TOKI Compression for Plasma Particle Simulations*)

Katsumi HAGITA, Hiroaki OHTANI[1)], Tsunehiko KATO[2)] and Seiji ISHIGURO[1)]

National Defense Academy, 1-10-20 Hashirimizu, Yokosuka 239-8686, Japan
[1)]*National Institute for Fusion Science, 322-6 Oroshi-cho, Toki 509-5292, Japan*
[2)]*Hiroshima University, 1-3-1 Kagamiyama, Higashi-Hiroshima 739-8526, Japan*

(Received 19 December 2013 / Accepted 28 April 2014)

We propose a TOKI (Time Order, Kinetic, and Irreversible) compression method for recording smooth trajectories of particles from PIC (electromagnetic particle-in-cell) simulations. In a TOKI compression, instead of storing entire time sequences of particle positions, we store particle trajectories in terms of coefficients of approximating polynomials. In the current implementation, these coefficients are determined either by the least-squares method or by the Chebyshev approximation formula to obtain quasi-minimax polynomials. In this paper, we present the technique of TOKI compression and compare it with other lossy compression schemes, such as XTC. Comparisons are made using data from a PIC simulation for 150,000 electrons and 150,000 ions. For smooth trajectories, the compression ratio by TOKI is better than that by the XTC format. However, for ballistic trajectories, the compression ratio by TOKI is not good because of the significant overhead in storing raw values of trajectories. We also found that the compression efficiency for ion trajectories is better than that for electron trajectories. This is attributed to different characteristic time scales of motions due to the difference in mass. We expect that the behavior of the compression ratio in TOKI can be used to characterize motions of plasma particles.

Keywords: TOKI compression, plasma particle simulation, visualization, smooth animation

DOI: 10.1585/pfr.9.3401083

1. Introduction

Large-scale simulations in fusion science are now being extensively performed thanks to recent progress in computer power. We can perform plasma particle simulations with billions of particles by the PIC (electromagnetic particle-in-cell) method [1]. However, in these large-scale simulations, recording particle data (e.g., trajectories) requires massive storage, which has become a serious problem because improvements in performance of data storage devices have been much slower than improvements in computer power. Hence, observations of large-scale simulation results, such as the detailed behavior of particles, require much more effort than previous simulations. For such situations, we propose an improvement by compressing trajectory data. To achieve a high compression ratio, we approximate the trajectory of a particle by piecewise polynomials and store the coefficients of the polynomials instead of storing the entire time sequence of particle positions. We call this scheme the TOKI (Time Order, Kinetic, and Irreversible) compression method.

The basic concepts of TOKI are as follows. (i) The trajectory of each particle can be approximated by piecewise polynomials, provided the trajectory is divided into appropriate subregions. (ii) In each subregion, the trajectory is approximated by a polynomial, so the behavior of the trajectory is captured by the coefficients of that polynomial. For example, if we use third-degree polynomials for approximation, only four coefficients are needed to represent the trajectory in a subregion. This can be an efficient (lossy) compression method for particle trajectory data. (iii) Subregions are determined so that the error between the original trajectory and the polynomial approximation is below a predetermined value that ensures the accuracy of the approximation.

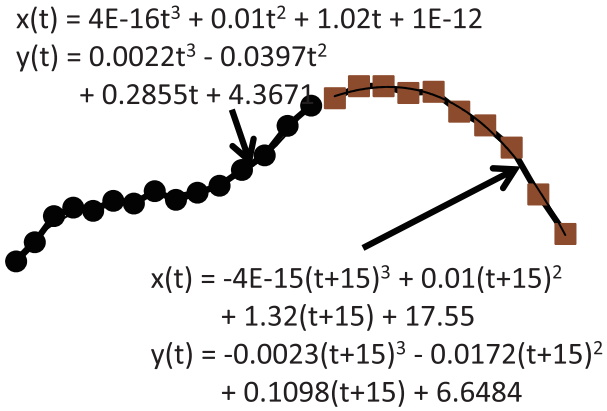

Fig. 1 Illustration of a particle trajectory in two dimensions (points) and curves fit by TOKI compression to third-degree polynomials.

As a brief explanation of the concept of TOKI, Figure 1 shows an illustration of a particle trajectory in two

author's e-mail: hagita@nda.ac.jp
*) This article is based on the presentation at the 23rd International Toki Conference (ITC23).

dimensions and the fitted curves obtained by TOKI compression. In this figure, the trajectory is divided into two intervals and each interval is fit by a third-degree polynomial. Division of the trajectory into intervals is needed to keep the maximum error between the data and the fitted curve (polynomial) below a given tolerance. (In the figure, the first interval corresponds to the time range t = 0 - 14 and the second interval to the range t = 15 - 24.) In the first interval, 15 points are replaced by the values of four coefficients in the third-degree polynomial; this is data compression.

Plasma is a quasi-neutral gas of charged and neutral particles that exhibit collective behavior [2]. In plasma, magnetic reconnection is an interesting phenomenon. During magnetic reconnection, the topology of magnetic field lines changes on a macroscopic scale and global plasma transport takes place. In this process, the frozen-in magnetic field line condition is broken due to microscopic effects, such as kinetic and particle-wave interactions. To understand the role of the microscopic trigger, such as complex individual motions of unmagnetized particles, extensive PIC simulations have been performed [3–5].

In space physics and astrophysics, there are also many important phenomena associated with kinetic collisionless plasmas. One example is the particle acceleration process in collisionless shocks; it is widely believed that in supernova remnants, cosmic rays with energies below 10^{15} eV are accelerated in collisionless shocks. Generally, these acceleration processes are kinetic and stochastic, and only a small fraction of particles are selectively accelerated to high energies. These processes have been studied with kinetic plasma simulations, and recent large-scale simulations show that some kind of acceleration processes indeed occur in collisionless shocks [6–8]. To investigate these processes in detail, it is essential to understand the acceleration histories of individual particles. For this purpose, it is desirable to record time sequences (histories) of positions, momenta, and energies for as many accelerated particles as possible and subsequently analyze them.

To study such related detailed motions, it is necessary to record the positions of all plasma particles at high sampling rates. In addition, smooth animations are expected to be one useful tool for analyzing complex plasma phenomena.

To efficiently record time sequences of positions, namely trajectories, a compression technique is necessary. As a versatile compression method, HDF-5 is widely used in various areas of computational simulation. HDF-5 is a hierarchical data format originally developed at the National Center for Supercomputing Applications. It is currently supported by the HDF Group [9]. HDF-5 is a set of file formats and libraries to store and organize large amounts of numerical data.

In compressed data, part of the file is occupied by random values smaller than the precision required for lossless compression. For molecular dynamics simulations of molecules, such as proteins, the XTC format is a portable format used as a lossy compression method [10, 11]. Trajectories are written using a reduced-precision algorithm that works in the following way: coordinates are multiplied by a scale factor (typically 1000) and then rounded to integer values. The algorithms for XTC are based on the observation that differences in atomic coordinates and velocities, in either time or space, are generally smaller than the absolute values of the coordinates and velocities themselves [12]. This is considered to be lossy compression with the elimination of random values that are less than the required precision; the result is suitable for smooth animations.

In the TOKI compression scheme, particle trajectory data are compressed with following features [13]:

- A particle trajectory is approximated by piecewise polynomial functions whose basis is related to the characteristic motion of the particle.
- The difference between the original trajectory and the polynomial approximation is guaranteed to be smaller than a given allowable constant. To ensure this, direct checks should be performed in an encoder.
- Time-sequence data of trajectories are treated independently for each particle. Thus, the divisions of trajectories into intervals are not the same for different particles.

In Section 2, we present the basic procedure used in a TOKI compression, and we describe the PIC simulation performed to obtain test data. Section 3 presents results from TOKI implementations with the least-squares method and preliminary results with Chebyshev approximation. Compression ratios from TOKI and XTC compression are compared for various sampling rates. Section 4 concludes the paper.

2. Method

2.1 TOKI compression with least-squares method

In the TOKI compression algorithm, we seek the longest interval over which the original trajectory can be approximated by a kth-degree polynomial within an allowed (given) error ε.

$$\max_{i=s_j}^{s_j+l_j-1} \left| x_i - \sum_{n=0}^{k} a_{n,j} t_i^n \right| < \varepsilon. \tag{1}$$

Here, $\{x_i\}$ denotes a time series of positions for a certain particle at times i, while s_j and l_j are the starting time and length (number of data points) of the jth interval, respectively. The quantities $a_{n,j}$ are coefficients of the kth-degree polynomial used for approximation in the jth interval. To reduce the file size, we record these coefficients, $a_{n,j}$, instead of $\{x_i\}$. When Eq. (1) is not satisfied for a minimum interval length, we record $\{x_i\}$ directly. Hence, the error in the compressed data is always, by definition, within the allowed error ε.

In our first implementation, we use the least-squares method to find the coefficients of the kth-degree polynomial in a certain interval (for ease of development of test codes). With the least-squares method,

$$\sum_{i=s_j}^{s_j+l_j-1} \left\{ x_i - \sum_{n=0}^{k} a_{n,j} t_i^n \right\}^2 \to \min, \tag{2}$$

we obtain values for the coefficients $a_{n,j}$. Then, we check whether the condition (1) is satisfied. When it is not even for the minimum interval length, we store $\{x_i\}$ directly.

The above procedure is performed in a work array. Here we present a brief explanation of the work array of length L_{work}.

1) The trajectory data for a selected particle are stored in a work array. Here, the number of stored values is L_{work}.
2) We fit the trajectory by a polynomial with the least-squares method (2) for the data in the work array whose length is L_{work}. Using the obtained fitting parameters $a_{n,j}$, we check condition (1).
3) When the condition (1) is satisfied, the parameters $a_{n,j}$ are recorded, then we move to step 1 for the next time sequence.
4) When condition (1) is not satisfied, we divide the interval in half and repeat the fitting with the least-squares method (2) for the data whose length is now $L_{\text{work}}/2$. Using the obtained fitting parameters $a_{n,j}$, we check condition (1).
5) When condition (1) is satisfied, we increase the length of the interval to seek the longest interval that can be represented by the approximation polynomial while satisfying (1). Otherwise, we decrease the length.
6) As a result, we obtain the longest interval and coefficients $a_{n,j}$ for that interval while preserving (1). Finally, these coefficients are recorded in a file. Then we move to step 1 for the next time sequence.

The length L_{work} of the work array limits the maximum interval for compression. Therefore, the size of the work array can significantly affect the efficiency of the compression. For example, consider a data set with 1024 raw values can be compressed in one interval to obtain one set of fitting parameters $a_{n,j}$. For $L_{\text{work}} = 1024$, one parameter set $a_{n,j}$ (namely, four values for a third-degree polynomial) is recorded. In contrast, for $L_{\text{work}} = 256$, the entire data must be divided into at least four intervals and four sets of parameters $a_{n,j}$ (16 values) must be recorded. Thus, the recorded data size with $L_{\text{work}} = 256$ becomes four times that with $L_{\text{work}} = 1024$.

To illustrate the relation between the raw data in a time sequence of values and the compressed parameter sets, Fig. 2 shows an illustration of TOKI compression with third-degree polynomials.

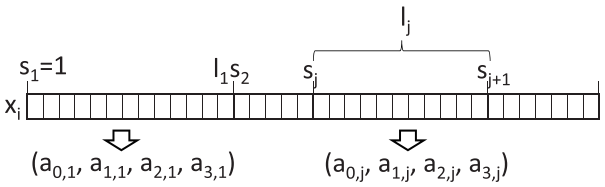

Fig. 2 Illustration of TOKI compression using third-degree polynomials.

2.2 TOKI compression with Chebyshev approximation

Instead of the least-squares method discussed above, we can use Chebyshev approximation [14] to find the coefficients of kth-degree polynomials. Chebyshev polynomials, which are orthogonal polynomials, are used to expand the particle trajectories. A Chebyshev approximation is similar to the minimax approximation that minimizes the *maximum* error within an interval, while the least squares method minimizes the *average* error. Therefore, Chebyshev polynomials are more favorable for satisfying condition (1) and, as a result, the intervals of trajectories can be made longer.

Chebyshev polynomials are defined by

$$T_n(x) = \cos(n \arccos x), \tag{3}$$

where n is an integer. With these polynomials, a function $f(x)$ can be approximated by a Chebyshev expansion

$$f(x) = \sum_n a_n T_n(x). \tag{4}$$

The coefficients in this expansion are determined from the following integral formula, which is similar to that of a Fourier expansion:

$$a_n = \frac{2}{\pi} \int_{-1}^{1} f(x) \frac{T_n(x)}{\sqrt{1-x^2}} \mathrm{d}x. \tag{5}$$

This integral can be calculated by direct numerical integration or by evaluating the integrand at predetermined sample points. Many numerical library programs support the latter method.

2.3 Test data from a PIC simulation

We performed a plasma particle simulation for 24,000 time steps (about 800 s at 30 frames/s) and obtained three-dimensional positions of 150,000 electrons and 150,000 ions. In this simulation, we solved the dynamics of the particles and the electromagnetic field self-consistently by an explicit electromagnetic PIC method [1]. The initial condition was given by a one-dimensional Harris-type equilibrium [15] in which the magnetic field was parallel to the x-axis and a function of the y-coordinate. The initial velocity distribution of particles was Maxwellian with a uniform temperature in which the thermal velocities of electrons and ions were respectively $0.256c$ and $0.0256c$,

where c denotes the speed of light. The ratio of ion mass to electron mass was 100. Periodic boundary conditions were employed, and the simulation box size was $3.85c/\omega_{ce} \times 1.92c/\omega_{ce} \times 3.08c/\omega_{ce}$, where ω_{ce} is the electron cyclotron frequency. We set the numbers of grids for field variables (electromagnetic field, charge and current densities) to $16 \times 15 \times 16$ as a test case. Based on the CFL (Courant–Friedrichs–Lewy) condition, $\Delta t_{CFL} = (0.5/c)\min(\mathrm{d}x, \mathrm{d}y, \mathrm{d}z)$ was used to avoid numerical instabilities. To obtain smooth particle trajectories, we set the sampling rate to $\Delta t_{sampling} = 0.016/\omega_{ce}$, which is a quarter of $\Delta t_{CFL} = (0.5/\omega_{ce})\min(3.85/16, 1.92/15, 3.08/16)$.

3. Results

3.1 TOKI compression with least-squares method

We examined the effects of the length of work array L_{work} and allowed error ε. To find the longest interval for polynomials, the bisection method was used for speed in the first implementation of the code. We measured the ratio of compressed data size to original data size; the latter is given by the size of all data in double precision binary format, that is, 8 bytes × 24,000 for one particle. For this paper, the compressed data size was calculated in the assumed data format to avoid the overhead involved in extracting data from the given format. Here, we recorded the compressed data of a certain interval in the following format:

1) start time: integer (4 bytes)
2) number of compressed data points: integer (4 bytes)
3) coefficients of polynomials: double (8 bytes ×(k+1))

Table 1 shows compression ratios for $L_{work} = 256$ and 1024, and for $\varepsilon = 0.001$ and 0.01. It is clear that the size of the work array and the value of the allowed error both impact the compression ratio. It is found that relative size becomes smaller for higher degree polynomials for large L_{work}. By using a higher degree polynomial for fit, although the number of coefficients to express a polynomial in each interval, $(k + 1)$, increases, the length (number of data points) of each interval, l_j, generally increases, too. This decreases the number of intervals for a particle trajectory. Therefore, the total size of the compressed data (the number of coefficients of the polynomial in an interval times the number of intervals for the trajectory) can be smaller depending on the characteristic of the trajectory. It denotes existence of optimal degree for given data set. We can see that the optimal degree of ions' trajectories is less than 6 for $L_{work} = 256$ and those of otherwise are not less than 3. We considered the optimal degree is related to behaviors of each particle. It is considered that given simulation conditions determine the optimal degree as well as behaviors of each particle. Although higher degree seems to be better for large L_{work}, there is also a trade-off in computing costs (elapsed time of compression). For sixth-degree polynomials, the Householder method (before optimization) was used in our implementation. This method is much slower than direct solution for the third-degree polynomials.

Table 1 Relative sizes of compressed data files using third- and sixth-degree polynomials.

	3rd (ε=0.001)	6th (ε=0.001)	3rd (ε=0.01)	6th (ε=0.01)
Elec/Ion	0.2858	0.2258	0.1517	0.1258
L_{work} = 1024	0.0385	0.0326	0.0193	0.0185
Elec/Ion	0.2948	0.2331	0.1565	0.1303
L_{work} = 256	0.0463	0.0741	0.0439	0.0700

Table 1 shows that the relative sizes of compressed data for ions are much smaller than those for electrons; this is because the motions of ions are relatively slow and the trajectories are smoother due to the difference in mass. In the simulations, the mass of ions was 100 times that of electrons. From a rough estimate based on the square root of the mass ratio, we expected the size of compressed data for electrons to be 10 times that for ions. For $L_{work} = 1024$, the obtained results are almost consistent with this rough estimation. However, this is not the case for $L_{work} = 256$. We think the reason is the shortness of L_{work}. Details of effects of work array length L_{work} on compression ratio should be examined in the future.

3.2 TOKI compression with Chebyshev approximation

As another test for efficient compression by the TOKI scheme, we tried Chebyshev approximation. To find the coefficients of the polynomials in this method, we used the library function gsl_cheb_eval_n_err() from the GNU Scientific Library [16]. This function approximately employs the sample points method mentioned in Sec. 2.2. In this function, we can change the degree of polynomials and set the maximum degree of polynomials, k_{max}. For points in the range $[a, b]$ and for the allowed error ε, we can use this library function to obtain the degree and coefficients of Chebyshev polynomials. In this preliminary implementation, we estimated the error in Chebyshev approximation with k_{max}-th degree polynomials and given maximum length L_{work} for trial. When this error is smaller than the allowed error ε, we seek the polynomial of lowest degree, k, that satisfies condition (1) on the error. Otherwise, we find the largest length for the k_{max}-th degree polynomial.

We examined how the size of the compressed data is affected by the length of the work array L_{work} and the maximum degree of Chebyshev polynomials k_{max}. We recorded the compressed data of a certain interval in the following format:

1) start time: integer (4 bytes)
2) length of this interval (4 bytes)
3) degree of Chebyshev polynomials: integer (4 bytes)
4) coefficients of polynomials: double (8 bytes × k)

Table 2 shows relative sizes of compressed data files for $L_{\rm work}$ = 256, 1024, and 4096, and for $k_{\rm max}$ = 3, 6, 10, 20, and 40. Here, we set $\varepsilon = 0.001$. The table shows that the compressed sizes using Chebyshev approximation are smaller than those for the usual polynomials with the least-squares method discussed in Section 3.1 (compare with Table 1).

For large $k_{\rm max}$ and $L_{\rm work}$, the relative size of the compressed file becomes smaller. For large $k_{\rm max}$ and large $L_{\rm work}$, the relative size becomes smaller. For large $k_{\rm max}$ and small $L_{\rm work}$, no difference occurs between electrons and ions, while a difference does appear for large $L_{\rm work}$ and $k_{\rm max}$. These behaviors of relative size for value of $k_{\rm max}$ denote existence of optimal value of $k_{\rm max}$ for given data set, $L_{\rm work}$ and our implemented algorithm. More details should be examined in the future.

It is noted that algorithm to choose length l_j (number of data points) and degree k of Chebyshev polynomials should be modified in order to obtain smallest file size. Although the algorithm of the present paper selected lowest degree for fixed $L_{\rm work}$ or largest l_j for fixed $k_{\rm max}$, we should find pair of k and l_j to minimize value of k/l_j. For this modified algorithm, computing cost becomes too much because computation of many pairs of k and l_j is required.

3.3 Comparison with other compression schemes

Sections 3.1 and 3.2 show that the relative size of a compressed data file depends on features of particle trajectories. Thus, we expect that the relative size of compressed data can be used as a representative parameter to categorize trajectories. So, we used other compression methods to test whether this idea is valid. Here, we checked the XTC format [4], which is widely used as a compression method for trajectory data in particle systems.

The sampling rate $\Delta t_{\rm sampling}$ is an important parameter in TOKI compression because $\Delta t_{\rm sampling}$ is directly related to the smoothness of the trajectories. We checked the effect of $\Delta t_{\rm sampling}$ on the compression efficiency for both TOKI compression and the XTC format.

The data for 24,000 time steps of three-dimensional positions for 300,000 particles requires about 525 GB in Fortran double-precision text format. For 30 frames/s, this corresponds to 657 MB/s, which is faster than SSD. Table 3 shows the sizes of compressed files from the XTC and TOKI schemes (with the least-squares method) at two precisions indicated by decimals 0.001 and 0.01. In the XTC format, trajectories were written as a difference of bits of rounded integers multiplied by a scale factor (typically 1000 for 0.001 accuracy).

When the time interval $\Delta t_{\rm sampling}$ is smaller than $\Delta t_{\rm CFL}$, the TOKI scheme shows a good compression ratio. For smooth trajectories ($\Delta t_{\rm sampling} \ll \Delta t_{\rm CFL}$), the ratio for TOKI is much better than that for the XTC format. However, for ballistic trajectories ($\Delta t_{\rm sampling} \gg \Delta t_{\rm CFL}$), the compression ratio for TOKI is not good because of the overhead associated with storing raw values of trajectories. We found that the size of TOKI compressed files depends on the motion of each particle, while the size for XTC is almost proportional to the number of frames and is independent of the motion of each particle. From this, we expect that the behavior of the compression ratio for TOKI can be used as a characterization parameter for particle motions.

Table 2 Relative sizes of compressed files by Chebyshev approximation for $\varepsilon = 0.001$.

	$k_{\rm max}$= 3	6	10	20	40
Elec/Ion	0.1632	0.1041	0.0928	0.0876	0.0877
$L_{\rm work}$= 4096	0.0340	0.0201	0.0178	0.0172	0.0247
Elec/Ion	0.2238	0.1334	0.1175	0.1113	0.1161
$L_{\rm work}$=1024	0.0369	0.0241	0.0290	0.0509	0.0957
Elec/Ion	0.2816	0.1592	0.1433	0.2032	0.3782
$L_{\rm work}$= 256	0.0536	0.0709	0.1103	0.1983	0.3745

4. Conclusions

We have confirmed that the TOKI compression scheme can reduce the size of data files by a lossy compression method within the allowed error. We still need to optimize the implemented codes to reduce the computing time for compression and decoding. For this, we chose the approximation using third-degree polynomials to keep precision and avoid the Householder computation. We also tested implementation of TOKI compression using Chebyshev approximation via the GSL library. We clarified the relation between $\Delta t_{\rm sampling}$ and performance of compression. TOKI compression is reliable for smooth trajectories that correspond to the condition $\Delta t_{\rm sampling} \ll \Delta t_{\rm CFL}$. For these situations, the performance of TOKI is much better than that of XTC. By comparing the compression performance of electrons and ions, we found performance of ions was better than that of electrons due to the difference in characteristic time scales of the motions, which, in turn,

Table 3 Comparisons of sizes of compressed files from XTC and TOKI.

Filesize (GB) Total / Electrons / Ions	24000frames $\Delta t_{\rm sampling}$ =0.016/$\omega_{\rm ce}$	6000frames $\Delta t_{\rm sampling}$ =0.048/$\omega_{\rm ce}$	1000frames $\Delta t_{\rm sampling}$ =0.384/$\omega_{\rm ce}$
Text (double)	525.6 / 262.8 / 262.8	131.4 / 65.7 / 65.7	21.9 / 11.0 / 11.0
Binary (double)	172.9 / 86.4 / 86.4	43.2 / 21.6 / 21.6	7.20 / 3.60 / 3.60
XTC(0.001)	43.7 / 22.7 / 21.1	11.0 / 5.67 / 5.29	1.82 / 0.94 / 0.87
TOKI(0.001)	17.1 / 14.7 / 2.41	5.13 / 4.28 / 0.85	2.77 / 1.49 / 1.28
XTC(0.01)	34.8 / 18.2 / 16.6	8.69 / 4.55 / 4.14	1.45 / 0.76 / 0.69
TOKI(0.01)	10.0 / 7.82 / 2.20	3.55 / 3.09 / 0.45	2.31 / 1.46 / 0.85

is due to the difference in mass. We expect that the behavior of the compression ratio from TOKI can be used to characterize the motions of plasma particles. In future work, distribution of TOKI software as a system software library will be considered. Studies toward this goal are in progress.

Acknowledgements

This work is partly supported by "Joint Usage/Research Center for Interdisciplinary Large-scale Information Infrastructures" (JHPCN-HPCI Project ID: jh130038, hp130062, and hp130122) in Japan, NIFS/NINS under the project of Formation of International Scientific Base and Network, and HPC project of Nagoya University. The authors thank Dr. A. M. Ito, Dr. T. Takeda, and Dr. T. Saito for useful discussions at early stage of this work under National Institutes of Natural Sciences (NINS) Program for Cross-Disciplinary Study.

[1] C.K. Birdsall and A.B. Langdon, *Plasma Physics Via Computer Simulation* (McGraw-Hill, New York, 1985).
[2] F.F. Chen, *Introduction to Plasma Physics* (Plenum Press, New York, 1974).
[3] W. Horton and T. Tajima, J. Geophys. Res. **96**, 15811 (1991).
[4] A. Ishizawa and R. Horiuchi, Phys. Rev. Lett. **95**, 045003 (2005).
[5] W. Daughton *et al.*, Nature Physics **7**, 539 (2011).
[6] T. Sugiyama, Phys. Plasmas **18**, 022302 (2011).
[7] M.A. Riquelme and A. Spitkovsky, Astrophys. J. **733**, 63 (2011).
[8] Y. Matsumoto *et al.*, Astrophys. J. **755**, 109 (2012).
[9] http://www.hdfgroup.org/
[10] D.G. Green, K.E. Meacham, M. Surridge, F. van Hoesel and H.J.C. Berendsen, Metecc-95 Proceedings (1955).
[11] http://www.gromacs.org/
[12] D. Spangberg, D.S.D. Larsson and D. van der Spoel, J. Mol. Model **17**, 2669 (2011).
[13] H. Ohtani, K. Hagita, A.M. Ito, T. Kato, T. Saitoh and T. Takeda, J. Phys: Conf. Ser. **454**, 012078 (2013).
[14] R.W. Hamming, *Numerical Methods for Scientists and Engineers*, second edition (McGraw-Hill, New York, 1973).
[15] E.G. Harris, Nuovo Cimento **23**, 115 (1962).
[16] http://www.gnu.org/software/gsl/