of the relaxation modulus $G(t)$ (black) and the mean-squared displacement of the middle monomer $g_{1,\text{mid}}$, as percentage of the value of the function at the terminal time, plotted as a function of the number of experiments $N_{\text{exp}}=N_{\text{ch}}T/\tau_R$ for BD and $N_{\text{exp}}=N_{\text{sim}}T/\mu$ for MD. The numbers show the slope of the least-squares linear fitting to the BD data.

glecting some of the high-frequency data with small fluctuations does not really affect the statistical error. Only in the flat regime of the MSD, the statistical error for the nonaveraged case starts to deviate significantly from the averaged one, with the deviation starting later for larger p. As the statistical error is so small, one can afford an increase—at $t=10^6\tau_0$, far beyond the terminal time, the error in the MSD is still two orders of magnitude smaller than the MSD itself for $p=16$.

Let us now inspect the second method, correcting the systematic error. The systematic error of Eq. (11) has two terms. The second one is equivalent to the mean-squared displacement at times earlier than the averaging time. As the averaging time grows monotonically with the lag time, this second term becomes dominant at late times. Since we know the exact averaging time k at each correlator level, we can correct for the systematic error using the following expression, taken from Eq. (11):

$$e_k \approx -\frac{1}{k^2}\sum_{p=-k+1}^{k-1}(k-|p|)g_{1,\text{mid},p} = -\frac{2}{k^2}\sum_{p=1}^{k-1}(k-p)g_{1,\text{mid},p}, \tag{17}$$

where time reversibility and the fact that $g_{1,\text{mid},0}=0$ have been used. This correction is constant at each level of the multiple-tau correlator, which is consistent with the steplike shape of the systematic errors in Fig. 9. In the last equation, the values of $g_{1,\text{mid},p}$ can be extracted from the multiple-tau correlator itself. Since the values in the correlator are stored with decreasing resolution, it is necessary to interpolate between different values of $g_{1,\text{mid},p}$ in order to calculate the sum in Eq. (17). The results of the multiple-tau correlator corrected with the expression above are shown in Fig. 10 by symbols. When compared with the previous figure, it can be clearly seen that the correction reduces the size of the systematic error, shown as symbols, by several orders of magnitude. The remaining systematic error has the same origin as in the stress correlation functions, i.e., due to smoothing [first sum in Eq. (11)]. The systematic error even falls below the level of the statistical error for our preferred set of correlator parameters, $p=16$ and $m=2$. Note also that, by correcting the systematic error of $g_{1,\text{mid}}$, we have also eliminated the effect of p on the statistical error. This is a sign that the systematic error has an important contribution to the statistics of the results.

If the correction in Eq. (17) is considered to be too complicated, a simpler alternative for the correction is possible. The systematic error can be approximated by shifting the values of $g_{1,\text{mid}}$ at each correlator level by a constant, such that the first point at each level is aligned with the last two points in the previous level. This simpler correction, although of a pure phenomenological origin, yields acceptable results (a reduction of the error of almost two orders of magnitude for $p=16$ and $m=2$), as can be seen in Fig. 10, where they are represented as thin lines.

Upon comparing the nonsmoothing algorithm and the algorithm in which a correction is applied, we see that there is not a large difference in the final accuracy; hence both are viable methods. However, the nonsmoothing algorithm can be applied to *any* observable without having to derive an analytical correction.

We conclude this section by showing the statistical errors of our simulations in the terminal time region for the relaxation modulus and the mean-squared displacement of the middle monomer. These are shown in Fig. 11 for G(t) (black line and symbols) and $g_{1,\text{mid}}$ (red line and symbols) as a function of the total number of "experiments" N_{exp}. For the BD simulations we use the product of the ensemble size (number of independent chains N_{ch}) times the length of the simulation (in number of terminal times T/τ_R), $N_{\text{exp}}=N_{\text{ch}}T/\tau_R$. We note that the BD simulations were run using a predictor-corrector integration method with a time-step $\Delta t=0.02$ (in the units of the simulation). As expected, the error depends linearly on $N_{\text{exp}}^{-1/2}$. As a simple rule, we can conclude that the error in percent of $G(t)$ in the terminal region for a

BD simulation of bead-spring chains with 21 beads is approximately equal to $600/\sqrt{N_{\mathrm{exp}}}$, i.e., one needs about 360 000 experiments to get the value of the correlation function exact within 1% near the terminal time. Also shown is the result for the accuracy in $G(t)$ at the terminal time μ from the MD simulations. As $G(t)$ is affected by cross-correlations between chains, we took the number of independent simulations N_{sim} instead of N_{ch} for calculating the number of independent experiments N_{exp}.

We recommend to select carefully the size of the simulation, N_{exp}, so that the size of the statistical error is known, and then choose the appropriate set of parameters for the multiple-tau correlator (the ones that we recommend are $p=16$ and $m=2$). This way, the results will have the desired accuracy with the least memory and computational effort.

IV. CONCLUSION

In this paper, we have reviewed and analyzed the multiple-tau correlator, a method for the calculation of time correlation functions on the fly during computer simulations, which is optimal in the sense of memory and computational requirements, and can be tuned to get the desired level of accuracy. The method is well known in the area of experimental light scattering but, to the best of our knowledge, is not so well known to researchers doing computer simulations, where time correlation functions are frequently calculated. We have derived some expressions which allow error estimation due to the multiple-tau correlator and extended these definitions for the calculation of mean-square displacements, also common in simulations. In Appendix B we also developed a correlator for the calculation of the dynamic structure factor in isotropic systems.

The multiple-tau correlator is based on the idea of preaveraging the data, with changing averaging time, before the calculation of time correlations. It has two parameters: m, which controls the number of values averaged every time, and p, which sets the amount of stored data at each resolution level. The combination sets the resolution of the results. Theoretical considerations show that the ratio p/m controls the minimum lag time between points at each level of the multiple-tau correlator, which in turns controls the size of the relative error. A careful analysis of the results of the multiple-tau correlator applied to molecular dynamics and Brownian dynamics simulations indicates that the best choice of correlator parameters is $p=16$ and $m=2$. This choice ensures that the systematic error is always below the level of the statistical error of the simulation (mostly determined by the ensemble size and simulation length). For the particular case of the mean-squared displacements, we have shown that two methods can be employed. One either corrects the results according to systematic error estimators provided. If this is done correctly, the multiple-tau correlator can be used very advantageously to calculate diffusion data and its error can be kept below the statistical error. The other way is to resort to the nonsmoothing version of the multiple-tau algorithm, which renders the systematic error void and still leads to an acceptable level of the statistical error. This version can be easily applied to other observables such as higher moments of displacements.

We are hoping that this work will attract some interest in the simulation community, specially in the area of computer simulation of dynamical properties of complex systems. The efficient method reviewed in this paper opens a lot of possibilities for the analysis of data from simulations and can help researchers to extract more useful information from their own work. For a more detailed explanation of the algorithm used for the multiple-tau correlator, we point the reader to Appendix A. Samples of the use of this algorithm in simple codes can be obtained by contacting the authors or directly from http://www.personal.reading.ac.uk/~sms06al2/.

ACKNOWLEDGMENTS

We gratefully acknowledge financial support from EPSRC, through the μPP^2 project (J.R. and B.V.), Grant No.GR/S94711/01 (S.K.S.), Grant No. EP/H016686/1 (B.V.), and an Advanced Research Fellowship (A.E.L.), and B. Loppinet for drawing our attention to the multiple-tau correlators.

APPENDIX A: DETAILED DATA STRUCTURE OF THE MULTIPLE-TAU CORRELATOR

The multiple-tau correlator with parameters m and p is composed of a series of $S+1$ levels. Each level i contains three arrays of size p, D_{ij} to store the data, C_{ij} to store the correlation results and N_{ij}, a counter used to calculate averages; in addition, each level contains an accumulator A_i and a counter M_i (the first index $i=0\ldots S$ always refers to the level, and the second index $j=0\ldots p-1$ to the elements of arrays inside each level). All the arrays are initialized to zero at the start of the simulation.

When a new data value ω is sent to correlator level i, the following sequence of operations takes place.

(1) ω is stored at the position 0 of the data array, pushing already stored values to the right (the performance of this operation can be improved by using data pointers),

$$\mathrm{D}_{ij}=\mathrm{D}_{ij-1}, \quad j=1\ldots p-1, \quad \mathrm{D}_{i0}=\omega.$$

(2) The correlation array and correlation counter are updated,

$$\mathrm{C}_{ij}=\mathrm{C}_{ij}+\mathrm{D}_{i0}\mathrm{D}_{ij}, \quad \mathrm{N}_{ij}=\mathrm{N}_{ij}+1,$$

where the calculation runs from $j=0\ldots p-1$ at level 0 and from $j=p/m\ldots p-1$ otherwise (lag distances smaller than p/m are already calculated at previous levels).

(3) ω is added to the accumulator and the counter is incremented,

$$\mathrm{A}_i=\mathrm{A}_i+\omega, \quad \mathrm{M}_i=\mathrm{M}_i+1.$$

In the nonsmoothened version ω is only added if $M_i=0$.

(4) If $\mathrm{M}_i=m$, the averaged accumulator A_i/m is sent to the next level, $i+1$, and both A_i and M_i are reset to zero. In

the nonsmoothened version, the only difference is that A_i instead of A_i/m is send to the next level.

Although it is not absolutely necessary, for the sake of simplicity, it is recommended that p is a multiple of m. The actual step where the correlation is calculated (step 2) can be adapted to the calculation of different time correlation functions, for example, mean-square displacements, vector-vector correlations, cross-correlations, or dynamic structure factors.

Data values coming from the simulation (values of the stress, pressure, velocity, etc.) are sent to correlator level 0 every Δt (which may be equal to the simulation time-step or a multiple of it). Averaged values from level i are sent to level $i+1$ every m steps. Therefore, the lag time between values stored in the array D_{ij} is $m^i\Delta t$. A schematic view of the hierarchical structure of the data array D_{ij} is depicted on the right hand side of Fig. 1. Some other characteristics of the multiple-tau correlator, such as the number of points, the averaging time, and the minimum and maximum lag times per level, can be found in Table I (along with the equivalent values for the correlator proposed by Frenkel). When the counter on the last correlator level $M_S=m$, the values of this level counter and accumulator are simply reset to zero, and no value is sent to another level.

At any time during the simulation, the resulting time correlation function $f_k=f(t_k)$ can be easily recovered by running a loop over the different levels of the multiple-tau correlator and averaging the values of correlation array: $t_k=jm^i\Delta t$, $f_k=C_{ij}/N_{ij}$, where $i=0\ldots S$ and, again, $j=0\ldots p-1$ on level 0 and $j=p/m\ldots p-1$ otherwise.

The total storage requirements for the multiple-tau correlator are $3(S+1)p$ numbers for the arrays D_{ij}, C_{ij}, and N_{ij}, plus $2(S+1)$ numbers for the arrays A_i and M_i. The maximum value of the lag time available in the correlator is $(p-1)m^S\Delta t$ or $(p-1)m^S$ time-steps. Our preferred set of values is $p=16$, $m=2$, and $S=40$ (these values might be slightly different depending on the problem at hand). With these values we can reach 1.65×10^{13} time-steps with a reasonable storage of just over 2000 numbers. The maximum number of operations that can take place in a single time-step is bounded by $2S(p+1)$ (and this happens only when a time-step is multiple of m^S, something that, with the previous values of m and S, only happens every 10^{12} steps). This number is reasonably small compared to the number of operations that are usually needed in an atomistic simulation in order to calculate the forces, and therefore allow the calculation of the time correlation function on the fly during the computer simulation. A simple C++ code for the multiple-tau correlator, with examples and wrappers for C and FORTRAN, can be obtained from http://www.personal.reading.ac.uk/~sms06al2/.

APPENDIX B: SINGLE CHAIN DYNAMIC STRUCTURE FACTOR

The dynamics of individual molecules can be studied in simulations by means of the dynamic structure factor. This function is also interesting because it can be compared with experimental data from dynamic light scattering or neutron spin-echo and in some cases, it can be related to theoretical concepts such as the tube diameter from polymer dynamics. The expression for the coherent dynamic structure factor for a single chain with $N'=N+1$ scattering beads is[15]

$$S(\mathbf{q},t)=\frac{1}{N+1}\sum_{n,m=0}^{N}\langle\exp[i\mathbf{q}\cdot(\mathbf{R}_n(t)-\mathbf{R}_m(0))]\rangle, \tag{B1}$$

where the brackets $\langle\,\rangle$ indicate an average over equilibrium conditions. If the system under study is isotropic, one can average over all possible orientations of the scattering vector $\mathbf{q}$. It is common in practice to average only over three different orthogonal vectors $\mathbf{q}$'s. In the latter case, it is possible to express Eq. (B1) in a form that is suitable for the multiple-tau correlator. Averaging over three orthogonal vectors along the axes of coordinates, $q\mathbf{i}$, $q\mathbf{j}$, and $q\mathbf{k}$, and taking the real part only, Eq. (B1) becomes

$$S(q,t)=\frac{1}{3(N+1)}\sum_{n,m=0}^{N}\sum_{\alpha=x,y,z}\cos[q(R_{n\alpha}(t)-R_{m\alpha}(0))].$$

After expanding the cosine of the difference and splitting the sums, we get

$$S(q,t)=\frac{1}{3(N+1)}\sum_{\alpha=x,y,z}\left\{\sum_{n=0}^{N}\cos[qR_{n\alpha}(t)]\times\sum_{m=0}^{N}\cos[qR_{m\alpha}(0)]+\sum_{n=0}^{N}\sin[qR_{n\alpha}(t)]\sum_{m=0}^{N}\sin[qR_{m\alpha}(0)]\right\}.$$

So, if we define the following time dependent functions:

$$C_\alpha(t)=\sum_{n=0}^{N}\cos[qR_{n\alpha}(t)],$$

$$S_\alpha(t)=\sum_{n=0}^{N}\sin[qR_{n\alpha}(t)],$$

the calculation of the chain dynamic structure factor will be simply

$$S(q,t)=\frac{1}{3(N+1)}\sum_{\alpha=x,y,z}[C_\alpha(t)C_\alpha(0)+S_\alpha(t)S_\alpha(0)]. \tag{B2}$$

This function can be easily calculated by using six different correlators for each molecule. We just need to add the quantities C_α and S_α to each correlator at each time-step and sum the correlation functions at the end.

[1] D. Chandler, *Introduction to Modern Statistical Mechanics* (Oxford University Press, Oxford, 1987).
[2] M. P. Allen and D. J. Tildesley, *Computer Simulation of Liquids* (Oxford Science, New York, 1987).
[3] D. Frenkel and B. Smit, *Understanding Molecular Simulation*, 2nd ed. (Academic, London, 2002), p. 90.
[4] B. J. Berne and R. Pecora, *Dynamic Light Scattering* (Wiley, New York, 1976).
[5] D. Magatti and F. Ferri, Appl. Opt. **40**, 4011 (2001).
[6] K. Schatzel, M. Drewel, and S. Stimac, J. Mod. Opt. **35**, 711 (1988).
[7] A. E. Likhtman, Macromolecules **38**, 6128 (2005).

[8] J. Ramirez, S. K. Sukumaran, and A. E. Likhtman, Macromol. Symp. **252**, 119 (2007).
[9] J. Ramirez, S. K. Sukumaran, and A. E. Likhtman, J. Chem. Phys. **126**, 244904 (2007).
[10] A. E. Likhtman, S. K. Sukumaran, and J. Ramirez, Macromolecules **40**, 6748 (2007).
[11] S. K. Sukumaran and A. E. Likhtman, Macromolecules **42**, 4300 (2009).
[12] A. E. Likhtman and S. K. Sukumaran, Macromolecules **43**, 3980 (2010).
[13] P. Rouse, J. Chem. Phys. **21**, 1272 (1953).
[14] K. Kremer and G. S. Grest, J. Chem. Phys. **92**, 5057 (1990).
[15] M. Doi and S. F. Edwards, *The Theory of Polymer Dynamics* (Clarendon, Oxford, 1986).