The Astrophysical Journal, 536:L39–L42, 2000 June 10

A VERY FAST AND MOMENTUM-CONSERVING TREE CODE

Walter Dehnen
Max-Planck-Institut für Astronomie, Königstuhl 17, D-69117 Heidelberg, Germany; dehnen@mpia-hd.mpg.de
Received 2000 March 13; accepted 2000 April 25; published 2000 June 6

ABSTRACT

The tree code for the approximate evaluation of gravitational forces is extended and substantially accelerated by including mutual cell-cell interactions. These are computed by a Taylor series in Cartesian coordinates and in a completely symmetric fashion, such that Newton's third law is satisfied by construction and that therefore momentum is exactly conserved. The computational effort is further reduced by exploiting the mutual symmetry of the interactions. For typical astrophysical problems with $N = 10^5$ and at the same level of accuracy, the new code is about 4 times faster than the tree code. For large N, the computational costs are found to scale almost linearly with N, which can also be supported by a theoretical argument, and the advantage over the tree code increases with ever larger N.

Subject headings: celestial mechanics, stellar dynamics — methods: n-body simulations — methods: numerical

1. INTRODUCTION

The tree code (see Barnes & Hut 1986, hereafter B&H) has become an invaluable tool for the approximate but fast computation of the forces in studies of collisionless gravitational dynamics. It has been applied to a large variety of astrophysical problems. The gravitational potential generated by N bodies of masses μ_n and at positions $\boldsymbol{X}_n$ is

$$\Phi(\boldsymbol{X}) = -\sum_{n=1}^{N} \mu_n g(|\boldsymbol{X} - \boldsymbol{X}_n|), \qquad (1)$$

where $g(r)$ denotes the Green's function; i.e., for unsoftened gravity $g(r) = G/r$. The essence of the tree code is to approximate this sum over N terms by replacing any partial sum over all bodies within a single cell that is well separated from $\boldsymbol{X}$ by just one term. The inner structure of the cell is partly taken into account using its multipole moments. This method reduces the overall costs for the computation of all forces from $O(N^2)$ to $O(N \log N)$.

The tree code, however, does not exploit the fact that the force generated by the contents of some source cell is very similar at similar sink positions (Barnes 1990 took advantage of the related fact that adjoining bodies have very similar interaction lists). Exploiting this is the idea behind the fast multipole method (FMM; Greengard & Rokhlin 1987). With the FMM, one employs a (usually) nonadaptive structure of hierarchical grids and considers only interactions between nodes on the same grid level according to their geometrical neighborhood. The gravitational field within some sink cell and generated by some source cell is approximated by a multipole expansion in spherical harmonics, the order of which is adapted to meet predefined accuracy limits. This method has been claimed to reduce the overall amount of operations to $O(N)$, but the tables given by Cheng, Greengard, & Rokhlin (1999) do not support this claim. Capuzzo-Dolcetta & Miocchi (1998) find that the FMM needs $O(N \log N)$ operations and is significantly *slower* for astrophysical applications than the tree code at comparable accuracy.

Instead of using a spherical multipole expansion of adaptive order, it is actually more efficient to use a Cartesian expansion of fixed order. Moreover, by preserving the symmetry of the gravitational interaction for mutual cell-cell interactions, one can first reduce the computational effort and second obtain a code that satisfies Newton's third law by construction and hence results in the exact conservation of momentum, a property not shared by the traditional tree code.

2. DESCRIPTION OF THE CODE

Like the B&H tree code, we use a hierarchical tree of cubic cells. Each cell has up to eight subnodes corresponding to its octants. A node can be either a single body or another cell. The tree-building phase (see B&H) also includes the computation of the cells' masses, centers of masses, and quadrupole moments.

2.1. *The Opening Criterion*

In order to benefit from the symmetry of the gravitational interaction, the opening criterion, which decides whether or not two nodes are well separated so that a direct mutual interaction is acceptable, must be symmetric too. We employ an extension of the criterion used in the tree code: nodes A and B are well separated if

$$|\boldsymbol{Z}_{\rm A} - \boldsymbol{Z}_{\rm B}| > (r_{\rm max,\,A} + r_{\rm max,\,B})/\theta, \qquad (2)$$

where the opening angle θ controls the accuracy of the code; $r_{\rm max}$ is the radius of a sphere centered on the node's center of mass $\boldsymbol{Z}$ and encircling all bodies within it. Bodies naturally have $r_{\rm max} \equiv 0$ (i.e., two bodies are always well separated), while for the interaction between a body and a cell, the criterion in equation (2) reduces to that used in the tree code. Note that if, in addition to equation (2), one requires $r_{\rm max,\,A} = 0$, the standard tree code is recovered, but the symmetry between A and B is broken.

There exist two upper limits for the radius $r_{\rm max}$. One is the distance $b_{\rm max}$ between the cell's center of mass $\boldsymbol{Z}$ and its most distant corner (Salmon & Warren 1994). The other is

$$\max_{\text{subnodes},\, i} \{r_{{\rm max},\,i} + |\boldsymbol{Z}_i - \boldsymbol{Z}|\} \qquad (3)$$

(Benz et al. 1990). After computation of both these upper limits, we take the smaller one to be $r_{\rm max}$. For cells with only a few bodies (like cell A in Fig. 1), the latter often gives values significantly smaller than $b_{\max}$, while for cells with many bodies (like cell B in Fig. 1), $b_{\max}$ is the tighter limit.

2.2. *Approximating Gravity*

Consider two bodies at $\boldsymbol{X}$ and $\boldsymbol{Y}$ that reside in two well-separated cells A and B with centers of mass at, respectively, $\boldsymbol{Z}_{\mathrm{A}}$ and $\boldsymbol{Z}_{\mathrm{B}}$ and separation $\boldsymbol{R} \equiv \boldsymbol{Z}_{\mathrm{A}} - \boldsymbol{Z}_{\mathrm{B}}$. We may rewrite $\boldsymbol{X} - \boldsymbol{Y} = \boldsymbol{R} + (\boldsymbol{x} - \boldsymbol{y})$, with $\boldsymbol{x} \equiv \boldsymbol{X} - \boldsymbol{Z}_{\mathrm{A}}$ and $\boldsymbol{y} \equiv \boldsymbol{Y} - \boldsymbol{Z}_{\mathrm{B}}$ being small in magnitude compared with $\boldsymbol{R}$ (because the cells are well separated; see Fig. 1). The Taylor expansion of $g(|\boldsymbol{X} - \boldsymbol{Y}|)$ around $\boldsymbol{R}$ reads

$$g(|\boldsymbol{X} - \boldsymbol{Y}|) = \sum_p \frac{1}{p!} \{[(\boldsymbol{x} - \boldsymbol{y}) \cdot \boldsymbol{\nabla}]^p g(|\boldsymbol{r}|)\}_{\boldsymbol{r}=\boldsymbol{R}}. \tag{4}$$

Separating powers of $\boldsymbol{x}$ from powers of $\boldsymbol{y}$ in equation (4) and subsequently taking the mass-weighted sum over cell B yields a Cartesian multipole expansion of the potential $\Phi_{\mathrm{B}\to\mathrm{A}}(\boldsymbol{X})$ at any position $\boldsymbol{X}$ within cell A and generated by all bodies inside cell B (Warren & Salmon 1995). Since $\boldsymbol{Z}_{\mathrm{B}}$ was chosen to be the center of mass of cell B, its dipole vanishes. The highest order multipole occurring in such an expansion may actually be omitted since it only contributes a constant to the approximation for g and does not affect the approximation for $\boldsymbol{\nabla} g$ (and hence the force). The expression of third order thus reads (without the octopole and using Einstein's sum convention)

$$\Phi_{\mathrm{B}\to\mathrm{A}}(\boldsymbol{X}) \approx M_{\mathrm{B}}\Big[\mathcal{D}^{(0)} + \tfrac{1}{2}\tilde{Q}_{\mathrm{B},ij}\mathcal{D}^{(2)}_{ij} + x_i(\mathcal{D}^{(1)}_i + \tilde{Q}_{\mathrm{B},jk}\mathcal{D}^{(3)}_{ijk}) + \frac{1}{2}x_i x_j \mathcal{D}^{(2)}_{ij} + \frac{1}{6}x_i x_j x_k \mathcal{D}^{(3)}_{ijk}\Big], \tag{5}$$

where M_{B} and $\tilde{Q}_{\mathrm{B}}$ are the mass and *specific* quadrupole moment

$$\tilde{Q}_{\mathrm{B},ij} \equiv \frac{1}{M_{\mathrm{B}}} \sum_{\boldsymbol{y}_n + \boldsymbol{Z}_{\mathrm{B}} \in \text{cell B}} \mu_n y_{ni} y_{nj} \tag{6}$$

of cell B, while $\mathcal{D}^{(n)} \equiv \boldsymbol{\nabla}^n g(r)|_{r=|\boldsymbol{R}|}$; i.e.,

$$\mathcal{D}^{(0)} = D^0, \tag{7a}$$

$$\mathcal{D}^{(1)}_i = R_i D^1, \tag{7b}$$

$$\mathcal{D}^{(2)}_{ij} = \delta_{ij} D^1 + R_i R_j D^2, \tag{7c}$$

$$\mathcal{D}^{(3)}_{ijk} = (\delta_{ij} R_k + \delta_{jk} R_i + \delta_{ki} R_j) D^2 + R_i R_j R_k D^3, \tag{7d}$$

with

$$D^n \equiv \left| \left(\frac{1}{r}\frac{\partial}{\partial r}\right)^n g(r) \right|_{r=|\boldsymbol{R}|}. \tag{8}$$

The symmetry between $\boldsymbol{x}$ and $\boldsymbol{y}$ at every order of the Taylor expansion in equation (4) has two important consequences. First, if this expansion is used to compute both $\Phi_{\mathrm{B}\to\mathrm{A}}(\boldsymbol{X})$ and $\Phi_{\mathrm{A}\to\mathrm{B}}(\boldsymbol{Y})$, Newton's third law is satisfied by construction. Note that our omission of the octopole term broke the symmetry only in the zeroth order and has no effect on the forces.

Second, the expressions for $M_{\mathrm{A}}\Phi_{\mathrm{B}\to\mathrm{A}}$ and $M_{\mathrm{B}}\Phi_{\mathrm{A}\to\mathrm{B}}$ are very similar: the expansion coefficients of second and third order differ only by mere signs, such that computing these coefficients for both Taylor series at one time is substantially faster than computing them at different times.

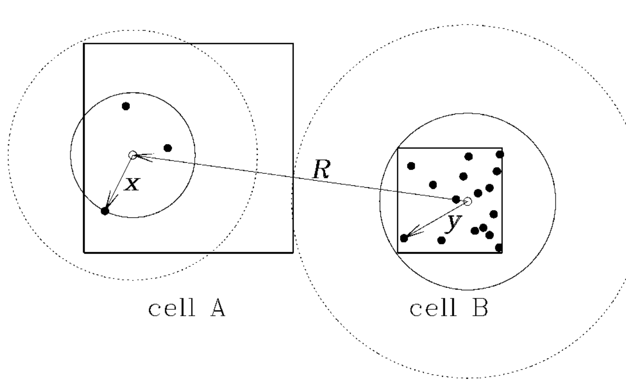

FIG. 1.—Two well-separated cells. The solid and dotted circles have radii $r_{\max}$ and $r_{\max}/\theta$, respectively.

The transformation, or shifting, of the expansion center to some other position is trivial compared with the analogous procedure in the FMM (see Cheng et al. 1999).

2.3. *The Algorithm*

The standard tree code computes the forces on each body by a recursive tree walk, which visits each node exactly once as a gravity sink, and thus exhibits an inherent asymmetry between sources and sinks. The new algorithm avoids this asymmetry.

First, in the interaction phase, the Taylor series coefficients are evaluated and accumulated in data fields associated with each node. This phase is based on the concept of *mutual interactions* (MIs), pairs of nodes, A and B, such that bodies in node A must receive forces from all bodies in node B and vice versa. We start by the MI describing the root-root self-interaction, and we process a given MI as follows: a body self-interaction is ignored; (2) a cell self-interaction is split into the MIs between the subnodes,[1] and the process is continued on each of the new MIs; (3) an MI representing a well-separated pair of nodes is executed: the Taylor coefficients are computed and added to the nodes' corresponding data fields; and (4) finally, in any other case, the node with larger $r_{\max}$ is split, and up to eight new MIs are created and processed.

Second, in the collection phase, the Taylor coefficients are passed down the tree, the expansion center is shifted to the currently active cell, and adding to its coefficients. The Taylor expansion is evaluated at the position of any body, and the values for potential and acceleration are added to its data fields (which may already contain contributions accumulated during the interaction phase).

3. PERFORMANCE TESTS

We tested the new algorithm and compared it with the tree code in three typical astrophysical situations: (1) a spherical Plummer model, representing a rather homogeneous stellar system, (2) a spherical Hernquist (1990) model galaxy, and (3) a group of five such galaxies with various masses and scale radii. We generated 10^5 random initial positions from each of these cases, truncating the density at 1000 scale radii, and evaluated the exact mutual forces at all positions and the approximated forces due to the tree code (up to quadrupole order) and the new code for opening angles θ between 0.2 and 1. We used an optimally chosen softening with the biweight softening ker-

[1] In a cubic oct-tree, these are at most 36 independent sub-MIs.

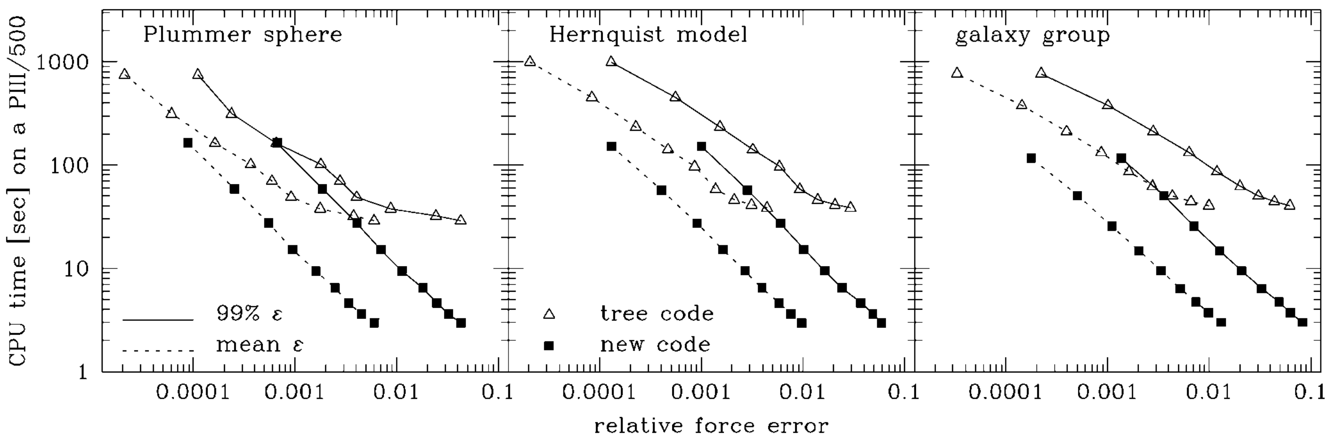

FIG. 2.—CPU time consumption plotted vs. the mean and 99% relative force error for the three test cases. The symbols correspond, from left to right, to θ = 0.2, 0.3, 0. 4, 0.5, 0.6, 0.7, 0.8, 0.9, and 1.

nel (see Dehnen 2000), but the results are insensitive to these settings. Both approximate methods have been coded by the author[2] and use the same opening criterion (§ 2.1). In order to measure the accuracy of the approximated forces, we evaluated (Capuzzo-Dolcetta & Miocchi 1998)

$$\epsilon_n = |a_n - a_n^{PP}| \Big/ a_n^{PP}, \tag{9}$$

where a_n denotes the magnitude of the acceleration of the nth body due to either of the approximate methods and a_n^{PP} denotes the magnitude of the exact computation.

Figure 2 plots the CPU time needed for the force approximation on a Pentium III/500 MHz PC versus the mean relative error and that at the 99th percentile. For the new code, the time consumption scales almost inversely with the error, while the tree code flattens off[3] at $\theta \gtrsim 0.7$. Evidently, at an acceptable level of accuracy, e.g., $\epsilon_{99\%} = 0.01$, the new code is about 4 times faster than the tree code, even though it requires a smaller value of θ (0.5 as compared with 0.7 for the tree code).

For the test case of the Hernquist model, Figure 3 plots the CPU time consumption per body versus N at fixed $\theta = 0.7$ for the tree code and 0.5 for the new code. The tree code shows the well-known log N scaling, while for $N \gtrsim 10^5$ the new code requires only a constant amount of CPU time per body. This can be explained as follows. Arranging eight root cells to a new root box increases N to $8N$ and the number N_I of interactions to $8(N_I + N_+)$, where $8N_+$ interactions are needed to compute the forces between the former root cells. Thus,

$$\frac{\mathrm{d}N_I}{\mathrm{d}N} \simeq \frac{N_I}{N}\frac{\Delta \log N_I}{\Delta \log N} = \frac{N_I + N_+/\ln 8}{N}. \tag{10}$$

In the tree code, $N_+ \propto N$, yielding $N_I \propto N \log N$. In the new code, N_+ may be estimated to contain two contributions, a constant term that accounts for the interactions with distant nodes and a term that is proportional to $N^{2/3}$ for those on the *surface* of the former root cells. Inserting this into equation (10) yields

$$N_I \propto N - c_1 N^{2/3} - c_2, \tag{11}$$

with constants c_1 and c_2 that depend on θ. Thus, at large N, a linear relation is approached. Note that this argument differs from that given for the FMM by Greengard & Rokhlin (1987), who assumed that the resolution may remain fixed when increasing N.

4. DISCUSSION

A new code for the approximate evaluation of gravitational forces has been presented, tested, and compared with the tree code. This new code is substantially faster than the tree code. Moreover, unlike the latter, it satisfies Newton's third law by construction, such that any N-body code based on it will not introduce spurious net accelerations. The new code is based on a Taylor expansion of the Green's function in Cartesian coordinates and incorporates mutual cell-cell interactions. The simple algorithm is well suited for implementation on parallel computers: different MIs can be passed to different CPUs.

The scaling of the CPU time required for the mutual forces of a number N of bodies becomes essentially linear at

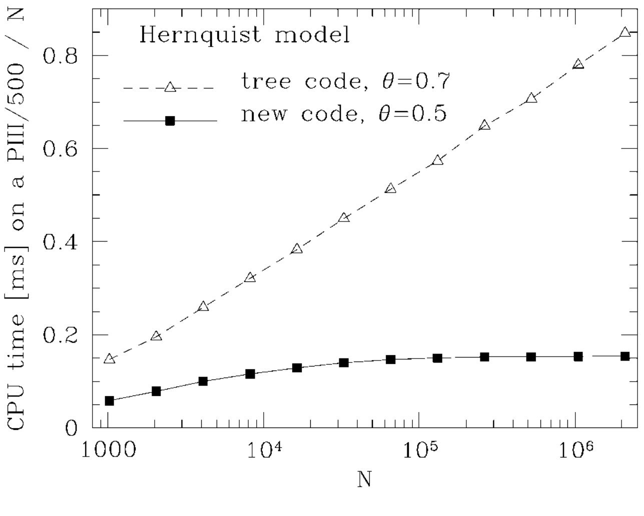

FIG. 3.—CPU time per body plotted vs. N for test case 2 (Hernquist model)

[2] At the same θ, the tree code is twice as fast as a code publicly available from J. Barnes, mainly because the new opening criterion leads to fewer interactions. However, even at the same number of interactions, the author's code was about 30% faster.

[3] This is because $r_{\max}$ is not proportional to the cell size, so that increasing θ from 0.7 does not decrease the number of interactions by much.

$N \gtrsim 10^5$, so that with ever larger N the new code is increasingly faster than the tree code, allowing for a substantial improvement in simulations employing large number of bodies. The only disadvantage is the increased requirement of memory compared with the standard tree code: 20 floating point numbers per cell are needed to hold the Taylor expansion coefficients. (By using a tree-walking algorithm instead of that given in § 2.3, one can avoid this at the price of enhanced CPU time consumption.)

In spirit, the new code is similar to Greengard & Rokhlin's (1987) fast multipole method, but it is more efficient because it uses a Cartesian instead of a spherical-harmonic multipole expansion and fixes the order of the expansion, while controlling the accuracy via the interaction condition rather than fixing the interactions and adapting the expansion order to the accuracy.

One concern that one may have with codes that are based on cell-cell interactions is their performance in the presence of individual time steps. Clearly, when not all the forces are to be computed, such codes fare less favorably. However, when the forces for all bodies within some domain are desired, the new code is still a significant improvement over the tree code.

The new code has been written in C++ and is electronically available from the author upon request.

The author thanks Tom Quinn for valuable discussions and Joshua Barnes, Lars Hernquist, Junichiro Makino, Andy Nelson, and Thorsten Naab for useful comments.

REFERENCES

Barnes, J. E. 1990, J. Comput. Phys., 87, 161
Barnes, J. E., & Hut, P. 1986, Nature, 324, 446 (B&H)
Benz, W., Bowers, R. L., Cameron, A. G. W., & Press, W. H. 1990, ApJ, 348, 647
Capuzzo-Dolcetta, R., & Miocchi, P. 1998, J. Comput. Phys., 143, 29
Cheng, H., Greengard, L., & Rokhlin, V. 1999, J. Comput. Phys., 155, 468
Dehnen, W. 2000, MNRAS, submitted
Greengard, L., & Rokhlin, V. 1987, J. Comput. Phys., 73, 325
Hernquist, L. 1990, ApJ, 356, 359
Salmon, J. K., & Warren, M. S. 1994, J. Comput. Phys., 111, 136
Warren, M. S., & Salmon, J. K. 1995, Comput. Phys. Commun., 87, 266