Random-walk statistics in moment-based $O(N)$ tight binding and applications in carbon nanotubes

Adam D. Schuyler and G. S. Chirikjian
Department of Mechanical Engineering, The Johns Hopkins University, Baltimore, Maryland 21218, USA

Jun-Qiang Lu and H. T. Johnson*
Department of Mechanical and Industrial Engineering, University of Illinois, Urbana, Illinois 61801, USA

(Received 8 December 2004; published 1 April 2005)

A computational framework for a moment-based $O(N)$ tight-binding atomistic method is presented, analyzed, and applied to the problem of electronic properties of deformed carbon nanotubes, where N is the number of atoms in the system. The moment-based approach is based on the maximum entropy and kernel polynomial methods for constructing the electronic density of states from local statistical information about the environment around individual atoms. Random-walk statistics are formally presented as the basis for several methods to collect the moments of the density of states in a computationally efficient manner. The computational complexity and accuracy of these methods are systematically analyzed. Using these methods for the problem of deformed carbon nanotubes, it is shown that the computational cost for some cases, per atom, scales as efficiently as $O(M \log M)$, where M is the desired number of moments in the expansion of the density of states. These methods are compared to other methods such as direct diagonalization and a Green's function approach.

DOI: 10.1103/PhysRevE.71.046701 PACS number(s): 02.70.−c, 71.15.−m, 73.22.−f, 72.80.Rj

I. INTRODUCTION

Semiempirical tight binding is an atomistic modeling method in which the total energy is assumed to be due to bonding valence electronic interactions and repulsive interactions primarily between nuclei. For regular, symmetric, and even uniformly strained atomic scale structures, a tight-binding model can be formulated in reciprocal space so that the electronic structure and total energy are derived from a basis as small as just a few atoms and their associated valence electrons. If a structure is nonuniform due to strain, defects, or impurities, for example, the tight-binding Hamiltonian in general should contain degrees of freedom for all valence electrons associated with every atom in the structure. Solving for the band structure and total energy using conventional methods then requires a costly direct diagonalization of the full Hamiltonian matrix that can scale as poorly as $O(N^3)$, where N is the number of atoms in the system. Within a tight-binding approximation, it is also possible to obtain information about the local density of states (LDOS) from the local structural environment around an atom. An information theoretic approach can be used to reconstruct the LDOS from its moments, or products of hopping integrals connecting random walks starting and finishing on a particular atom [1–3]. Thus, based on a collection of statistical information about local atomistic structure, the LDOS can be approximated, and the density of states (DOS) and the total energy of the whole system, in turn, can be computed in an $O(N)$ manner. In this statistical approach, the accuracy is dependent on the number of moments used in the approximation; sharp features of the DOS are poorly reproduced if insufficient moments are used in the expansion, while the total energy is well approximated with relatively few moments [4].

Two key components of the moments method for computing electronic structure and total energies are (i) the computational algorithm used to construct the LDOS from the set of moments, which can strongly affect the energy resolution and accuracy, and (ii) the computational algorithm for collecting the moment data, which is typically the most costly part of the technique and is likely the reason why the method has seen relatively little use over the last decade. Typically, the maximum entropy method (MEM) is used to determine the LDOS from the power moment data and assumes that the correct LDOS can be determined by maximizing the information theoretic entropy associated with all possible distributions consistent with the moments data [5,6]. The MEM always yields a broadened approximation to the real DOS; furthermore, because it is based on power moments, the method suffers from problems with machine precision as the moment data range over many orders of magnitude. The other method, the kernel polynomial method (KPM), uses Chebyshev moments generated from a set of orthogonal polynomials to approximate the LDOS [7,8]. The Chebyshev moment data are modulated by Gibbs damping factors to reduce unphysical oscillations in the resulting LDOS. In general, the KPM is more convenient and preferable for total energy calculations while the MEM gives better energy resolution for computing fine structure in the DOS. Extensive work has been done over the last decade to compare these methods, and to combine the best features of both in order to more accurately compute electronic structure [8].

Generating the moment data is the most costly step in implementing a real-space, statistical tight-binding calculation that is $O(N)$. The power moments, for example, amount

*Corresponding author. Email address: htj@uiuc.edu to the trace of the matrix generated by taking powers of the Hamiltonian. This operation can be constructed in physical space by delineating and summing all closed random walks from a particular atom that move around the lattice collecting products of the hopping integrals in the Hamiltonian. Thus, without exploiting any symmetry in the lattice or sparsity in the Hamiltonian, the brute force cost of collecting moments would scale exponentially with the number of moments, or $O(NC^M)$, where C is a constant $\geqslant 1$ and M is the number of moments calculated. To obtain acceptable accuracy in the DOS often requires a moment expansion that includes more than 1000 moments, so while the overall method can be $O(N)$, the cost per atom is prohibitive. However, in addition to symmetry and sparsity, information theoretic approaches make it possible to greatly reduce the cost of delineating random walks on lattices. For example, instead of the so-called brute force approach of assembling moments from random walks on the lattice, powers of the Hamiltonian can be generated directly in a graph-theoretic approach. However, this approach scales as $O(N^2M)$ without exploiting any symmetry or sparsity in the system; if a truncated subspace is used where the subspace is chosen to be large enough to support the full calculation of the moments, the cost can be reduced to $O(NM^2)$. Most information theoretic approaches involve iterative convolution; using discrete fast Fourier transforms (DFFT's) on lattices it is possible to reduce the scaling to $O(NM \log M)$ depending on the system dimensionality. In the work presented here, graph-theoretic and convolution methods including DFFT approaches are analyzed in a unified framework for a variety of problems, with specific examples given for deformed carbon nanotubes.

Carbon nanotubes are of interest for many applications because of their high mechanical strength and stiffness and desirable electrical properties [9]. Perhaps more importantly, carbon nanotubes have strongly coupled mechanical and electrical properties, and thus may be useful in applications where strain-tunable electrical properties are needed. From a modeling perspective, it is important to use methods that consider both mechanical and electronic degrees of freedom in a fully coupled way. Carbon nanotubes are part of the class of materials known as fullerenes, which have been examined before as a case study for moment-based $O(N)$ tight-binding calculations [4]. The emphasis of the work presented here is on using the method of moments to study the effects of deformation on electronic properties; comparisons are made to Green's function methods (GFM's), and both the KPM and MEM approaches are used to reconstruct the DOS for a range of uniformly and nonuniformly deformed structures.

In the following section the basic orthogonal tight-binding method of interest is outlined. Then the computational approach for determining the DOS and total energy using the moment method is introduced, and both the KPM and MEM are briefly described. Then a formal analysis of various methods for collecting moment data is presented, and several fast algorithms for collecting moment data are described in detail. Most of the methods are convolution-based approaches that are generalized versions of the recursion relation used most often in the literature to generate Chebyshev moments [8]. It is shown that within this class of methods it is also possible to build up large numbers of power moments in a computationally efficient way. Application of the moment algorithms for both uniform and nonuniform lattices is discussed. To illustrate these approaches, the effects of uniform and nonuniform deformation on the electrical properties of single wall carbon nanotubes are then computed.

II. COMPUTATIONAL FRAMEWORK FOR MOMENT-BASED TIGHT BINDING

A. Tight-binding framework

A tight-binding atomistic modeling method assumes that the total energy of the system is divided into separate parts associated with (i) repulsive interatomic interactions and (ii) the attractions due to bonding valence electron interactions, so that

$$E_{\text{tot}} = E_{\text{rep}} + E_{\text{elec}}, \tag{1}$$

where E_{rep} is a short range energy associated with repulsion between nuclei as well as other interelectron repulsive effects, and the electronic portion of the energy, E_{elec}, contains basic information regarding electronic properties.

The repulsive energy E_{rep} is generally given by a pair potential or an environment-dependent pair functional. Details of the repulsive term used in the analysis of carbon nanotubes are given in the last section of this paper. The electronic portion of the energy, E_{elec}, is defined as

$$E_{\text{elec}} = 2\int_{-\infty}^{E_f} ED(E)dE, \tag{2}$$

where $D(E)$ is the DOS and E_f is the Fermi energy. The DOS is considered to capture the basic electronic properties of the system. From the DOS, many other properties including the energy band gap E_g can be extracted.

The tight-binding Hamiltonian H, from which the DOS is computed, consists of diagonal elements corresponding to atomic orbital energies, and off-diagonal elements corresponding to orthogonal two-center hopping parameters. For the carbon nanotube analysis undertaken here, the sp^3 Hamiltonian is given in detail in the final section of the paper. Once the Hamiltonian H is constructed, the DOS can be calculated from the Hamiltonian by direct diagonalization. This method is costly, with computational complexity of $O(N^3)$, so for large systems it makes sense to consider methods with larger cost per atom if better scaling with the total number of atoms N is possible. This motivates the development of the method of moments.

B. Constructing the DOS from the moments

1. Power-moment-based MEM

The power moments of $D(E)$ are defined as

$$\mu_m = \text{Tr}\{H^m\} = \int E^m D(E)dE. \tag{3}$$

In Eq. (3), the trace operation may be performed within only the subsystem of interest. For example, if only the atom (or orbital) i is of interest, $\mu_m = \langle i|H^m|i\rangle$.

To generate power moments from H using Eq. (3), the calculation amounts to taking powers of the matrix H. Tight-binding Hamiltonians are usually sparse matrices, so there are ways to reduce the complexity of the calculation; this is the subject of the next section of the paper.

Once the power moments are computed the $D(E)$ can in principle be determined [5]. However, since only a finite number of moments can be generated from Eq. (3), there are an infinite number of possible forms of $D(E)$ consistent with the finite number of moments. In the maximum entropy approximation, the information theoretic entropy associated with all these possible forms of $D(E)$ is maximized to find the most likely form [5]. The entropy is a functional of $D(E)$, described by

$$S = -\int D(E)\ln D(E)dE + \sum_{m=0}^{M} \lambda_m\left(\int E^m D(E)dE - \mu_m\right), \tag{4}$$

where M is the number of moments calculated, and λ_m are Lagrange multipliers. Functional variation with respect to the unknown $D(E)$ yields

$$\frac{\delta S}{\delta D(E)} = 0 \Rightarrow D(E) = \exp\left(-\lambda_0 - \sum_{m=1}^{M} \lambda_m E^m\right) = \frac{1}{Z}\exp\left(-\sum_{m=1}^{M} \lambda_m E^m\right). \tag{5}$$

The $D(E)$ can be computed using Newton's method of iteration. Starting with initial values λ_m^0, a preliminary $D(E)$ can be found from Eq. (5). Using Eq. (3), the respective moments $\hat{\mu}_m$ can be calculated from the preliminary $D(E)$, which is then checked against moments μ_m. The iteration process continues until the error between μ_m and $\hat{\mu}_m$ falls within a certain limit. Unfortunately this process is computationally intensive, and the successive approximations to $D(E)$ are unacceptably oscillatory [5] and limited by machine precision [8]. This problem motivates the use of other methods such as the KPM described in the next section.

2. Chebyshev-moment-based KPM

The kernel polynomial method is a linear Chebyshev approximation to the real DOS based on the Chebyshev moment data. To compute Chebyshev moments of the DOS, the Hamiltonian must be rescaled according to $H=aX+b$, where the magnitude of every eigenvalue x_n of X falls between 0 and 1. Then $D(E)$ is rewritten as $D(x)$, and the Chebyshev moments are defined as [8]

$$\mu_m = \mathrm{Tr}\{T_m(X)\} = \int_{-1}^{1} T_m(x)D(x)dx, \tag{6}$$

where T_m, the Chebyshev polynomials of the first kind, are defined by $T_0(X)=1, T_1(X)=X$, and the recurrence relation

$$T_m(X) = 2XT_{m-1}(X) - T_{m-2}(X). \tag{7}$$

As before, in Eq. (6), the trace operation can be performed for a subsystem of interest.

In generating Chebyshev moments through the recursion equation (7), the computational effort comes mainly from the portion $XT_{m-1}(X)$, which is the same as the power moments. In the following section algorithms for this step are formally analyzed. It is shown that a method known as single step iterative convolution (SSIC) can be used to generate both Chebyshev and power moments using analogous recursion relations. Furthermore, for the case of carbon nanotubes, the methods are shown to scale as $O(CM^2)$, where the constant C is identified from the periodic structure of the nanotube.

The DOS can be exactly constructed from the Chebyshev moments according to

$$D(x) = \frac{1}{\pi\sqrt{1-x^2}}\left[\mu_0 + 2\sum_{m=1}^{\infty} \mu_m T_m(x)\right]. \tag{8}$$

However, since only a finite number of moments is available, a factor g_m^M is introduced to damp the Gibbs phenomenon. The available data $\hat{\mu}_m$ are substituted for the moments in Eq. (8), and the DOS is written as [8]

$$D_K(x) = \frac{1}{\pi\sqrt{1-x^2}}\left[1 + 2\sum_{m=1}^{M} \hat{\mu}_m g_m^M T_m(x)\right], \tag{9}$$

with

$$g_m^M = \sum_{v=0}^{M-m} a_v a_{v+m}, \tag{10}$$

$$a_v = \frac{U_v(\lambda)}{\sqrt{\sum_{v=0}^{M} U_v^2(\lambda)}}. \tag{11}$$

Here $U_v(\lambda)$ are the Chebyshev polynomials of the second kind.

3. MEM combined with KPM

To obtain the best approximation to $D(E)$, the MEM can be used in combination with the same Chebyshev moments used in the KPM. In fact, the Chebyshev moments are Fourier integrals

$$\mu_m = \int_{-1}^{1} T_m(x)D(x)dx = \int_{0}^{\pi} \cos(m\phi)D(\phi)d\phi, \tag{12}$$

where $x\equiv\cos(\phi), T_m(x)=\cos(m\phi), D(x)=D(\phi)/\sin(\phi)$. So the $D_K(\phi)$ can be represented as

$$D_K(\phi) = \int_{0}^{2\pi} \delta_K(\phi-\phi_0)D(\phi_0)d\phi_0, \tag{13}$$

$$\delta_K(\phi) = \frac{1}{2\pi}\left[g_0 + 2\sum_{m=1}^{M} g_m^M \cos(m\phi)\right]. \tag{14}$$

To improve the energy resolution of the DOS a kernel polynomial approximation for $M\times K$ moments is sought based on the available M moments [8], where K is some integer. To avoid arbitrary extrapolation of the moment series, the MEM can be used here to provide the criterion. In the method described by Silver and co-workers [8], the relative entropy given by

$$S = \int_0^{\pi} \left[D(\phi) - D_0(\phi) - D(\phi)\ln\left(\frac{D(\phi)}{D_0(\phi)}\right)\right] d\phi \qquad (15)$$

is maximized. The iteration method is adopted which starts from the initial DOS $D_0(\phi)$, and $D_0(\phi)$ is the $D_K(\phi)$ from the KPM in Eq. (13) with M moments.

Finally, once the DOS is known, many other electronic properties can be extracted. First of all, given the number of valence electrons,

$$N_{\text{val}} = 2\int_{-\infty}^{E_f} D(E)dE, \qquad (16)$$

the Fermi energy E_f can be calculated. Then the electronic portion of energy E_{elec} is computed from Eq. (2). From the DOS near E_f, it is straightforward to estimate the energy band gap E_g. In the following section, a detailed analysis of fast methods for determining moments is presented. Then in the last section it is shown that with the KPM alone it is difficult to accurately evaluate E_g for carbon nanotubes. Even with fast methods for evaluating moments, the huge amount of data needed leads to impractical computational cost. Thus, the combined KPM and MEM approach is demonstrated for both uniformly and nonuniformly deformed single wall carbon nanotubes.

III. CALCULATING MOMENT DATA

A. Notation for random walk distributions on an integer lattice

For an ensemble of atoms with an arbitrary number of orbitals per atom, the process of collecting moment data to determine the LDOS at a particular atom is identical to the process of summing the costs of all possible closed weighted random walks on a lattice, where the total cost of each random walk is simply the product of individual hopping parameters associated with steps in the walk. This process is formally analyzed here within the framework of efficient random-walk statistics to introduce the use of convolution in determining moments of the DOS.

Given a neighbor list and an ensemble of nodes, or atom and orbital pairs, a random step is defined as a move from one node to a randomly selected neighbor. A random walk is a sequence of random steps starting from some initial node. A random walk of m steps is also called a random walk of length m.

Node positions on the integer lattice are defined by the coordinate vector $\boldsymbol{x}=[x_1,\ldots,x_d]$ and the origin node is defined by $\boldsymbol{o}=[0,\ldots,0]$. For any pair of connected nodes, or neighbors, the associated cost of hopping from one node to the other is written as $c(\boldsymbol{a},\boldsymbol{b})$. The cost of a self-hop, or the site energy in tight binding, is given by $c(\boldsymbol{a},\boldsymbol{a})$.

The value of a random walk is defined as the product of the costs of each hop. For the case of a random walk of length m starting at node $\boldsymbol{a}$ and ending at node $\boldsymbol{x}$, the scalar valued walk function is written as $w_m[\boldsymbol{a};\boldsymbol{x}]$. In general, $\boldsymbol{x}$ can be left as a variable so that the function $w_m[\boldsymbol{a};\boldsymbol{x}]$ is evaluated over all possible values of $\boldsymbol{x}=[x_1,\ldots,x_d]$ and each value is then placed in the $[x_1,\ldots,x_d]$ entry of the d-dimensional walk distribution array $W_m[\boldsymbol{a};\boldsymbol{x}]$, where d is the spatial dimensionality of the lattice. The lattice node coordinate $\boldsymbol{x}$ is used as both the argument for w_m and also the entry index specifying the corresponding location in W_m. When the context is clear, W_m is used as a shorthand for $W_m[\boldsymbol{o};\boldsymbol{x}]$.

Although walk distributions are defined over the entire lattice structure, it is often convenient to work with the finite subset of the walk distribution that contains all the nonzero information. In each x_i coordinate dimension of W_m, for finite values of m, there is an index range, or minimal bounding box, that captures all nonzero entries. If the index range in the ith dimension is denoted by r_i, then the minimal bounding box is the portion of W_m defined by $r_1\times\cdots\times r_d$. The dimensions of the minimal bounding box can be written as the vector

$$\dim(W_m) = [|r_1|,\ldots,|r_d|] \qquad (17)$$

where $|r_i|$ is used to denote the magnitude of the index range in dimension i. The minimal bounding box of W_m may still contain zero valued entries, but it is the smallest region with continuous index ranges in each dimension that contains all nonzero entries of W_m. For example, on an unconstrained, homogeneous, integer lattice, with starting node $\boldsymbol{a}=[a_1,\ldots,a_d]$

$$r_i = \{a_i - m, a_i + m\} \quad \text{for all } i=\{1,\ldots,d\}, \qquad (18)$$

$$\dim(W_m) = [2m+1,\ldots,2m+1], \qquad (19)$$

and there are $(2m+1)^d$ nodes in the minimal bounding box.

B. Methods for collecting random-walk distributions

In order to compare the various methods for collecting random-walk distributions, or moment data, the homogeneous integer lattice is considered first. In this case the cost of hopping between a pair of neighboring nodes is the same across the entire lattice structure. Thus, for simplicity, the value of each random walk does not need to be calculated one step at a time because all random walks of length m will have a value of c^m, where c is the cost of each step. The methods developed for enumerating random-walk distributions on this simple lattice will serve as the basis for approaching more complex structures. A brute force, recursive algorithm is presented first, to establish a reference. Second, the graph-theoretic approach is given, which leads to a set of convolution-based techniques.

1. The brute force approach

This method is implemented recursively by starting from a given node and calling a function from each of the node's neighbors with the walk length reduced by 1 after each round of function calls. In the d-dimensional lattice, there are $2d$ possible neighbors for each step. The recursive function thus makes one function call to get started and then $2d$ more calls (one from each node neighbor) for the remaining steps: $2,\ldots,m$. This is a total of $1+(2d)^{m-1}$ function calls, making the algorithm of complexity $\mathcal{O}\boldsymbol{(}(2d)^{m-1}\boldsymbol{)}$.

2. The graph-theoretic approach

Within the minimal bounding box of W_m, nodes $\boldsymbol{a}_i$ are indexed by $i\in\{1,\ldots,N\}$. The $N\times N$ adjacency matrix A is defined by

$$A_{i,j}=c(\boldsymbol{a}_i,\boldsymbol{a}_j). \tag{20}$$

In an orthogonal tight-binding formulation, the adjacency matrix is the Hamiltonian. Matrices of this form have the property that for

$$A^m=\underbrace{A\cdots A}_{m\text{ times}} \tag{21}$$

the i,j entry of A^m is the sum of the values of all walks of length m that connect nodes $\boldsymbol{a}_i$ and $\boldsymbol{a}_j$. That is,

$$[A^m]_{i,j}=w_m[\boldsymbol{a}_i;\boldsymbol{a}_j]. \tag{22}$$

Thus, all the entries of A^m across row i (or down column i) can be used to fill in the entries of $W_m[\boldsymbol{a}_i;\boldsymbol{x}]$. The number of closed random walks for each lattice node can then be determined by reading the entries off the diagonal of A^m. That is, the moments, or sums of closed random-walk costs, can be obtained simply by taking powers of the Hamiltonian.

Given a node numbering system, this method can easily be applied to inhomogeneous lattice structures with spatially varying geometries. The matrix A^m has more information than just random-walk distributions from a single preselected starting node; it contains the random-walk distributions from *any* starting node. This additional information can be used to calculate quantities such as the number of random walks between nodes on opposite sides of an inhomogeneity in the lattice.

3. Derivation of iterative convolution from the graph-theoretic method

By selecting the row of A^{m-1} that corresponds to node $\boldsymbol{a}$, and the column of A that corresponds to any node $\boldsymbol{x}$, Eq. (22) can be used to express the matrix multiplication of Eq. (21) as

$$w_m[\boldsymbol{a};\boldsymbol{x}]=\sum_{\boldsymbol{b}}w_{m-1}[\boldsymbol{a};\boldsymbol{b}]w_1[\boldsymbol{b};\boldsymbol{x}] \tag{23}$$

where the d-dimensional summation over $\boldsymbol{b}$ covers every node in the lattice (this is equivalently done by just summing over the nodes in the minimal bounding box of $W_{m-1}[\boldsymbol{a};\boldsymbol{x}]$). The equation considers all walks of length $m-1$ from $\boldsymbol{a}$ to all possible intermediate nodes $\boldsymbol{b}$, and then considers how to hop from each one of these locations to the desired ending node $\boldsymbol{x}$. This covers all possible ways to walk from $\boldsymbol{a}$ to $\boldsymbol{x}$ in m steps.

In the case of lattice homogeneity, $c(\boldsymbol{a},\boldsymbol{b})=c(\boldsymbol{o},\boldsymbol{b}-\boldsymbol{a})$, and thus $w_1[\boldsymbol{a};\boldsymbol{b}]=w_1[\boldsymbol{o};\boldsymbol{b}-\boldsymbol{a}]$. From this relation Eq. (23) can be rewritten as

$$w_m[\boldsymbol{a};\boldsymbol{x}]=\sum_{\boldsymbol{b}}w_{m-1}[\boldsymbol{a};\boldsymbol{b}]w_1[\boldsymbol{o};\boldsymbol{x}-\boldsymbol{b}]. \tag{24}$$

By considering all possible values of $\boldsymbol{x}$, the scalar equation takes the array form

$$W_m[\boldsymbol{a};\boldsymbol{x}]=\sum_{\boldsymbol{b}}w_{m-1}[\boldsymbol{a};\boldsymbol{b}]W_1[\boldsymbol{o};\boldsymbol{x}-\boldsymbol{b}] \tag{25}$$

which is readily identified as a convolution "in $\boldsymbol{x}$," because the starting nodes of the quantities involved are constants and the ending node $\boldsymbol{x}$ is experiencing all possible shifts by $\boldsymbol{b}$. The weighting coefficients $w_{m-1}[\boldsymbol{a};\boldsymbol{b}]$ are scalars and the shifted masks $W_1[\boldsymbol{o};\boldsymbol{x}-\boldsymbol{b}]$ are d-dimensional arrays; the equation sweeps the one-step masks over the values of the previous distribution. By the definition of convolution this equation is written as

$$W_m[\boldsymbol{a};\boldsymbol{x}]=W_{m-1}[\boldsymbol{a};\boldsymbol{x}]*W_1[\boldsymbol{o};\boldsymbol{x}]. \tag{26}$$

The most general form of the convolution of two walk distributions $W_m[\boldsymbol{a};\boldsymbol{x}]$ and $W_n[\boldsymbol{b};\boldsymbol{x}]$ on a homogeneous, d-dimensional, integer lattice is

$$\begin{aligned}W_m[\boldsymbol{a};\boldsymbol{x}]*W_n[\boldsymbol{b};\boldsymbol{x}]&=W_{m+n}[\boldsymbol{a}+\boldsymbol{b};\boldsymbol{x}]\\&\triangleq\sum_{\boldsymbol{\Delta}=-\infty}^{\infty}w_m[\boldsymbol{a};\boldsymbol{\Delta}]W_n[\boldsymbol{b};\boldsymbol{x}-\boldsymbol{\Delta}]\end{aligned} \tag{27}$$

where $\boldsymbol{\Delta}=[\Delta_1,\ldots,\Delta_d]$ and the summation is d dimensional. The homogeneity of the lattice is what allows the starting nodes $\boldsymbol{a}$ and $\boldsymbol{b}$ to be combined into $(\boldsymbol{a}+\boldsymbol{b})$. If the minimal bounding box dimensions of the involved walk distributions are

$$\dim(W_m)=[p_1,\ldots,p_d], \tag{28}$$

$$\dim(W_n)=[q_1,\ldots,q_d], \tag{29}$$

then the resulting walk distribution is of dimension

$$\dim(W_{m+n})=[p_1+q_1-1,\ldots,p_d+q_d-1]. \tag{30}$$

The complexity of convolution between two distribution arrays W_m and W_n on a d-dimensional integer lattice when computed using Eq. (27) is

$$\begin{aligned}\text{complexity }\{W_m*W_n\}\triangleq(m,n)_d&=(2m+1)^d(2n+1)^d\\&\approx O(m^dn^d).\end{aligned} \tag{31}$$

Convolution can also be implemented using either of the following Fourier transform relations:

$$W_{m+n}=\mathcal{F}^{-1}(\mathcal{F}(\overline{W}_1)^{m+n}), \tag{32}$$

$$W_{m+n}=\mathcal{F}^{-1}(\mathcal{F}(\overline{W}_m)\mathcal{F}(\overline{W}_n)), \tag{33}$$

where a bar over a walk distribution indicates that the array has been padded with zeros so that its dimensions are the same as the resulting array $\overline{W}_{m+n}$, the power on $\overline{W}_1$ implies exponentiating each term of $\overline{W}_1$ individually, and the product of the transforms is taken element by element (and not by matrix multiplication). These transforms can be implemented using the discrete fast Fourier transform for an even greater performance boost.

In the context of random-walk enumerations for computing moments, all distributions with lengths from 1 to m are of interest, and not just the distribution of walks of length m. While the DFFT implementation of convolution achieves better performance when pairs of large matrices are convolved, for example in the calculation of $W_m * W_m$, the calculation of $W_1 * W_m$ is more useful here because it corresponds to building up a distribution one step at a time, and it can be used iteratively to yield all the intermediate distributions. According to the definition of convolution, the complexities of $W_1 * W_m$ and $W_m * W_m$ on a two-dimensional integer lattice are $O(m^2)$ and $O(m^4)$, respectively, but with the DFFT method, they are reduced to complexity $O(m\log m)$ and $O(m^2\log m)$.

4. Single step iterative convolution

Based on the discussion in the previous section we introduce the convolution-based method referred to as single step iterative convolution. The distribution of a single step, W_1, is first identified. Then the distribution W_m is calculated using the iterative convolution according to

$$W_m[\boldsymbol{o};\boldsymbol{x}] = W_{m-1}[\boldsymbol{o};\boldsymbol{x}] * W_1[\boldsymbol{o};\boldsymbol{x}]. \tag{34}$$

This process involves $m-1$ convolutions, and has computational complexity

$$\mathrm{SSIC}(m) = (1,1)_d + (2,1)_d + (3,1)_d + \cdots + (m-1,1)_d \tag{35}$$

$$\approx O(1^d \times 1^d + 2^d \times 1^d + \cdots + (m-1)^d \times 1^d) \tag{36}$$

$$= O(1^d + 2^d + \cdots + (m-1)^d) \tag{37}$$

$$= \begin{cases} O\left(\dfrac{m(m-1)}{2}\right), & d=1, \\ O\left(\dfrac{m(m-1)(2m-1)}{6}\right), & d=2, \\ O\left(\left(\dfrac{m(m-1)}{2}\right)^2\right), & d=3, \end{cases} \tag{38}$$

$$\approx \begin{cases} O(m^2), & d=1, \\ O(m^3), & d=2, \\ O(m^4), & d=3. \end{cases} \tag{39}$$

5. Smallest prime iterative convolution

An alternative convolution-based method can be constructed by first determining p, the smallest prime factor of m. SSIC is used to generate the walk distribution W_p, which is then convolved with itself $m/p-1$ times to get the desired distribution W_m. The computational complexity of this method, referred to as the smallest prime iterative convolution (SPIC), is given by

$$\mathrm{SPIC}(m) = \mathrm{SSIC}(p) + (p,p)_d + (2p,p)_d + (3p,p)_d + \cdots + [(m/p-1)p,p]_d \tag{40}$$

$$\approx p^{d+1} + p^d p^d + (2p)^d p^d + (3p)^d p^d + \cdots + (m/p-1)^d p^d p^d \tag{41}$$

$$= p^{d+1} + p^{2d}[1^d + 2^d + 3^d + \cdots + (m/p-1)^d] \tag{42}$$

$$= \begin{cases} O\left(p^2 + p^2\left(\dfrac{(m/p-1)(m/p)}{2}\right)\right), & d=1, \\ O\left(p^3 + p^4\left(\dfrac{(m/p-1)(m/p)(2m/p-1)}{6}\right)\right), & d=2, \\ O\left(p^4 + p^6\left(\dfrac{(m/p-1)(m/p)}{2}\right)^2\right), & d=3, \end{cases} \tag{43}$$

$$\approx \begin{cases} O(m^2), & d=1, \\ O(pm^3), & d=2, \\ O(p^2m^4), & d=3. \end{cases} \tag{44}$$

The p coefficients have been left in the complexity expressions to show how strongly they may impact the computations. Even though the m dependence of these expressions is the same as in SSIC, SSIC performs more efficiently, especially in cases where p is large.

6. Binary iterative convolution

The desired walk length m can be written in a normalized $(a_n \neq 0)$ binary form as

$$m = \sum_{i=0}^{n} a_i \times 2^i = (a_n, \ldots, a_0)_2 \tag{45}$$

where $n = \lceil \log_2(m) \rceil$. The set of walk distributions $\{W_{2^0}, \ldots, W_{2^n}\}$ is then generated by the recursive definition

$$W_{2^k}[\boldsymbol{o};\boldsymbol{x}] = W_{2^{k-1}}[\boldsymbol{o};\boldsymbol{x}] * W_{2^{k-1}}[\boldsymbol{o};\boldsymbol{x}] \tag{46}$$

where each walk distribution has the associated complexity

$$W_2 = W_1 * W_1 \rightarrow (1,1)_d, \tag{47}$$

$$W_4 = W_2 * W_2 \rightarrow (2,2)_d, \tag{48}$$

$$W_8 = W_4 * W_4 \rightarrow (4,4)_d, \tag{49}$$

$$\vdots$$

$$W_{2^n} = W_{2^{n-1}} * W_{2^{n-1}} \rightarrow (2^{n-1}, 2^{n-1})_d. \tag{50}$$

The total complexity for calculating this set is

$$(1,1)_d + (2,2)_d + (4,4)_d + \cdots + (2^{n-1},2^{n-1})_d \approx O(1^{2d} + 2^{2d} + 4^{2d} + \cdots + (2^{n-1})^{2d}) \tag{51}$$

$$= O(2^0 + 2^{2d} + 2^{4d} + 2^{6d} + \cdots + 2^{2d(n-1)}) \tag{52}$$

TABLE I. A summary of the computational complexities for the integer lattice random-walk enumeration methods on one, two, and three dimensions.

	Order of complexity		
Method	$d=1$	$d=2$	$d=3$
DFFT	$O(m \log m)$	$O(m^2 \log m)$	$O(m^3 \log m)$
SSIC	$O(m^2)$	$O(m^3)$	$O(m^4)$
SPIC	$O(m^2)$	$O(pm^3)$	$O(p^2m^4)$
BIC	$O(m^2)$	$O(m^4)$	$O(m^6)$
Graph	$O(m^3)$	$O(m^6)$	$O(m^9)$
Brute force	$O(2^m)$	$O(4^m)$	$O(6^m)$

$$=O\left(\sum_{k=0}^{n-1} 2^{2dk}\right) \tag{53}$$

$$=O\left(\sum_{k=1}^{n} (2^{2d})^{k-1}\right) \tag{54}$$

$$=O\left(\frac{(2^{2d})^n - 1}{2^{2d} - 1}\right) \tag{55}$$

$$\approx O(2^{2d(n-1)}) \tag{56}$$

$$=O\left(\left(\frac{m}{2}\right)^{2d}\right). \tag{57}$$

The desired walk distribution W_m is then constructed by convolving the distributions from the above set which have a nonzero digit in the binary representation of m. In the worst case all bits are a “1.” This requires

$$W_1 * W_2 * W_4 * \cdots * W_{2^{n-1}} = (1,2)_d + (3,4)_d + (7,8)_d + \cdots + (2^{n-1}-1,2^{n-1})_d \tag{58}$$

$$< (2,2)_d + (4,4)_d + (8,8)_d + \cdots + (2^{n-1},2^{n-1})_d. \tag{59}$$

This is less complex than the above computations which are required to generate the set $W_2, W_4, \ldots, W_{2^n}$. The overall performance of this binary iterative convolution (BIC) thus scales as $O\big((m/2)^{2d}\big)$.

7. Comparative analysis

A summary of the computational complexities for the walk distribution enumeration techniques in one, two, and three dimensions is given in Table I. The brute force method (very slow), the SPIC method, and the BIC method do not produce all intermediate distributions and are slower than the SSIC method, which can generate the desired ensemble. The DFFT method has the lowest order of complexity for generating a random walk of a specific length, but it is actually slower than the SSIC when the full ensemble of random walks or moment distributions is required. The SSIC method is faster than the graph-theoretic method so it is the method of choice for homogeneous lattice regions. The graph-theoretic method is better suited for handling inhomogeneous lattice structures that cannot be uniformly handled by convolution. In the context of compiling moments for electronic structure calculations, the SSIC, for power moments, is equivalent to recursive methods for generating Chebyshev moments. The graph-theoretic method describes the basic approach of forming powers of the Hamiltonian. The DFFT method described here represents a type of approach that has not been used previously to our knowledge in computing moments for electronic structure applications.

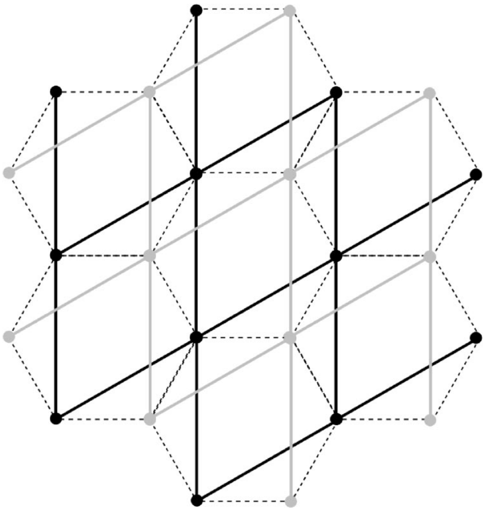

FIG. 1. Representation of the hexagonal graphene lattice as two superimposed integer lattices.

C. Applying convolution methods to carbon nanotubes

1. Decomposition of the homogeneous, planar, hexagonal lattice

The hexagonal graphene lattice can be decomposed into two triangular lattice layers. The two triangular lattice layers can be viewed as integer lattices, as shown in Fig. 1, allowing for use of the analysis from the previous sections.

The direction and magnitude of a single edge in each of the integer lattices can be described by the lattice basis vectors $\boldsymbol{b}_x$ and $\boldsymbol{b}_y$, respectively. The offset between the lattices can then be written as $\boldsymbol{b}_s$. These quantities are shown in Fig. 2.

Any point on the hexagonal lattice is represented by the coordinate

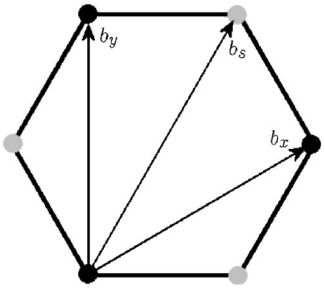

FIG. 2. Relative positions of the basis vectors.

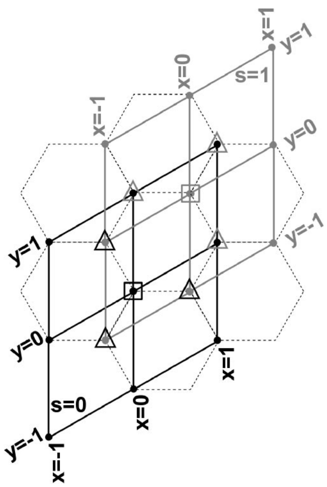

FIG. 3. A portion of the hexagonal lattice showing the distribution of one-step walks from the two possible origin nodes. The lattice coordinates x and y are labeled along the edges and the lattice coordinate s is given in a corner of each layer. Black is used for layer 0 and gray is used for layer 1.

$$\boldsymbol{x}=[x,y,s]\in\mathbb{Z}^2\times\{0,1\} \tag{60}$$

where s denotes the layer (either 0 or 1) and x and y are the lattice coordinates of the node on its layer. An arbitrary node on layer s is referred to as $\boldsymbol{x}_s$ and the origin of this lattice is defined as $\boldsymbol{o}_s=[0,0,s]$. The corresponding Cartesian location of any lattice point with respect to $\boldsymbol{o}_0=[0,0,0]$ and the right-handed reference frame where $\boldsymbol{b}_y$ is in the y direction is

$$\overline{\boldsymbol{x}}=x\boldsymbol{b}_x+y\boldsymbol{b}_y+s\boldsymbol{b}_s\in\mathbb{R}^2. \tag{61}$$

The basis vectors corresponding to a hexagonal lattice with unit side length are

$$\boldsymbol{b}_x=\begin{pmatrix}3/2\\ \sqrt{3}/2\end{pmatrix},\quad \boldsymbol{b}_y=\begin{pmatrix}0\\ \sqrt{3}\end{pmatrix},\quad \boldsymbol{b}_s=\begin{pmatrix}1\\ \sqrt{3}\end{pmatrix}. \tag{62}$$

2. Hexagonal lattice convolution

The SSIC method is the most suitable approach for computing random-walk distributions on the integer lattice in two and three dimensions, so hexagonal lattice convolution will therefore be based on this approach. The notation $W_m[0,0;x,y]$, which is the distribution of random walks of length m starting at the origin on a two-dimensional integer lattice, is now augmented to the form $W_m[0,0,t;x,y,s]\triangleq W_m[\boldsymbol{o}_t;\boldsymbol{x}_s]$, which is the distribution of random walks of length m on layer s when the origin is taken as $\boldsymbol{o}_t$. The noncentrosymmetry of the hexagonal lattice requires that both possible origin locations be considered.

The distribution of one-step random walks must first be enumerated for hexagonal lattice convolution, as it was in the SSIC method on the integer lattice. Figure 3 shows the region of hexagonal lattice defined by $\{-1,1\}\times\{-1,1\}\times\{0,1\}$. Here the origin on layer 0, from which one-step walks are marked with black triangles, is the black node in the black square; the origin on layer 1, from which one-step walks are marked with gray triangles, is the gray node in the gray square. All single steps from the layer 0 origin end on layer 1, and single steps from the layer 1 origin end on layer 0. Thus $W_1[\boldsymbol{o}_0;\boldsymbol{x}_0]$ and $W_1[\boldsymbol{o}_1;\boldsymbol{x}_1]$ are both zero distributions.

The results are written as the walk distributions

$$W_1[\boldsymbol{o}_0;\boldsymbol{x}_0]=\begin{bmatrix}0&0&0\\0&0&0\\0&0&0\end{bmatrix},\quad W_1[\boldsymbol{o}_0;\boldsymbol{x}_1]=\begin{bmatrix}0&0&0\\1&0&0\\1&1&0\end{bmatrix},$$
$$W_1[\boldsymbol{o}_1;\boldsymbol{x}_0]=\begin{bmatrix}0&1&1\\0&0&1\\0&0&0\end{bmatrix},\quad W_1[\boldsymbol{o}_1;\boldsymbol{x}_1]=\begin{bmatrix}0&0&0\\0&0&0\\0&0&0\end{bmatrix}. \tag{63}$$

From these basic definitions a hexagonal lattice SSIC method can be defined in the same way as for the integer lattice. The random-walk distribution after $m+1$ steps is obtained by taking the distribution after m steps and convolving on the one-step distribution as defined in Eq. (63). The resulting hexagonal convolution is thus given by the equation

$$W_{m+1}[\boldsymbol{o}_t;\boldsymbol{x}_s]=\sum_{u=0}^{1}W_m[\boldsymbol{o}_t;\boldsymbol{x}_u]*W_1[\boldsymbol{o}_u;\boldsymbol{x}_s]. \tag{64}$$

This equation is evaluated once for $s=0$ and once for $s=1$ to get the walk distribution.

3. Convolution on a four-orbital, two-dimensional lattice

Random walks or moment distributions can also be collected using the same methods for the general case in which each atom or node has a four-orbital sp^3 basis more representative of the real electronic structure of carbon. This can be accomplished simply by adding one more integer valued parameter to the coordinate system. Any node on the integer lattice with four orbitals denoted by $i=\{0,1,2,3\}$ can then be represented by the coordinate

$$\boldsymbol{x}=[x,y,i]\in\mathbb{Z}^2\times\{0,1,2,3\}. \tag{65}$$

Level i of this arbitrary node is referred to as $\boldsymbol{x}_i$ and the origin of this lattice is defined as $\boldsymbol{o}_i=[0,0,i]$.

For any pair of atoms $\boldsymbol{a}$ and $\boldsymbol{b}$, the total set of interactions is represented by the cost function $c(\boldsymbol{a}_i,\boldsymbol{b}_j)$, which describes the cost of hopping from $\boldsymbol{a}_i$ to $\boldsymbol{b}_j$. In a tight-binding model this information is stored in the 8×8 Hamiltonian (or cost) matrix

$$H(\boldsymbol{a},\boldsymbol{b})=\begin{bmatrix}H^{\boldsymbol{a}\to\boldsymbol{a}}&H^{\boldsymbol{a}\to\boldsymbol{b}}\\H^{\boldsymbol{b}\to\boldsymbol{a}}&H^{\boldsymbol{b}\to\boldsymbol{b}}\end{bmatrix} \tag{66}$$

where each subblock in $H(\boldsymbol{a},\boldsymbol{b})$ is a 4×4 matrix whose (i,j) entry is the hop cost of moving from orbital state i to orbital state j for the corresponding atom pair given in the superscript. The entries in each subblock map to the cost function according to

$$[H^{\boldsymbol{a}\to\boldsymbol{a}}]_{i,j}=c(\boldsymbol{a}_i,\boldsymbol{a}_j),\quad [H^{\boldsymbol{a}\to\boldsymbol{b}}]_{i,j}=c(\boldsymbol{a}_i,\boldsymbol{b}_j), \tag{67}$$

$$[H^{\boldsymbol{b}\to\boldsymbol{a}}]_{i,j}=c(\boldsymbol{b}_i,\boldsymbol{a}_j),\quad [H^{\boldsymbol{b}\to\boldsymbol{b}}]_{i,j}=c(\boldsymbol{b}_i,\boldsymbol{b}_j). \tag{68}$$

The tight-binding Hamiltonian $H(\boldsymbol{a},\boldsymbol{b})$ can be completely general in form. Since interactions are reciprocal (each edge has the same hop cost for moving in both directions) $H(\boldsymbol{a},\boldsymbol{b})$ is symmetric. If all atoms are internally the same, then the self-energies are equal, or $H^{\boldsymbol{a}\to\boldsymbol{a}}=H^{\boldsymbol{b}\to\boldsymbol{b}}$.

Finally, combining the two-layer decomposition of the hexagonal lattice ($s=\{0,1\}$) with the four-orbital model ($i=\{0,1,2,3\}$) produces the node coordinate system

$$\boldsymbol{x}=[x,y,s,i]\in\mathbb{Z}^2\times\{0,1\}\times\{0,1,2,3\} \qquad (69)$$

for the four-orbital, two-dimensional, hexagonal lattice. In shorthand notation, such an arbitrary node is written as $\boldsymbol{x}_{s,i}$ and the origin is written as $\boldsymbol{o}_{s,i}=[0,0,s,i]$.

The set of 64 single hop masks is given by

$$W_1[\boldsymbol{o}_{t,j};\boldsymbol{x}_{s,i}]=\begin{bmatrix} 0 & c(\boldsymbol{o}_{t,j},\boldsymbol{a}_{s,i}) & 0 \\ c(\boldsymbol{o}_{t,j},\boldsymbol{d}_{s,i}) & c(\boldsymbol{o}_{t,j},\boldsymbol{o}_{s,i}) & c(\boldsymbol{o}_{t,j},\boldsymbol{b}_{s,i}) \\ 0 & c(\boldsymbol{o}_{t,j},\boldsymbol{c}_{s,i}) & 0 \end{bmatrix}. \qquad (70)$$

The set of eight iterative convolution equations is

$$W_{m+1}[\boldsymbol{o}_{t,j};\boldsymbol{x}_{s,i}]=\sum_{u=0}^{1}\sum_{k=0}^{3}W_m[\boldsymbol{o}_{t,j};\boldsymbol{x}_{u,k}]*W_1[\boldsymbol{o}_{u,k};\boldsymbol{x}_{s,i}]. \qquad (71)$$

This equation sums over all lattice layers and orbital levels of the intermediate distribution W_m, and then uses the single hop masks and convolution to determine the contributions made to the distribution for any lattice layer s and orbital layer i in the new walk distribution W_{m+1}. Using this method and notation, random walks for power moments (or Chebyshev moments) can be efficiently collected on the graphene lattice.

4. Handling chirality, vacancies, and interfaces

Hexagonal convolution as described in the previous section can be used to enumerate walks on a single walled carbon nanotube, which is topologically equivalent to a rectangular section of hexagonal lattice with one set of opposite edges connected. A periodic boundary condition is implemented for any walks that go beyond the connected opposite edges, while any walks that go beyond the nonconnected opposite edges, or ends of the nanotube, are ignored. This implementation is also easily extended to include chirality. When walks go beyond the opposite connected edges, they are relocated onto the other side, but with a shift in lattice coordinate along the nanotube axis equal to the chirality. Thus any such structure can be defined by size and chirality parameters.

Vacancies can also be handled by defining the set of one-step hops (which is also W_1) as a function of position. By allowing the vacant nodes to be temporarily present (thus permitting use of the convolution method) and then removing the incorrect additional walks that have been allowed on the lattice, the homogeneity in the lattice is preserved for the sake of computational performance. This is achieved by zeroing out the entries in W_m that correspond to vacant nodes after each iteration of convolution. As an example the integer lattice is considered with a set of vacant nodes defined by

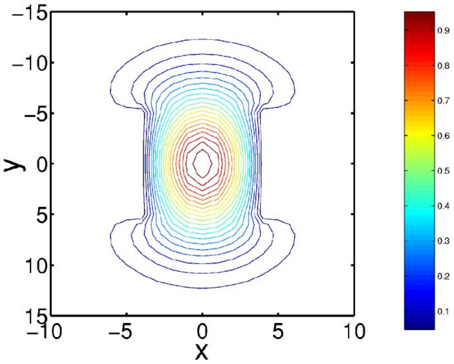

FIG. 4. (Color online) Walk distribution after 50 steps on the integer lattice with node vacancies as defined in Eq. (72).

$$\mathcal{V}=\{[x,y]:x=\pm4 \quad \text{and} \quad |y|\le5\}. \qquad (72)$$

This set effectively restricts possible walks from the origin to positions left and right. As the walk length increases, the distribution is first bounded on the sides, then spreads vertically, and finally "spills" out of the confined area. The walk distribution after 50 steps is shown in Fig. 4.

For material interfaces the hop costs become functions of lattice position. This can be handled by an exponentially slow brute force method that varies its function calls by the current node position. Alternatively, a hybrid, convolution-based method can be created by partitioning the lattice into regions, such that the overall walk distribution is calculated by combining the walk distributions of the simpler subregions. In particular, the lattice is separated into homogeneous regions, where efficient convolution methods can be used, and inhomogeneous regions, where the graph-theoretic method can handle all possible inhomogeneous configurations.

For example, if the lattice is composed of atoms of types a and b, the following partition can be created on the lattice: $\mathcal{A}$, all atoms of type a that have only neighbors of type a; $\mathcal{B}$, all atoms of type b that have only neighbors of type b; $\mathcal{C}$, all remaining atoms. Lattices $\mathcal{A}$ and $\mathcal{B}$ are homogeneous and $\mathcal{C}$ forms the interface between them. A similar partition of the walk distribution W_m is created. The walk distributions $W_m^{\mathcal{A}}$, $W_m^{\mathcal{B}}$, $W_m^{\mathcal{C}}$ contain the entries of W_m *only* for the atoms in $\mathcal{A}$, $\mathcal{B}$, or $\mathcal{C}$, respectively, and zero entries elsewhere, yielding the decomposition given by

$$W_m=W_m^{\mathcal{A}}+W_m^{\mathcal{B}}+W_m^{\mathcal{C}}. \qquad (73)$$

To calculate the walk distribution W_{m+1} all random walks of length m must be extended by a single step, by considering each node in the lattice and extending all random walks that end at that node by one step. The results can be combined as

$$W_{m+1}=W_{m+1}^{\mathcal{A}}+W_{m+1}^{\mathcal{B}}+W_{m+1}^{\mathcal{C}}. \qquad (74)$$

The contribution to each walk distribution after $m+1$ steps is obtained by the convolution of the m step distribution and the one-step mask. By extending random walks by one step in each region, the distributions may branch out to nodes in neighboring regions. Nonhomogeneous regions cannot be handled with convolution because the one-step mask is not of constant value, so walk distributions in these regions are determined directly by the graph-theoretic method.

IV. APPLICATIONS

To illustrate the methods described above, a convolution-based tight-binding moments method is used to study deformed carbon nanotubes. A tight-binding model described in [10] is used as a basis for the method. In this model the repulsive energy $E_{\rm rep}$ is given by

$$E_{\rm rep} = \sum_i f\left(\sum_j \phi(r_{ij})\right), \tag{75}$$

where r_{ij} is the distance between atoms i and j, and

$$\phi(r) = \phi_0\left(\frac{d_0}{r}\right)^m \exp\left\{m\left[-\left(\frac{r}{d_c}\right)^{m_c} + \left(\frac{d_0}{d_c}\right)^{m_c}\right]\right\}, \tag{76}$$

$$f(x) = \sum_{n=0}^{4} c_n x^n, \tag{77}$$

with $\phi_0=8.185\,55$ eV, $m=3.303\,04$, $m_c=8.6655$, $d_c=2.1052$ Å, $d_0=1.64$ Å, $c_0=-2.590\,976\,511\,819\,1$, $c_1=0.572\,115\,149\,861\,9$, $c_2=-1.789\,634\,990\,399\,6\times10^{-3}$, $c_3=2.353\,922\,151\,675\,7\times10^{-5}$, $c_4=-1.242\,511\,695\,515\,87\times10^{-7}$.

The tight-binding Hamiltonian H has off-diagonal elements described by a set of orthogonal sp^3 two-center hopping parameters $V_{ss\sigma}=-5.0$ eV, $V_{sp\sigma}=4.7$ eV, $V_{pp\sigma}=5.5$ eV, $V_{pp\pi}=-1.55$ eV, scaled with interatomic separation r according to the function

$$s(r) = \left(\frac{r_0}{r}\right)^n \exp\left\{n\left[-\left(\frac{r}{r_c}\right)^{n_c} + \left(\frac{r_0}{r_c}\right)^{n_c}\right]\right\}, \tag{78}$$

with $n=2.0$, $n_c=6.5$, $r_c=2.18$ Å, $r_0=1.536\,329$ Å. The scaling function $s(r)$ and the pair potential $\phi(r)$ are cut off at $r_{\rm cut}=2.6$ Å and go smoothly to zero at the cutoff distance [10]. The on-site elements are the atomic orbital energies of the corresponding atoms, given by $E_s=-2.99$ eV and $E_p=3.71$ eV.

To analyze single walled carbon nanotubes using this model, Chebyshev moments are found using a SSIC-based method. Symmetry is exploited to treat the structure as a one-dimensional system, instead of a fully two-dimensional hexagonal lattice. According to Eq. (37) the computational complexity is $O([M(M-1)/2]N_bN_c)$, where M is the total number of moments calculated, and N_b and N_c are constants associated with the nanotube geometry. N_b is the maximum number of neighbor orbitals, which, for carbon nanotubes is typically $N_b=37$. N_c is the number of orbitals within a nanotube of $2r_{\rm cut}$ length, as the minimal bounding length increases by $2r_{\rm cut}$ for a single step.

By using the recursive rules for multiplying Chebyshev polynomials,

$$T_{2m} = 2T_mT_m - 1, \tag{79}$$

$$T_{2m-1} = 2T_mT_{m-1} - T_1, \tag{80}$$

only $(M/2-1)$ convolutions need to be performed. So the computational complexity can be further reduced to $O([(M/2)(M/2-1)/2]N_bN_c)$. Therefore, to exactly evaluate M Chebyshev moments, the CPU time of the method scales as $O(M^2)$, as in the more general case of SSIC discussed in the previous section for either power moments or Chebyshev moments.

The method is demonstrated for a zigzag (10,0) carbon nanotube under tension and torsion [11] and for a squashed armchair (10,10) carbon nanotube [12] as shown in Fig. 5.

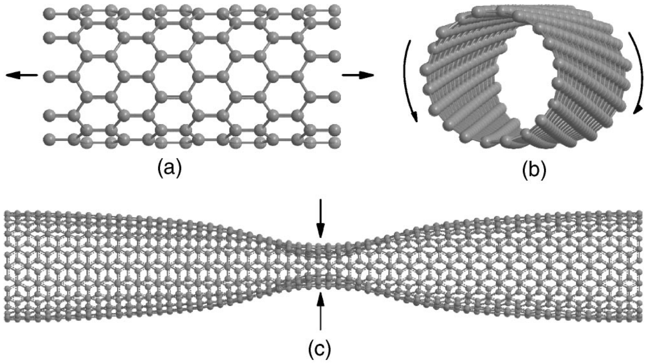

FIG. 5. Schematic illustrations of the zigzag (10,0) carbon nanotube under tension (a) and torsion (b); and the squashed armchair (10,10) carbon nanotube (c).

A. Tension

Figure 6(a) presents the DOS near the Fermi energy for different deformed zigzag (10,0) carbon nanotubes, as evaluated using several different methods. The initial structures are formed by rolling up graphene sheets and then relaxing using tight-binding molecular dynamics [10]. For the unstretched structure, the DOS's are presented from the KPM, MEM (combined with the KPM), and GFM. The GFM is based on the Hamiltonian through the Green's function given by [13,14]

$$G(E) = \frac{1}{E + i\delta - H}, \tag{81}$$

and the DOS is in turn defined by

$$D(E) = -\frac{1}{\pi}\,{\rm Im}\,G(E). \tag{82}$$

For the KPM and MEM results presented here, 1024 Chebyshev moments are used to construct the DOS. Compared to the GFM results, the resolution of the KPM DOS is insufficient to estimate the band gap, while the MEM result is significantly improved. The MEM result for the stretched structure with 3.0% strain shows an enhanced band gap near the Fermi energy. The total energies of unrelaxed and relaxed zigzag (10,0) carbon nanotubes under different strains are shown in Fig. 6(b). Clearly, the relaxed structures have lower total energy than the unrelaxed structures. The elastic strain has increased the total energy of the system quadratically as

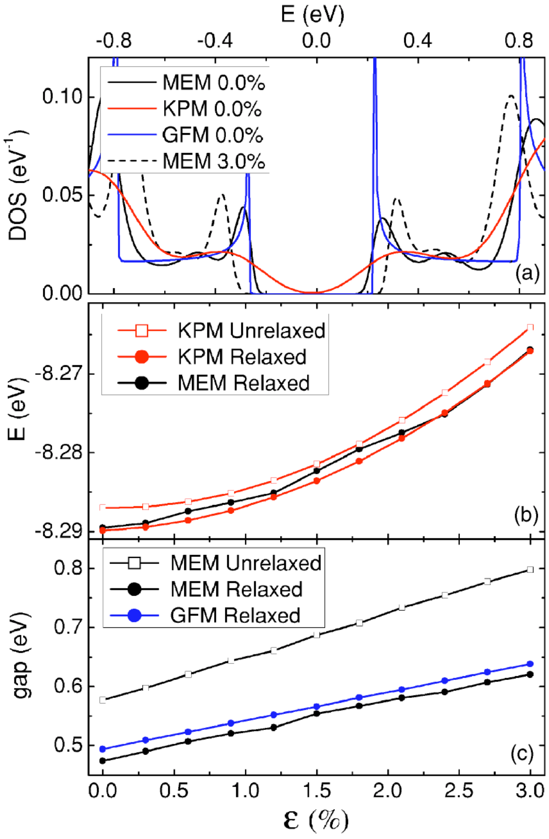

FIG. 6. (Color online) (a) The DOS near the Fermi energy for unstretched and stretched structures of zigzag (10,0) carbon nanotube calculated with different methods. The total energy (b) and band gap (c) of unrelaxed and relaxed zigzag (10,0) carbon nanotubes under different strains.

a function of strain, as expected from continuum theory. But despite the increased DOS resolution of the combined MEM approach, the total energy is computed less reliably using the MEM. In Fig. 6(c), band gaps are presented for unrelaxed and relaxed zigzag (10,0) carbon nanotubes subjected to varying strain. The values estimated from MEM are slightly smaller than those from GFM. Without relaxing the structures the band gaps are overestimated.

B. Torsion

The DOS's, total energies, and band gaps of zigzag (10,0) carbon nanotubes under torsion are also computed. Figure 7(a) shows the DOS near the Fermi energy for three different structures with κ, or twist per unit length, equal to 0.00,0.06,0.08, respectively. Here, only the results from the MEM are presented. With twist, the band gap near the Fermi energy is enhanced first and then narrowed again; also, the position of the band gap is shifted. The total energies of unrelaxed and relaxed zigzag (10,0) carbon nanotubes under torsion, computed using the KPM, are presented in Fig. 7(b). As in the case of tension, the relaxed structures have lower total energy than the unrelaxed structures, and the total energy of the system increases with strain. The band gaps of unrelaxed and relaxed zigzag (10,0) carbon nanotubes under torsion are presented in Fig. 7(c). The results estimated from the MEM are comparable with those from the GFM. As mentioned above, the band gap near the Fermi energy is enhanced and then narrowed.

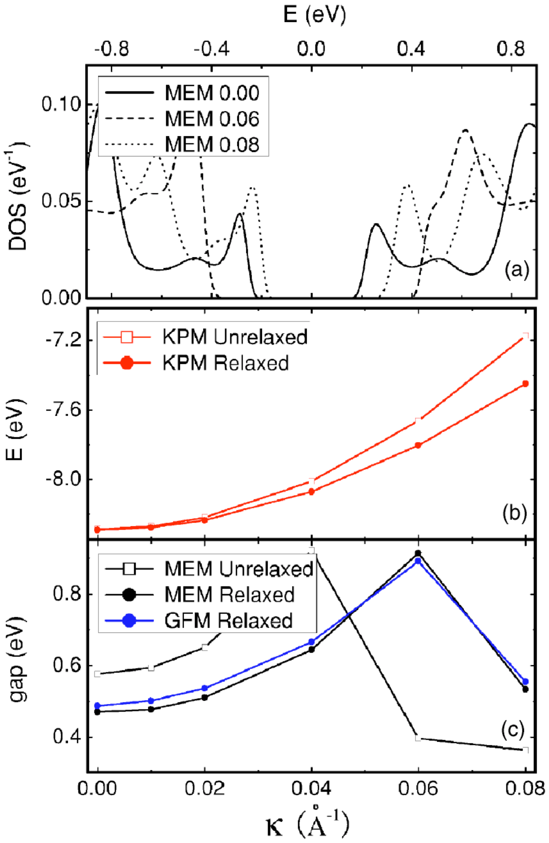

FIG. 7. (Color online) (a) The DOS near the Fermi energy of zigzag (10,0) carbon nanotubes with different twists. The total energy (b) and band gap (c) of unrelaxed and relaxed zigzag (10,0) carbon nanotubes under torsion.

C. Squashing

The two examples shown above are both periodic structures. To compare the moments methods for a structure with broken symmetry, an armchair (10,10) carbon nanotube is squashed by two tips of length 3.0 nm. Details of this problem can be found in Ref. [12]. In Fig. 8, the LDOS for one

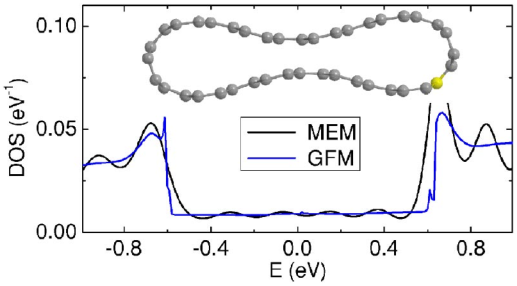

FIG. 8. (Color online) The LDOS for one atom in the squashed part of an armchair (10,10) carbon nanotube from different methods.

atom, located in the middle of a squashed section of the nanotube, is presented for both the GFM and MEM approaches. The LDOS of any other atom in the structure can be calculated using the same methods. The GFM and MEM results are clearly consistent with each other, yet the MEM method is significantly less computationally demanding for considering the local electronic structure of a single atom in the structure.

V. DISCUSSION AND CONCLUSIONS

The computational complexity and accuracy of various methods for collecting moments of the electronic DOS is analyzed here with applications in studying electronic properties of deformed carbon nanotubes. Convolution-based methods, including SSIC, are shown to be the most practical methods for obtaining power moments, and of the same complexity order as recursive methods for obtaining Chebyshev moments. A DFFT method for obtaining moments on a uniform lattice is shown to have the most desirable computational complexity, scaling as $O(M \log M)$ for carbon nanotubes if the full symmetry of the quasi-one-dimensional structure is taken into account.

While the number of power moments that can be used to reconstruct the DOS is limited by machine precision, an essentially unlimited number of Chebyshev moments can be used. With the same number of moments, the ill-posed inverse problem of inferring a spectrum from Chebyshev moments is much better conditioned than from power moments. As the Chebyshev polynomials are orthogonal to each other, it is straightforward to generate the DOS from Chebyshev moments using the KPM. The KPM approach gives accurate deformed carbon nanotube total energies with 1024 Chebyshev moments, which is consistent with the results reported in Ref. [8] for other systems. However, to accurately reconstruct DOS for deformed carbon nanotubes, it is desirable to combine the KPM with a MEM developed by Silver *et al.* [8]. Both the moments methods and the GFM can be used to obtain the DOS. A direct comparison of the computational demands of the two methods is not practical, as they are efficient for computing different results. Moments methods are desirable for computing the DOS for specific atoms, across the entire energy spectrum, while the GFM approaches may be more desirable for computing the DOS for the entire system, but within a limited energy range. As an example, moment-based tight binding is useful for studying local electronic structure in disordered or nonuniformly deformed nanostructures such as the squashed carbon nanotube considered here.

ACKNOWLEDGMENTS

The financial support of NSF Grant No. DMR 02-10131 is gratefully acknowledged. H.T.J. thanks R. Phillips for many helpful and encouraging discussions, and R. N. Silver for sharing his KPM/MEM program.

[1] F. Cyrot-Lackmann, J. Phys. Chem. Solids **29**, 1235 (1968); F. Ducastelle and F. Cyrot-Lackmann, *ibid.* **31**, 1295 (1970).

[2] A. P. Sutton, *Electronic Structure of Materials* (Oxford University Press, Oxford, 1993).

[3] D. Pettifor, *Bonding and Structure of Molecules and Solids* (Oxford University Press, Oxford, 1995).

[4] D. A. Drabold and O. F. Sankey, Phys. Rev. Lett. **70**, 3631 (1993).

[5] L. R. Mead and N. Papanicolaou, J. Math. Phys. **25**, 2404 (1984).

[6] R. H. Brown and A. E. Carlsson, Phys. Rev. B **32**, 6125 (1985).

[7] L. W. Wang, Phys. Rev. B **49**, 10 154 (1994).

[8] R. N. Silver and H. Roder, Phys. Rev. E **56**, 4822 (1997); A. F. Voter, J. D. Kress, and R. N. Silver, Phys. Rev. B **53**, 12 733 (1996).

[9] R. Saito, G. Dresselhaus, and M. S. Dresselhaus, *Physical Properties of Carbon Nanotubes* (Imperial College Press, London, 1998).

[10] C. H. Xu, C. Z. Wang, C. T. Chan, and K. M. Ho, J. Phys.: Condens. Matter **4**, 6047 (1992).

[11] H. T. Johnson, B. Liu, and Y. Huang, J. Eng. Mater. Technol. **126**, 222 (2004); B. Liu, H. Jiang, H. T. Johnson, and Y. Huang, J. Mech. Phys. Solids **52**, 1 (2003).

[12] J. Q. Lu, J. Wu, W. Duan, and B. L. Gu, Appl. Phys. Lett. **84**, 4203 (2004); J. Q. Lu, J. Wu, W. Duan, F. Liu, B. F. Zhu, and B. L. Gu, Phys. Rev. Lett. **90**, 156601 (2003).

[13] E. N. Economou, *Green's Functions in Quantum Physics* (Springer-Verlag, New York, 1993).

[14] S. Datta, *Electronic Transport in Mesoscopic Systems* (Cambridge University Press, Cambridge, U.K., 1995).