An easy and efficient way to treat Green's function for nano-devices with arbitrary shapes and multi-terminal configurations*

Yang Mou(杨 谋)†, Ran Xian-Jin(冉先进), Cui Yan(崔 岩), and Wang Rui-Qiang(王瑞强)

Laboratory of Quantum Information Technology, School of Physics and Telecommunication Engineering, South China Normal University, Guangzhou 510006*, China*

(Received 3 March 2011; revised manuscript received 24 April 2011)

The efficiency of the calculation of Green's function (GF) for nano-devices is very important because the calculation is often needed to be repeated countlessly. We present a set of efficient algorithms for the numerical calculation of GF for devices with arbitrary shapes and multi-terminal configurations. These algorithms can be used to calculate the specified blocks related to the transmission, the diagonals needed by the local density of states calculation, and the full matrix of GF, to meet different calculation levels. In addition, the algorithms for the non-equilibrium occupation and current flow are also given. All these algorithms are described using the basic theory of GF, based on a new partition strategy of the computational area. We apply these algorithms to the tight-binding graphene lattice to manifest their stability and efficiency. We also discuss the physics of the calculation results.

Keywords: Green's function, tight-binding, numerical calculation

PACS: 72.10.Bg, 73.23.–b, 72.80.Vp **DOI:** 10.1088/1674-1056/20/9/097201

1. Introduction

Transport properties of a mesoscopic system have been intensively studied in recent years.[1] The conductance of a nano-device is often calculated by means of the Landauer–Bütiker formula,[2,3] in the scattering matrix version[4] or in the Green's function (GF) version.[1] The latter is becoming more and more popular due to its compatibility with the treatment of electron interactions. The GF method is widely used in a variety of terminal-connected systems, including those from quantum dots to two-dimensional (2D) devices for the study of not only electronic transport but also other properties such as heat transport and phonon dynamics.[5,6] Because GF has to be calculated repeatedly to obtain an I–V curve, for thousands of impurity configurations, and for the convergence of the consistent potential according to the density functional theory, the efficiency the GF calculation is a vital problem for mesoscopic electronic transport. The hamiltonians of the 2D devices, which are often generated in the frame of tight-binding or finite difference, are sparse matrices of the dimension the same as the total grids in the device area (if the spin is considered, one real space grid combined with a certain spin orientation should be regarded as a new grid). According to GF's definition $\boldsymbol{G} = (E - \boldsymbol{H})^{-1}$, the mathematical problem is how to invert a large sparse matrix. To avoid the time-consuming full matrix inversion, a more efficient algorithm called the recursive GF method for the calculation of GF was developed by virtue of the sparsity of hamiltonian matrices, and it attracted the attention of both physicists[7−14] and mathematicians.[15,16] Initially only the transmission could be obtained,[7] and later it was extended to calculate local observables such as local density of state and current flow density.[10,14] However, there is a huge limitation in that they can only be applied to two-terminal quasi-one-dimensional systems,[8−11] such as one-dimensional atom chains, quantum wire, and nano-ribbon with uniform width. The algorithm was recently extended to treat non-uniform nano-ribbon with a slice-by-slice structure.[14] Currently, an easy to handle, easy to understand, efficient and widely adaptable method is very much in demanded.

In this paper, we present a set of algorithms for the efficient calculation of GF using the language of basic theory of GF based on the new strategy of layer division of a device, regardless of the device shape and the configuration of terminals. In realistic physical

*Project supported by the National Natural Science Foundation of China (Grant Nos. 10904040 and 10974058).

†Corresponding author. E-mail: yangmou1222@gmail.com

© 2011 Chinese Physical Society and IOP Publishing Ltd

calculations, different types of matrix inversion are required. For the transmission calculation only a block of GF corresponding to the contact grids is needed; when the electron–electron interaction is treated in the frame of density function theory, all diagonals of GF are required to determine the self-consistent potential; if the correlation of any two grids is demanded, all elements of GF are needed. Therefore, we classify the matrix inversion problem into three types: 1) block version, only a block of the inversal matrix is calculated; 2) diagonal version, all diagonal blocks of the inversal matrix are obtained; 3) full version, all elements of the inversal matrix are returned. The full matrix of GF is indeed rarely needed. Here, we present the algorithm for occasional need and for the completeness of the theory. In addition, we also present the algorithms for non-equilibrium electron occupation and current flow distribution. We notice that a solution of the GF problem for arbitrary device shapes was proposed in Ref. [17]. However, it is quite complicated both in theoretical description and in computational implementation, and most importantly, its efficiency does not match the complexity. We will give a comparison between the algorithm and ours.

2. Theory and discussion

All of our algorithms can be understood by means of the theory of GF. We start with an arbitrary 2 by 2 block hamiltonian matrix $\boldsymbol{H}$,

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{H}_1 & \boldsymbol{V}_{12} \\ \boldsymbol{V}_{21} & \boldsymbol{H}_2 \end{bmatrix}, \tag{1}$$

where we use symbol $\boldsymbol{V}$ to denote the off-diagonal blocks, which indicate the coupling between $\boldsymbol{H}_1$ and $\boldsymbol{H}_2$. Green's function is correspondingly a 2 by 2 block matrix,

$$\boldsymbol{G} = \begin{bmatrix} \boldsymbol{G}_1 & \boldsymbol{G}_{12} \\ \boldsymbol{G}_{21} & \boldsymbol{G}_2 \end{bmatrix}. \tag{2}$$

From the identity of the definition of Green's function, $\boldsymbol{G}(E-\boldsymbol{H}) = (E-\boldsymbol{H})\boldsymbol{G} = 1$, one can easily establish the following equations:

$$\boldsymbol{G}_1 = (E - \boldsymbol{H}_1 - \boldsymbol{V}_{12}\boldsymbol{g}_2\boldsymbol{V}_{21})^{-1}, \tag{3}$$

$$\boldsymbol{G}_{12} = \boldsymbol{G}_1\boldsymbol{V}_{12}\boldsymbol{g}_2, \tag{4}$$

$$\boldsymbol{G}_{21} = \boldsymbol{g}_2\boldsymbol{V}_{21}\boldsymbol{G}_1, \tag{5}$$

$$\boldsymbol{G}_2 = \boldsymbol{g}_2 + \boldsymbol{g}_2\boldsymbol{V}_{21}\boldsymbol{G}_{12} = \boldsymbol{g}_2 + \boldsymbol{g}_2\boldsymbol{V}_{21}\boldsymbol{G}_1\boldsymbol{V}_{12}\boldsymbol{g}_2, \tag{6}$$

where $\boldsymbol{g}_i = (E - \boldsymbol{H}_i)^{-1}$ are the isolated GFs of the diagonal blocks of the hamiltonian. These equations are indeed the Dyson's equations in the matrix block version. Equations (3)–(6) contain all the techniques of the matrix inversion in this paper.

In typical cases, the leads are attached to the device through very few edge grids of the device (see Fig. 1(a)). According to the Landauer–Büttiker formula, the transmission between lead p and lead q is calculated by $T = \mathrm{Tr}(\boldsymbol{G}\boldsymbol{\Gamma}^p\boldsymbol{G}^+\boldsymbol{\Gamma}^q)$, where $\boldsymbol{\Gamma}^i$ is the line-width function of lead i. Because the $\boldsymbol{\Gamma}$s are only relevant to the contact grids, calculating GF for the contact grids is enough for the transmission. Before we describe the algorithm, we divide the device area into layers and will treat GF layer by layer. The first layer includes only the contact grids, and the second layer contains the grids in the rest area coupling to the first layer, and so on. The process of layer division can be proceeded step by step until the remaining region cannot be divided any more.

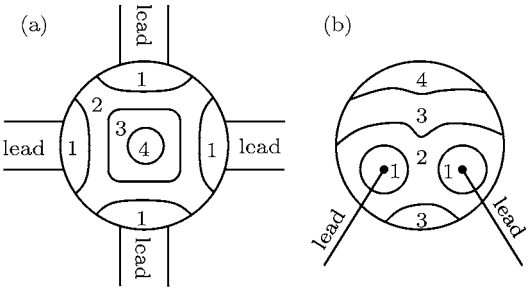


Fig. 1. Sketch of layer layout for (a) a four-terminal device with leads contacting the device through edge grids; (b) two-terminal device with leads contacting the device through inner grids.

Figure 1(a) shows a sketch of the layer layout of a four-terminal device. By the layer division, the sparse hamiltonian matrix is transformed to a block tri-angular matrix. The layer division can be applied not only to the case of the leads contacting the device through edge grids, but also to other lead configurations. Figure 1(b) shows a sketch of the layer layout for the situation in which the contact grids are two inner grids. The GF of layer 1 depends on the block of the isolated GF of the rest area, as indicated by Eq. (3), and the new block depends on that of the latter adjacent layer. The dependence is transferred layer by layer, finally onto the last layer. Therefore, we construct an update process, which starts from the last layer and ends at the first layer. Figure 2(a) illustrates the sequence of the matrix block updating. The update process is described as

Algorithm 1

$$\begin{aligned}
&\boldsymbol{g}_K = (E - \boldsymbol{H}_K)^{-1},\\
&\boldsymbol{g}_{K-1} = (E - \boldsymbol{H}_{K-1} - \boldsymbol{V}\boldsymbol{g}_K\boldsymbol{V})^{-1},\\
&\vdots\\
&\boldsymbol{g}_1 = (E - \boldsymbol{H}_1 - \boldsymbol{V}\boldsymbol{g}_2\boldsymbol{V})^{-1},
\end{aligned} \tag{7}$$

where $\boldsymbol{K}$ is the total number of layers. In the above equation and hereafter, we omit the subscript of $\boldsymbol{V}$ for abbreviation, but one could keep in mind that $\boldsymbol{V}$ only couples between adjacent layers. In each step of the update, the coupling effect of all latter layers is included in $\boldsymbol{g}_i$. After the process is completed, the GF of the contact layer is obtained as $\boldsymbol{G}_1 = \boldsymbol{g}_1$. The choice of which grids are included in the first layer is flexible and can be modified according to variety of calculation purposes. For example, if one wants to calculate the correlation of two grids of the sample, one can regard the two grids as the first layer, regardless of the real lead configuration. At each update step, two matrix-matrix productions and one matrix inversion are involved. The matrix-matrix production is very cheap because the coupling matrices between adjacent layers, $\boldsymbol{V}$'s, are all sparse. Though the hamiltonian of each layer is sparse too, the dense matrix block inversion is inevitable since the self-energy matrix caused by the layer coupling is not sparse. Therefore, the matrix inversion is the main time-consuming operation. For a 2D system, the number of layers, K, and the average grid number in layers, $n_{\rm A}$, both increase linearly with the the device dimension, and the total grid number N, i.e., the device hamiltonian dimension, increases quadratically with it, so the computational time increases with $Kn_{\rm A}^3$–N^2. The algorithm is very economical for memory consumption, and only needs a temporary memory of the size $n_M \times n_M$ to store the intermediated result $\boldsymbol{gV}$ during the update, where n_M is the maximum of the grid numbers of individual layers.

If the electron–electron interaction is taken into account in the frame of the local density function theory, the hamiltonian includes a self-consistent potential which depends on the electron distribution on all grids. For the equilibrium case, this distribution is determined by the diagonal elements of the spectrum function of the device i$(\boldsymbol{G}-\boldsymbol{G}^+)$ and the Fermi-distribution function (the non-equilibrium case will be discussed after a while). Before calculating the demanded quantities, one has to determine all the diagonals of GF. According to Eq. (6), the GF of layer 2 can be obtained through the GF of the first layer and the isolated GF block of layer 2, which are both already obtained in algorithm 1. Similarly, the GF of layer 3 can be obtained, and so on. So we construct an algorithm for calculating all the GF diagonal blocks, updating from the first layer onwards, as shown in Fig. 2(b). The algorithm is presented as

Algorithm 2 (including algorithm 1)

$$\begin{aligned}
&\boldsymbol{G}_1 = \boldsymbol{g}_1,\\
&\boldsymbol{G}_2 = \boldsymbol{g}_2 + \boldsymbol{g}_2\boldsymbol{V}\boldsymbol{G}_1\boldsymbol{V}\boldsymbol{g}_2,\\
&\vdots\\
&\boldsymbol{G}_K = \boldsymbol{g}_K + \boldsymbol{g}_K\boldsymbol{V}\boldsymbol{G}_{K-1}\boldsymbol{V}\boldsymbol{g}_K.
\end{aligned} \tag{8}$$

This algorithm involves no additional matrix inversion but four matrix-matrix production operations, only two of which are dense-dense matrix productions. The algorithm is very economical because it returns all the GF diagonal blocks at the cost of the time of the same order of that in algorithm 1, which only returns one diagonal block of the GF.

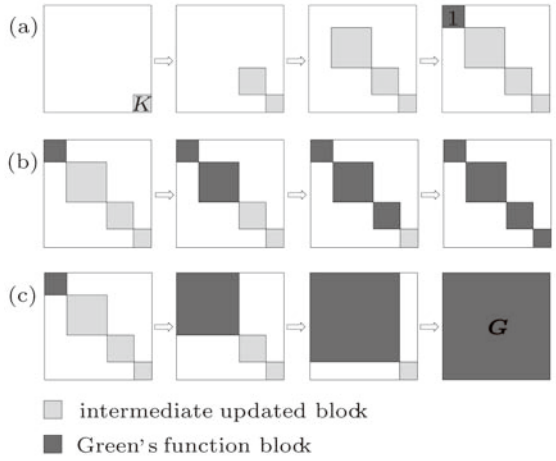


Fig. 2. Sketch of the sequences of the matrix block updating: (a) Updating sequence of algorithm 1, (b) of algorithm 2, and (c) of algorithm 3.

If one needs the full matrix of the GF, one has to additionally update the non-diagonal blocks. According to Eqs. (4) and (5), we can build an algorithm to obtain all the blocks of the GF as

Algorithm 3 (including algorithm 2)

$$\begin{aligned}
&\boldsymbol{G}_{2,i} = \boldsymbol{g}_2\boldsymbol{V}\boldsymbol{G}_{1,i}, \qquad \boldsymbol{G}_{i,2} = \boldsymbol{G}_{i,1}\boldsymbol{V}\boldsymbol{g}_2,\ i<2,\\
&\boldsymbol{G}_{3,i} = \boldsymbol{g}_3\boldsymbol{V}\boldsymbol{G}_{2,i}, \qquad \boldsymbol{G}_{i,3} = \boldsymbol{G}_{i,2}\boldsymbol{V}\boldsymbol{g}_3,\ i<3,\\
&\quad\vdots \qquad\qquad\qquad\quad \vdots\\
&\boldsymbol{G}_{K,i} = \boldsymbol{g}_K\boldsymbol{V}\boldsymbol{G}_{K-1,i}, \quad \boldsymbol{G}_{i,2} = \boldsymbol{G}_{i,K-1}\boldsymbol{V}\boldsymbol{g}_K,\ i<K.
\end{aligned} \tag{9}$$

At each update step, the intersection blocks between the current layer block and the whole block correspond to the area including all the former layers, as illustrated in Fig. 2(c).

The relations indicated in algorithm 3 allow us to calculate the electron non-equilibrium occupation, the diagonal elements of $\mathcal{G} = \boldsymbol{G}\boldsymbol{\Gamma}^f\boldsymbol{G}^+$ with $\boldsymbol{\Gamma}^f = \sum_i f^i\boldsymbol{\Gamma}^i$, where f^i is the Fermi distribution function of lead i. Because $\boldsymbol{\Gamma}^f$ is introduced through layer 1, i.e., $\boldsymbol{\Gamma}^f = \boldsymbol{\Gamma}_1^f$, the diagonal blocks of $\mathcal{G}$ are determined by $\mathcal{G}_i = \boldsymbol{G}_{i,1}\boldsymbol{\Gamma}_1^f(\boldsymbol{G}^+)_{1,i}$. Substituting the expressions of $\boldsymbol{G}_{i,1}$ indicated in algorithm 3, we have $\mathcal{G}_i = \boldsymbol{g}_i\boldsymbol{V}\mathcal{G}_{i-1}\boldsymbol{V}\boldsymbol{g}_i^+$. So we can construct the update process to obtain all the diagonal blocks of $\mathcal{G}$ as follows:

Algorithm $2'$ (including algorithm 1)

$$\begin{aligned}
&\mathcal{G}_1 = \boldsymbol{g}_1\boldsymbol{\Gamma}_1^f\boldsymbol{g}_1^+,\\
&\mathcal{G}_2 = \boldsymbol{g}_2\boldsymbol{V}\mathcal{G}_1\boldsymbol{V}\boldsymbol{g}_2^+,\\
&\vdots\\
&\mathcal{G}_K = \boldsymbol{g}_K\boldsymbol{V}\mathcal{G}_{K-1}\boldsymbol{V}\boldsymbol{g}_K^+.
\end{aligned} \tag{10}$$

The computational complexity of algorithm $2'$ is the same as that of algorithm 2. The distribution of current flow in real space is also an interesting quantity, the definition of which can be found in, e.g., Refs. [17] and [18]. The current flow needs both the diagonal blocks (intra-layer flow) and the secondary diagonal blocks (inter-layer flow), the former can be obtained by algorithm $2'$, and the latter can be calculated by $\mathcal{G}_{i,i-1} = \boldsymbol{G}_{i,1}\boldsymbol{\Gamma}_1^f(\boldsymbol{G}^+)_{1,i-1}$ and $\mathcal{G}_{i-1,i} = \mathcal{G}_{i,i-1}^+$. Substituting the expressions of $\boldsymbol{G}_{i,1}$ appearing in algorithm 3, we have the update process for the blocks needed by the current flow calculation.

Algorithm $2''$ (including algorithm $2'$)

$$\begin{aligned}
&\mathcal{G}_{2,1} = \boldsymbol{g}_2\boldsymbol{V}\mathcal{G}_1,\\
&\mathcal{G}_{3,2} = \boldsymbol{g}_3\boldsymbol{V}\mathcal{G}_2,\\
&\vdots\\
&\mathcal{G}_{K,K-1} = \boldsymbol{g}_K\boldsymbol{V}\mathcal{G}_{K-1}.
\end{aligned} \tag{11}$$

Comparing algorithm $2''$ with algorithm $2'$, one can find that $\mathcal{G}_{i-1,i}$ are the byproducts of $\mathcal{G}_i$ because $\mathcal{G}_i = \boldsymbol{g}_i\boldsymbol{V}\mathcal{G}_{i-1,i}$, that is to say, if we need the occupation and the current distributions simultaneously, the latter will be obtained without any cost. Both algorithms $2'$ and $2''$ rely on the relations indicated in algorithm 3, while they do not depend on algorithm 3. The advantage of the two algorithms lies in the explicit calculation of GF blocks $\boldsymbol{G}_{i,1}$, which is not needed.

The algorithm proposed by Kazymyrenko *et al.*,[17] which is attractive due to its block-inversion-free feature, works for almost the same things as ours. They calculated the demanded GF elements one-by-one. Each GF element calculation required an intermediate GF block with the average dimension $n_{\rm A}$, and the intermediate block had to be updated N times. Therefore their computational time for the GF of the contact grids is of the order of $n_{\rm c}^2N^2$, more costly than ours (see the statement following algorithm 1), where $n_{\rm c}$ is the number of contact grids. Additionally, their algorithm is based on a very complicated categorizing of the grids, and the indices of the intermediated updating blocks are very intricate, which brings much more complexity for both the theory and the computational implementation.

To test the efficiency and validation of these algorithms, we implement them with Matlab codes. The test martrices are those of the tight-binding hamiltonians[19–21] of four-terminal square graphene[22] samples with on-site energies being randomly generated. Figure 3(a) shows the curves of the computational time for algorithms 1–3 and the Matlab internal command for matrix inverse *inv* as functions of the matrix dimension, and Fig. 3(b) shows the ratio between the computational time of algorithms 1–3 and that of *inv*. The insert shows these curves in a larger computational time interval so that the comparison of them is clearer for the large matrix dimension (the curves of algorithms 1 and 2 almost overlap with the horizontal axis so as to be hardly distinguishable). One can see that algorithms 1 and 2 are extremely economical and algorithm 3 consumes much more computational

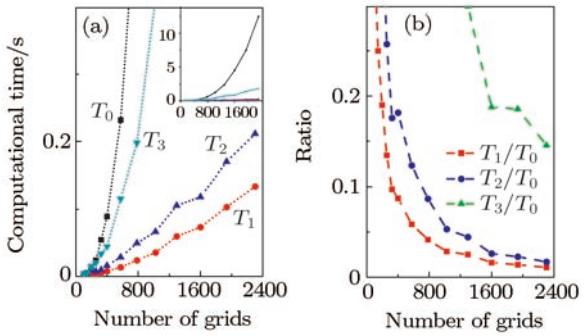


Fig. 3. (colour online) (a) The curves of computational time of algorithms 1–3 (T_1, T_2 and T_3) and the Matlab internal matrix inversion function *inv* (T_0) versus the hamiltonian matrix dimension. The insert shows the curves of T_0 through T_3 in a larger computational time interval with the same coordinate units as those in the main figure. (b) The curves of the ratio T_1/T_0, T_2/T_0 and T_3/T_0 as functions of the hamiltonian matrix dimension.

time than algorithms 1 and 2 while much less computational time than the inverse matrix. The numbers

of contact grids on each edge are fixed to 10 when the size of the sample varies. Our algorithms depend slightly on the number of contact grids.

Figure 4(a) shows the computational time of algorithm 2 for different contact grid numbers. The indices of the contact grids, which indicate the positions of these grids, are randomly generated. The computational time has the trend of increasing with increasing contact grid number. However, the increase saturates when the contact grid number becomes larger. The increase of the contact grid number results in an increase in the computational time for the grid numbers of the first several layers. The saturation is due to the interconnection of the area of layers (the areas of contact grids are disconnected in our examples). Figure 4(b) shows the validity and stability of algorithms 3. We define the maximum element of $|(E-\boldsymbol{H})\boldsymbol{G}-1|$ as the measurement of the deviation of algorithm 3. One can see that the deviation always remains at a low level and does not exhibit a tendency to increase when the matrix dimension increases, which means that the algorithm is stable.

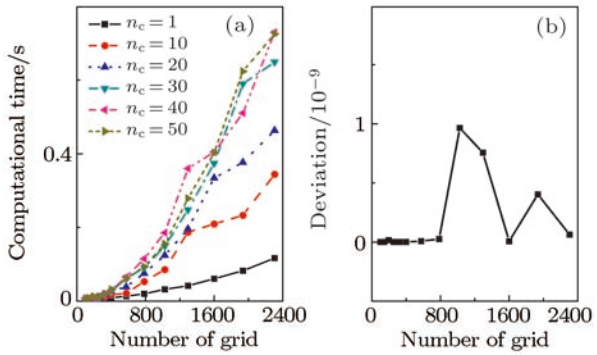


Fig. 4. (colour online) (a) The curves of computational time of algorithm 2 versus the hamiltonian dimension for different contact grid numbers n_c. (b) The curve of the deviation of algorithm 3 versus the matrix dimension, where the deviation is defined as the maximum element of the matrix $|(E-\boldsymbol{H})\boldsymbol{G}-1|$.

Finally, we present example calculations for nonequilibrium occupation and electron flow distribution using algorithm $2'$ and algorithm $2''$. The calculations are carried out on a square graphene containing about 2200 atoms with four terminals attached to the edges. A magnetic field of 0.034 flux quanta per hexagonal cell is applied to the sample, and a harmonic quantum well is placed at the sample centre with the well depth t and well diameter $35a_0$, where t is the lattice hopping intensity and a_0 is the C–C bond length. The hamiltonian is generated by the tight-binding model. The electrons are injected from the left terminal with the energy $E=-0.5t$ (the Dirac point energy outside the quantum well is set to be the energy reference). The calculation results are shown in Fig. 5. Electrons enter from the left side, are scattered by the potential and bent by the magnetic field, and leak through the other three sides. One can expect that the electron density decays from left to right as a result of the leakage, it has an extra structure due to the potential, and is asymmetric between the upper and lower sides because of the influence of the magnetic field. All the features from the physical understanding can be verified in Fig. 5(a). The electron flow distribution can display the physical picture more intuitively as shown in Fig. 5(b). One can see that most of the electrons entering the computational area go anti-clockwise and get out through the lower side. Moreover, an interesting exclusive phenomenon of graphene can be identified in the figure. The electrons that enter the quantum well curve reversely, i.e., move clockwise. This is because the electrons experience a hole–electron transition when they enter the centre area. When a magnetic field exists, the cyclotron trajectories of the hole-like particles and the electron-like particles are reverse.

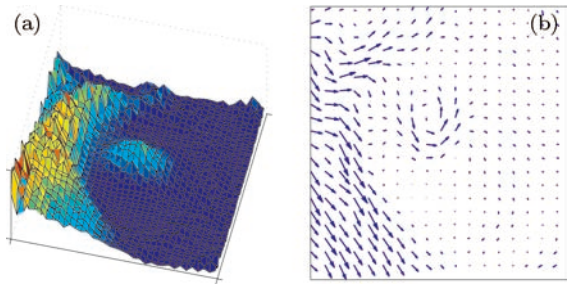


Fig. 5. (colour online) The profiles of the nonequilibrium occupation (a) and the electron flow distribution (b) of a four-terminal square graphene with a harmonic potential at the centre under a unified magnetic field.

3. Conclusion

We have presented several simple and efficient algorithms for calculating the lattice GF, the nonequilibrium occupation and current flow distribution, based on a general layer division strategy compatible with arbitrary device shapes and lead configurations. These algorithms are described using the theory of GF itself. The efficiency of these algorithms is benefits from the fact that no unwanted GF blocks are calculated. The stability and efficiency of these algorithms are tested by adopting the hamiltonians of the tight-binding graphene lattice as sample matrices.

References

[1] Datta S 1995 *Electronic Transport in Mesoscopic Systems* (Cambridge: Cambridge University Press)

[2] Landauer R 1957 *IBM J. Res. Dev.* **1** 223

[3] Büttiker M, Imry Y, Landauer R and Pinhas S 1985 *Phys. Rev.* B **31** 6207

[4] Kassubek F, Stafford C A and Grabert H 1999 *Phys. Rev.* B **59** 7560

[5] Zhang W, Mingo N and Fisher T S 2007 *Phys. Rev.* B **76** 195429

[6] Huang Z, Fisher T S and Murthy J Y 2010 *J. Appl. Phys.* **108** 094319

[7] Lee P A and Fisher D S 1981 *Phys. Rev. Lett.* **47** 882

[8] Sanvito S, Lambert C J, Jefferson J H and Bratkovsky A M 1999 *Phys. Rev.* B **59** 11936

[9] Metalidis G and Bruno P 2005 *Phys. Rev.* B **72** 235304

[10] Cresti A, Farchioni R, Grosso G and Parravicini G P 2003 *Phys. Rev.* B **68** 075306

[11] Polinák P K, Lambert C J, Koltai J and Cserti J 2006 *Phys. Rev.* B **74** 132508

[12] Lassl A, Schlagheck P and Richter K 2007 *Phys. Rev.* B **75** 045346

[13] Pecchia A, Penazzi G, Salvucci L and Carlo A D 2008 *New J. Phys.* **10** 065022

[14] Areshkin D A and Nikolić B K 2010 *Phys. Rev.* B **81** 155450

[15] Petersen D K, Sørensen H H B, Hansen P C, Skelboe S and Stokbro K 2008 *J. Comput. Phys.* **227** 3174

[16] Wimmer M and Richter K 2009 *J. Comput. Phys.* **228** 8548

[17] Kazymyrenko K and Waintal X 2008 *Phys. Rev.* B **77** 115119

[18] Caroli C, Combescot R, Nozieres P and Saint-James D 1971 *J. Phys.* C **4** 916

[19] Peng J and Li S S 2010 *Appl. Phys. Lett.* **97** 242105

[20] Chen X W, Song K H, Zhou B H, Wang H Y and Zhou G H 2011 *Appl. Phys. Lett.* **98** 093111

[21] Hu H, Cai J M, Zhang C D, Gao M, Pan Y, Du S X, Sun Q F, Niu Q, Xie X C and Gao H J 2010 *Chin. Phys.* B **19** 037202

[22] Novoselov K S, Geim A K, Morozov S V, Jiang D, Zhang Y, Dubonos S V, Grigorieva I V and Firsov A A 2004 *Science* **306** 666