BEILSTEIN JOURNAL OF NANOTECHNOLOGY

Simple and efficient way of speeding up transmission calculations with *k*-point sampling

Jesper Toft Falkenberg and Mads Brandbyge*

Full Research Paper

Open Access

Address:
Center for Nanostructured Graphene (CNG), Department of Micro- and Nanotechnology (DTU Nanotech), Technical University of Denmark, DK-2800 Kongens Lyngby, Denmark

Email:
Mads Brandbyge* - mads.brandbyge@nanotech.dtu.dk

* Corresponding author

Keywords:
density functional theory; electronic conductance; interpolation; post-processing; shortest-path

Beilstein J. Nanotechnol. **2015,** *6,* 1603–1608.
doi:10.3762/bjnano.6.164

Received: 30 April 2015
Accepted: 09 July 2015
Published: 24 July 2015

This article is part of the Thematic Series "Functional nanostructures – electronic structure, charge and heat transport".

Guest Editor: F. Pauly

Abstract

The transmissions as functions of energy are central for electron or phonon transport in the Landauer transport picture. We suggest a simple and computationally "cheap" post-processing scheme to interpolate transmission functions over *k*-points to get smooth well-converged average transmission functions. This is relevant for data obtained using typical "expensive" first principles calculations where the leads/electrodes are described by periodic boundary conditions. We show examples of transport in graphene structures where a speed-up of an order of magnitude is easily obtained.

Introduction

Calculations of electronic conductance based on first principle methods such as density functional theory (DFT) provide a valuable tool in order to gain insights into electronic transport in nano-conductors and comparison to experiments without employing fitting parameters. This is for example the case in the field of single-molecular devices [1]. Popular methods are based on DFT in combination with the non-equilibrium Green's function approach (DFT-NEGF), see, e.g., [2-4], or scattering wave-function approaches [5]. The electrodes in such calculations are typically treated employing periodic boundary conditions in the direction transverse to the transport direction with a corresponding *k*-point average of the electronic states and transmissions. This means that for each transverse *k*-point the system essentially behaves as a one-dimensional conductor with diverging density of states and discontinuities in the transmission function at energies corresponding to band on-sets/channel openings. It is well known that often in order to obtain smooth, well-converged density of states and transmissions as a function of energy, a substantial number of transverse *k*-points are needed due to the rapid variations of these functions for individual *k*-points [6]. Certain quantities, for example the Seebeck coefficient and thermo-electric figure of merit (ZT), are based on the detailed behavior of the transmission [7,8] and thus exceedingly sensitive to energy and *k*-resolution of the calculations. This can amount to a significant computational burden for large systems treated by first principle methods. Thus it is highly interesting to devise simple ways to make this more efficient and get maximum information from the data.

Sophisticated methods to tackle this include the transformation to a smaller basis-set using maximally localized Wannier functions [9], or to construct a optimized minimal basis-set and using this to determine the transmission [10]. Both require one to examine the details of the chemical bonds in the system, relevant energy windows, and storing wavefunctions, which tend to be elaborate. In this paper we present a simple and efficient post-processing interpolation scheme which can significantly speed up the convergence with respect to *k*-points. We illustrate the method by applying it to various graphene-based nanostructures which are prone to bad convergence due to its vanishing density of states at the Fermi level.

In the remaining parts of the paper we first explain the workings of the interpolation scheme in section Results and Discussion, while we investigate various test cases in section Example cases, and finally discuss limitations to the scheme and conclude in section Conclusion.

Results and Discussion

Description of the method

The use of computationally "expensive" first principles DFT-NEGF calculations for determining the transmission through nano-structured systems is limited by the amount of time one can afford to spend on the *k*-grid resolution. Often the rough behavior of the transmission is already seen with a limited number of *k*-points but the convergence of the average is slow since the functions are changing abruptly with energy, e.g., around a band onset or a resonance. The position of the abrupt feature will typically shift in a smooth way with changing *k*-point, but a linear interpolation of the curves between two consecutive *k*-points will be of little use since it will simply contain, say, half of each abrupt feature. Instead, we propose an interpolation scheme which can make use of a coarse, non-converged *k*-grid, and thus reduce the computation cost simply by using a "clever" technique to approximate the transmission curves for intermediate *k*-points. The method does not magically guess the correct interpolated curves, and one has to have a reasonable amount of *k*-point resolved transmission curves in order to obtain a useful result, but smooth averaged curves can be obtained, as we will illustrate below, using a significantly smaller number of *k*-points. The interpolation is done by using a shortest-path solver to determine a correspondence between two *k*-adjacent curves. The correspondence is then used to find intermediate curves which can be used to determine the averaged transmission. The proposed interpolation scheme consists of three separate steps, which will be described in detail in the following sections.

In order to show the validity of our proposed scheme we have determined the transmission through pristine graphene for increasing number of transverse *k*-points (see Figure 1a). The computational details are given in section Example cases. We compare $N = 6, 8, \ldots, 54$ to a well-converged calculation with $N = 600$ *k*-points. By applying our algorithm we significantly reduce the mean absolute deviation from a fully converged transmission, as shown in Figure 1b. We see that the interpolated transmissions converge much quicker than the raw data. Fitting data with a power law, we can estimate the amount of *k*-points needed in order to obtain a deviation of less than one percent, thus giving a speed-up factor of $\eta = 220/41 = 5.37$.

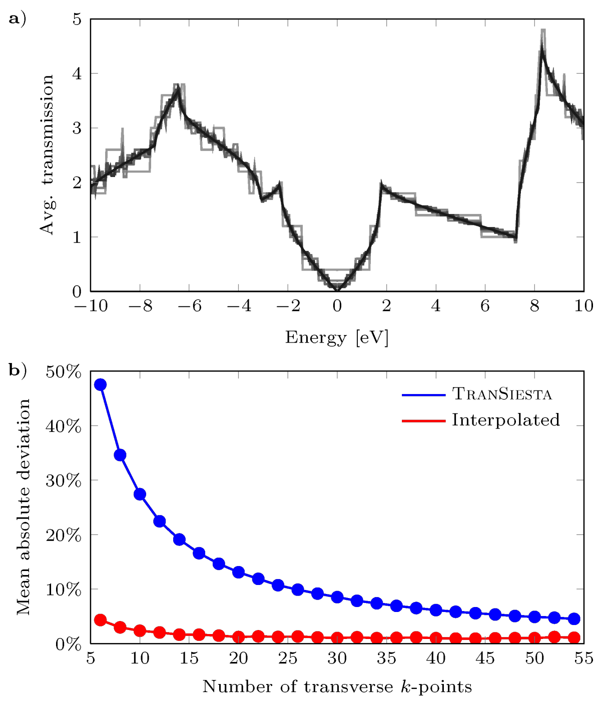

Figure 1: Analysis of transmission in pristine graphene: (a) Transmission converges as the number of transverse *k*-points increases (darker colors). (b) The mean absolute deviation from fully converged data shown for TranSiesta data before and after using our interpolation scheme.

Transform of data

In the following we will outline the method in general terms and denote the data points by (x, y), corresponding to the transmission function data point, $(E, T_k(E))$, for given *k*-point and energy in the concrete examples. Initially, we transform the set of data points (x, y) into points with a maximum Euclidian distance ℓ. The new data points span line segments (when connected) on the curve $\mathcal{L}$ consisting of N different individual points. In practice the transformation of data is done by looping through all segment lengths L and inserting additional data points if $L > \ell$. Figure 2a shows data where the distance between points a, c is much larger than ℓ, and thus requires extra points between points a, c. The optimal value of the segment length ℓ depends on the given data. A value close to the median of the original point distance is a good starting guess, and is chosen as the default value.

The two curves in Figure 2a are required to smoothly interpolate into each other – a process which can be estimated by hand (gray arrows). The proper path is found by overall minimization of the interpolation distance, which will described in the following subsection.

Correspondence between curves

Given two data sets, i.e., the curves $\mathcal{L}_1$, $\mathcal{L}_2$, we construct a matrix $\mathcal{D}$ containing the weighted Euclidean distances between points on opposite curves,

$$\mathcal{D}_{ij} = \sqrt{w_x\left(\tilde{x}_1^i - \tilde{x}_2^j\right)^2 + w_y\left(\tilde{y}_1^i - \tilde{y}_2^j\right)^2}, \qquad (1)$$

where $i = 1 \ldots N_1$, $j = 1 \ldots N_2$ denote all points on the curves $\mathcal{L}_1$, $\mathcal{L}_2$, respectively. Thus, the dimension of the matrix $\mathcal{D}$ is $N_1 \times N_2$. The weights $w_{x,y}$ are added to ensure a degree of tunability when interpolating. This is due to the fact that x and y are different quantities and thus have different scales. In order to interpolate one curve into the other we need to determine the shortest path for all points on either curve. This is done by considering the distance matrix as a landscape and moving from the start (1, 1) to the destination (N_1, N_2) using the smallest cost possible. This is essentially identical to finding the minimum energy path on an potential energy surface, which can be done using either string methods or the nudged elastic band approach [11]. We will instead use Dijkstra's algorithm, which loops through coordinates in the distance landscape and iteratively compares all found paths until the target point is found. Each iteration is done while keeping track of the cost and path. For more information see [12]. The coordinates along the returned shortest path is saved in a correspondence matrix $\mathcal{C}$ containing two columns ($\mathcal{C}_1$, $\mathcal{C}_2$), which is used in the final step of our routine. The distance matrix $\mathcal{D}$ in Figure 2b shows the shortest path as a white line propagating along the distance landscape minimum (black/dark blue).

Interpolation of curves

Once the data is transformed and a correspondence matrix is obtained we interpolate the curves $\mathcal{L}_1$, $\mathcal{L}_2$. Instead of a simple linear interpolation we use the correspondence matrix to obtain intermediate curves. A simple linear, discretized interpolation is used, i.e.,

$$x_{12\alpha} = \alpha x_1[\mathcal{C}_1] + (1-\alpha)x_2[\mathcal{C}_2] \qquad (2)$$

$$y_{12\alpha} = \alpha y_1[\mathcal{C}_1] + (1-\alpha)y_2[\mathcal{C}_2], \qquad (3)$$

where α is the interpolation parameter between zero (curve $\mathcal{L}_2$) and one (curve $\mathcal{L}_1$), and x_i[M] means that we use the indices M of the vector x_i. The found intermediate data sets ($x_{12\alpha}$, $y_{12\alpha}$) can then be sampled back to the original grid. In practice we loop over α to generate the different interpolated curves. Figure 2c shows the data in Figure 2a interpolated into $N = 10$ intermediate curves and transformed back onto the original grid.

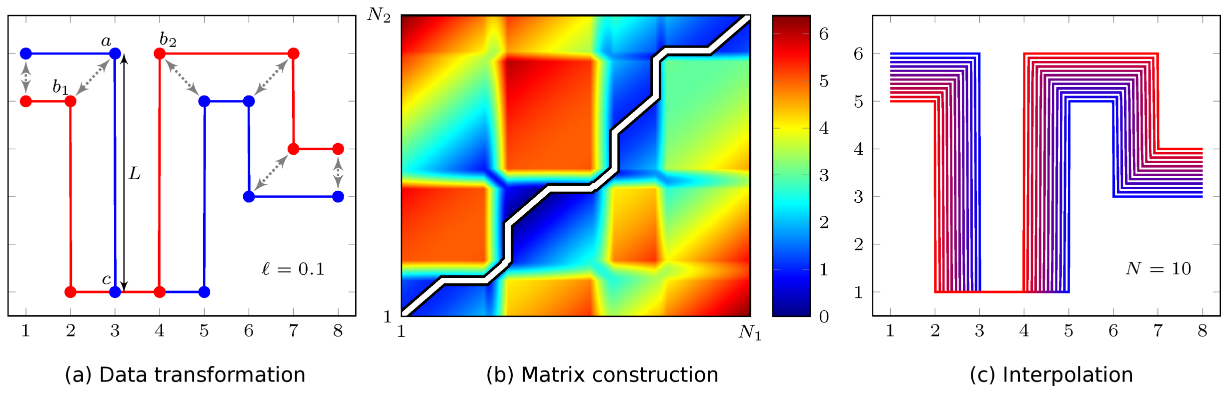

Figure 2: We seek an interpolation of the two datasets (red, blue) containing N_1 and N_2 points, respectively. The three steps in the algorithm: (a) The original data sets are transformed into line segments of a maximum length ℓ = 0.1. Thus, for example the line segment *a*–*c* is split into 50 points. (b) The weighted Euclidean distance is calculated for all point on both curves, and the shortest path from (1, 1) to (N_1, N_2) is found. (c) The curves are linearly interpolated using the found path with N = 10 intermediate curves.

We note that the choice of weights w_x, w_y depends on the input data scales. Changing the values can highly affect the outcome of the shortest-path solver, since the distance landscape is changed. Usually, it is advisable to rescale the data (using the weights) so that the two data ranges are comparable. Similarly, the length ℓ has to be chosen wisely: A large value can result in crude interpolations while a too small value makes the algorithm too time-consuming.

Finally, we note that the algorithm described in the previous subsections allows us to interpolate data in general. We can apply the algorithm specifically to transmissions and DOS by providing it with the needed data. In the case of TranSiesta and the utility TBTrans we need to weight the interpolated curves using k-grid weights and sum to obtain average transmission curves. In the case of 3D transport we have a 2D k-grid, which can be investigated using bilinear interpolation. In the following section we apply the interpolation scheme to three example cases.

Example cases

The usefulness of the presented algorithm is showcased by considering transmission calculations through the simulated nano-systems shown in Figure 3: (a) a pristine graphene sheet, (b) a graphene nano-constriction [13], and (c) hydrogenated kinked graphene [14]. The shown structures have minimal unit-cells in the transverse transmission direction due to periodic boundary conditions, and carbon atoms are shown in black while hydrogen atoms are shown in white.

The transmission through the graphene sheet in Figure 3a is calculated with the TranSiesta simulation suite, which utilizes a localized basis set. For the present work a DZP basis set is used in conjunction with a mesh cut-off of 300 Ry, a force tolerance of 0.02 eV/Å, and a Monkhorst–Pack grid of 24 × 5 ensuring absolute convergence. Exchange and correlation is described using the PBE GGA functional [15]. A minimal transverse unit-cell is used due to periodic boundary conditions, while an energy window of ±10 eV around the Fermi energy is considered both for a crude transverse transmission k-grid ($N_k = 20$) and a fully converged k-grid ($N_k = 600$). The coarse transmission spectrum is interpolated using the presented interpolation scheme, as can be seen in the upper row in Figure 4. The upper window of each column describes the full transmission versus k-point. A good agreement is seen between the interpolated and the converged data sets. A few places the interpolation guesses incorrectly (for instance around $E = -4$ eV for $k = 0.25\ \pi/a$). These occur since the input data is too coarse. However, despite the few differences between interpolated data and converged data, the averaged curves have a remarkable resemblance and the mean deviation is below 2% (Figure 1b).

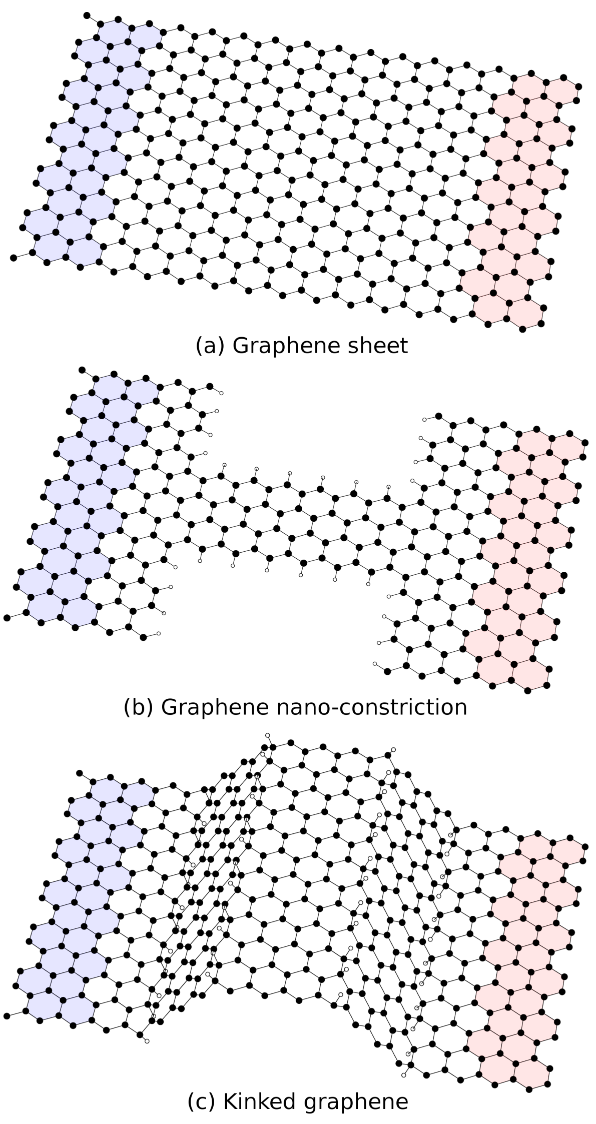

Figure 3: Structures used as example cases: (a) a graphene sheet consisting entirely of carbon atoms, (b) a graphene hydrogen-passivated nano-constriction, and (c) a kinked graphene sheet decorated with hydrogen along the kink lines. The left and right electrodes have been highlighted with blue and red, respectively. Minimal unit-cells in the transverse transmission direction have been used due to the periodicity. Carbon atoms are shown in black, while hydrogen atoms are shown in white.

The transmission through the graphene nano-constriction and kinked graphene shown in Figure 3b and Figure 3c have been extracted from the original datasets (see [13,14]), and are shown in the middle and lower windows in Figure 4. As with the pristine graphene example we see a remarkable resemblance to the converged data set. The discrepancies are negligible, which stems from the fact that a finer k-point sampling has been performed in the original data. In the three cases we obtain speed-ups of approximately 5, 6, and 8, for the pristine graphene sheet, the graphene nano-constriction, and the kinked

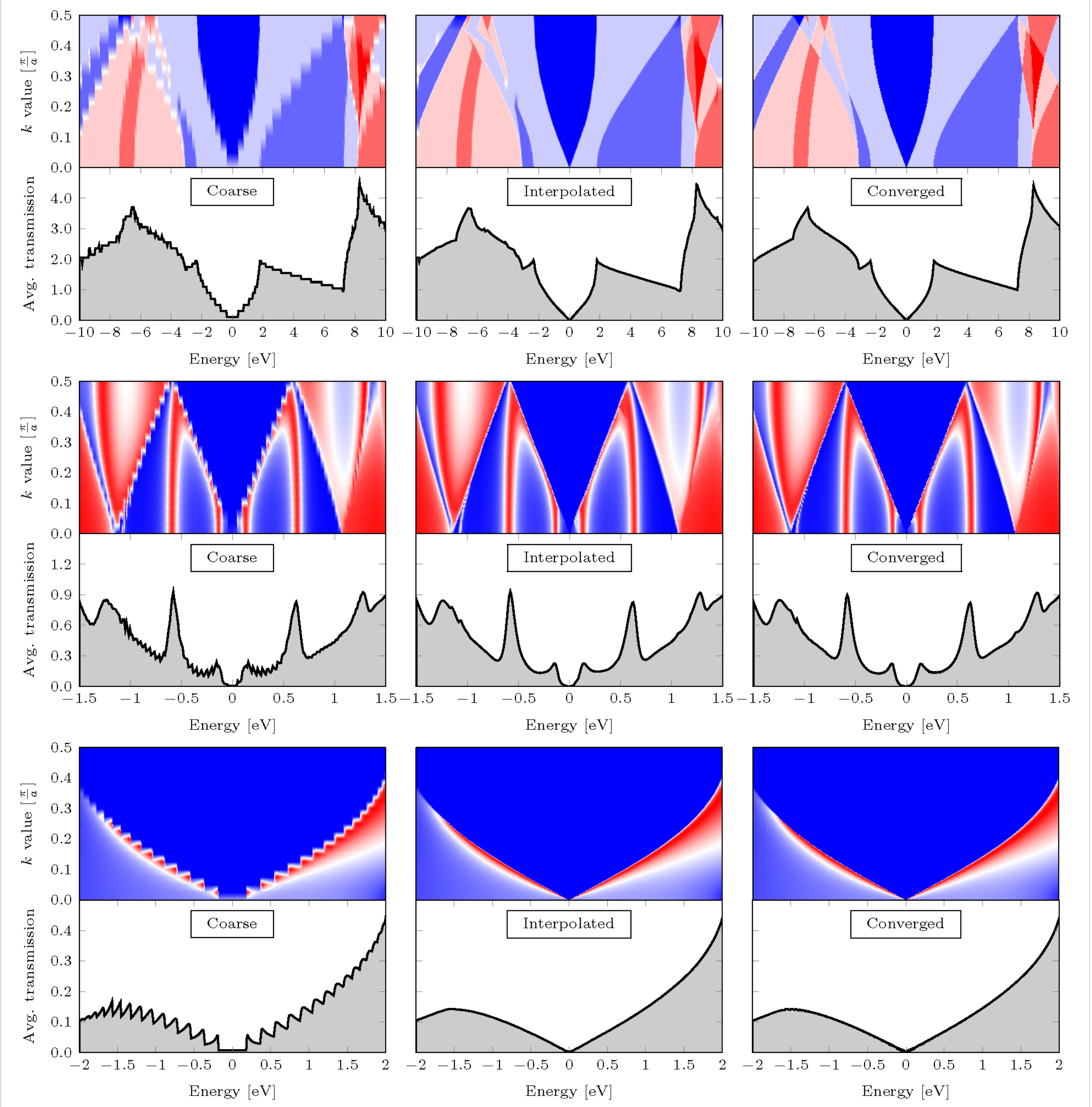

Figure 4: Algorithm applied to raw transmission data for the graphene-based systems in Figure 3. The raw data (left column) is interpolated (middle column) and compared to the fully converged transmission curve (right column). The upper window of each subfigure is the full transmission for each *k*-value, while the lower window shows the averaged value.

graphene sheet, respectively. Thus, we have demonstrated that by applying a simple post-processing interpolation scheme we can speed up convergence of roughly an order of magnitude.

Conclusion

We have presented a simple post-processing interpolation scheme which speeds up transmission calculations on nanostructured materials. The algorithm uses a shortest-path solver to determine the optimal interpolation of a set of *k*-dependent transmission curves, which ultimately can be summed to obtain a smoothed average transmission.

We note that as a post-processing tool the algorithm relies on the quality of the original data. Since this data is used as a base for the interpolation any fluctuations will be present in the interpolated data and thus propagate to the final result. The present implementation of the shortest-path solver is based on Dijkstra’s algorithm which is stable but very slow [12]. A far quicker implementation would be to use a heuristic to guide the shortest-path search in the distance landscape, thus changing the solver to an industry standard algorithm known as an A*-search [16]. However, due to the complexity of the input data the construction of such a heuristic is not a straight-forward task.

By considering three sample cases we have demonstrated that it can speed up calculations by roughly an order of magnitude. We have illustrated the method using electron transport through graphene nano-structures and *k*-point averaging of transmission functions. However, the method is generally applicable also to phonon transport and to other functions such as density of states or other types of interpolation parameters such as electrostatic gating etc.

Our interpolation scheme can easily be implemented in already existing code. We provide a sample MatLAB code (Supporting Information File 1) that can read and interpolate data obtained from TranSiesta and TBTrans [2], which are built on-top of the ab initio software package Siesta [17].

Supporting Information

Supporting Information File 1

A sample MatLAB code that can read and interpolate data obtained from TranSiesta and TBTrans.
[http://www.beilstein-journals.org/bjnano/content/supplementary/2190-4286-6-164-S1.m]

Acknowledgements

The authors thank the Danish e-Infrastructure Cooperation (DeIC) for providing computer resources, and the Lundbeck foundation for support (R95-A10510). The Center for Nanostructured Graphene is sponsored by the Danish National Research Foundation.

References

1. Cuevas, J. C.; Scheer, E. *Molecular Electronics - An Introduction to Theory and Experiment;* World Scientific Series in Nanoscience and Nanotechnology, Vol. 1; World Scientific Publishing Co Pte Ltd: Singapore, 2010.
2. Brandbyge, M.; Mozos, J.-L.; Ordejón, P.; Taylor, J.; Stokbro, K. *Phys. Rev. B* **2002,** *65,* 165401. doi:10.1103/PhysRevB.65.165401
3. Rocha, A. R.; García-suárez, V. M.; Bailey, S. W.; Lambert, C. J.; Ferrer, J.; Sanvito, S. *Nat. Mater.* **2005,** *4,* 335–339. doi:10.1038/nmat1349
4. Strange, M.; Kristensen, I. S.; Thygesen, K. S.; Jacobsen, K. W. *J. Chem. Phys.* **2008,** *128,* 114714. doi:10.1063/1.2839275
5. Garcia-Lekue, A.; Wang, L. W. *Phys. Rev. B* **2010,** *82,* 035410. doi:10.1103/PhysRevB.82.035410
6. Thygesen, K. S.; Jacobsen, K. W. *Phys. Rev. B* **2005,** *72,* 033401. doi:10.1103/PhysRevB.72.033401
7. Paulsson, M.; Datta, S. *Phys. Rev. B* **2003,** *67,* 241403. doi:10.1103/PhysRevB.67.241403
8. Zotti, L. A.; Bürkle, M.; Pauly, F.; Lee, W.; Kim, K.; Jeong, W.; Asai, Y.; Reddy, P.; Cuevas, J. C. *New J. Phys.* **2014,** *16,* 015004. doi:10.1088/1367-2630/16/1/015004
9. Marzari, N.; Vanderbilt, D. *Phys. Rev. B* **1997,** *56,* 12847. doi:10.1103/PhysRevB.56.12847
10. Pizzi, G.; Volja, D.; Kozinsky, B.; Fornari, M.; Marzari, N. *Comput. Phys. Commun.* **2014,** *185,* 422–429. doi:10.1016/j.cpc.2013.09.015
11. Sheppard, D.; Terrell, R.; Henkelman, G. *J. Chem. Phys.* **2008,** *128,* 134106. doi:10.1063/1.2841941
12. Dijkstra, E. W. *Numerische Mathematik* **1959,** *1,* 269–271. doi:10.1007/BF01386390
13. Gunst, T.; Lü, J.-T.; Hedegård, P.; Brandbyge, M. *Phys. Rev. B* **2013,** *88,* 161401. doi:10.1103/PhysRevB.88.161401
14. Rasmussen, J. T.; Gunst, T.; Bøggild, P.; Jauho, A.-P.; Brandbyge, M. *Beilstein J. Nanotechnol.* **2013,** *4,* 103–110. doi:10.3762/bjnano.4.12
15. Perdew, J. P.; Burke, K.; Ernzerhof, M. *Phys. Rev. Lett.* **1996,** *77,* 3865. doi:10.1103/PhysRevLett.77.3865
16. Hart, P. E.; Nilsson, N. J.; Raphael, B. *IEEE Trans. Syst. Sci. Cybern.* **1968,** *4,* 100–107. doi:10.1109/TSSC.1968.300136
17. Soler, J. M.; Artacho, E.; Gale, J. D.; García, A.; Junquera, J.; Ordejón, P.; Sánchez-Portal, D. *J. Phys.: Condens. Matter* **2002,** *14,* 2745–2779. doi:10.1088/0953-8984/14/11/302

License and Terms