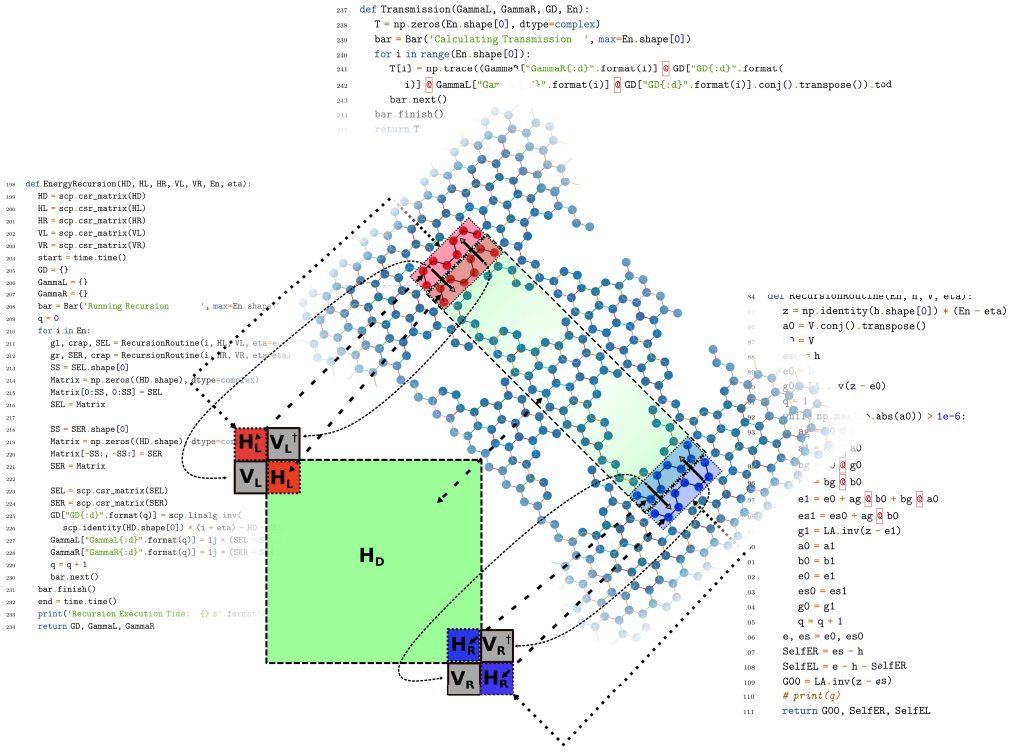
Quantum Transport in Nanoporous Graphene

Rasmus Kronborg Finnemann Wiuff (s163977)* and Christoffer Vendelbo Sørensen (163965)†
Technical University of Denmark‡

Supervisors: Mads Brandbyge and Isaac Alcón
DTU Physics
(-1Dated: June 17th 2019)

Abstract: Normally, both your asses would be dead as fucking fried chicken, but you happen to pull this shit while I'm in a transitional period so I don't wanna kill you, I wanna help you. But I can't give you this case, it don't belong to me. Besides, I've already been through too much shit this morning over this case to hand it over to your dumb ass.[a]



[a] https://slipsum.com/

* E-mail at rwiuff@dtu.dk
† E-mail at chves@dtu.dk
‡ Homepage of the Technical University of Denmark http://www.dtu.dk/english/;
Project Repository: https://github.com/rwiuff/QuantumTransport

Contents

I. INTRODUCTION

In 2018, an article[1], published in Science presented a novel bottom-up approach to synthesise so called nano-porous graphene devices (NPGs). It proved to be ground breaking work that, at the present time, is a new exiting field of research in These devices are made up of single layered graphene with periodic holes (hence the "porous"). The remaining graphene constitutes ribbons called graphene nano ribbons (GNR) and bridges in the structures. Fig. 1 shows how one such structure can look like. Because of graphenes electrical properties [Insert litt], one should be able to finely control the electron currents in the devices and thus create nanometer circuits for use as e.g. chemical detectors. As a result of its novelty, the fabrication of such devices are limited. Before fabrication one must show promising effects through theoretical simulations.

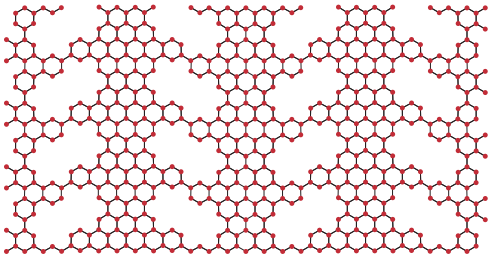


Figure 1: *Drawing of a nanoporous graphene device.*

The aim of this project is to develop numerical tight-binding routines in Python using NumPy. Electron transport can then be simulated using non-equilibrium Green's functions as well as clever recursion algorithms. From here we obtain transmission and band structures for simulated nanodevices.
Generally speaking the community uses DFT-based simulations through tools like those from the SIESTA project (TBtrans). Results are then analysed using SISL[2]. The DFT results can then be extrapolated to larger scales[3]. However DFT programs are complicated and might seem as a blackbox. To get a better understanding of electron transport, we avoid DFT and rely solely on tight-binding simulations. We then confirm the validity of the developed products by comparing result to those from SIESTA.
To summarise:

1. Apply quantum mechanics for electron transport in NPGs.
2. Develop numerical methods (using recursion algorithms, linear algebra) with NumPy to implement tight-binding.
3. Calculate band structures and transmission plots for various devices.
4. Gather single-particle Green's functions and LDOS of said devices.
5. Compare the obtained results and discuss whether or not they sufficiently ressemble DFT based simulations.

The report is organised on the following way:

1. Sections II to V deals with the development of the scripts. First by introduction of basic theoretical concepts, followed by how these concepts are implemented practically through programming.
2. Section VI deals with the generated results of calculations on various NPGs and the comparison with DFT calculations done on the same systems.
3. Section VII summarises the results and concludes the project.

The code repository (which also includes the LaTeX files for this report) can be found on Github: https://github.com/rwiuff/QuantumTransport

II. QUANTUM TRANSPORT

In this section, the basics of the tight-binding approximation for electron transport will be explained. This motivates the use of numerical routines using NumPy.

A. Ballistic quantum transport

As graphene is a two dimensional material that consists of carbon atoms arranged in a hexagonal pattern. Features in such a material can approach nanometer and sub nanometer scales. Because of the small scale the electrical properties of the material is vastly different from normal materials. Usually when describing the electrical properties of a material, drift-diffusion current models are used. They describe electric charges per area and current per area. This is usually a good description in systems where electron-electron and electron-atom scattering frequently occurs. The distance an electron travels before such a event is called its *mean free path*. However, in small systems as those of NPG-devices, the mean free path can be longer than the system itself. Experiments have shown that electrons can move ballistically in graphene and carbon nanotubes[litt], that is, without phonon scattering. Therefore, we model electron transport in NPG using the *ballistic model*. In this model the electrons move through the material as waves. The fact that the electrons moves as waves will prove important later on because it gives rise to *Quantum Interference* which can be exploited as a tool when engineering graphene-based devices[4]. Furthermore the model looks at only one electron at a time in the presence of an electron gas. In addition, the work of this project considers the material in question to be strictly planar, so strictly 2D. The ballistic model has been used with big success for regular graphene and it seems that it also gives a good approximation for NPGs.

B. π-orbitals and π-electrons

When modelling the electron transport in graphene one needs to address the orbital structure of carbon lattices. The orbital structure is exactly what motivate the use of tight binding approximation and Green's functions. The two concepts of tight-binding approximation and Green's functions will be elaborated further in the coming sections. In its basic form graphene can be divided into rings of carbon atoms as shown in Fig. 2a. In the (x,y)-plane the carbon atoms are bound in sp^2 orbitals as shown in Fig. 2b. This hybridisation lock all but one valence electron for the carbon atoms. These electrons exists in a p-orbital in the z-direction. Fig. 3 shows the valence orbitals of carbon.

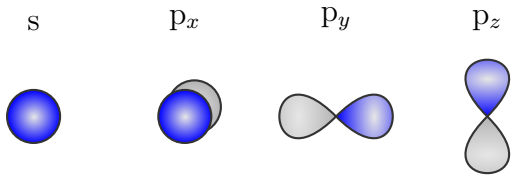
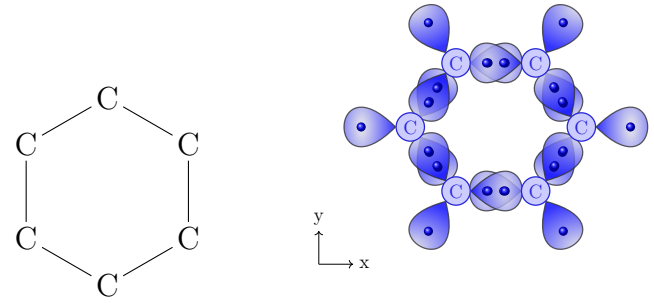


Figure 3: *The valence orbitals of carbon.*

The last electron in the p_z orbital does not mix with the tightly bound s, p_x and p_y electrons and moves freely. Thus these electrons have higher energies compared to the sp^2 electrons and occupy states at the Fermi level. These electrons dominates transport in the graphene lattice. The p_z orbital is also known as the π-orbital and as such the electron lying



(a) *Graphene lattices consists of hexagonal arrangements of carbon atoms.*

(b) *Carbon atoms in a hexagonal lattice are sp^2 hybridised in the (x,y)-plane.*

Figure 2: *Benzene ring and its sp^2 hybradised orbitals.*

there is called a π-electron. Through a carbon lattice the π-electrons will travel through π-orbitals. For a benzene ring the π-electrons at the highest occupied molecular state will travel through the π-orbitals switching sign as they travel as shown in Fig. 4.

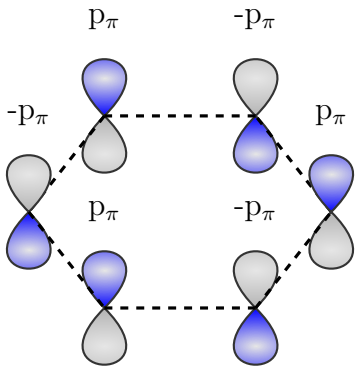


Figure 4: *When jumping from one carbon atom to another, the π-electron goes between p_π-orbitals. Such a jump is described by two matrix elements in the system's Hamiltonian.*

C. Tight-binding

Now that the transport carrying electrons are defined the next step is describing the transport itself. For this purpose we employ the *tight-binding* approximation. In this approximation the electrons are considered being tightly bound to the atoms. Contrary to a free electron gas approximation, the electrons does not spend time in between orbitals, but jump from orbital in atom a to orbital in atom b. The Hamiltonian is represented as a matrix of hopping elements for a collection of neighbouring atomic orbitals, i.e. molecular orbitals, as well as the energy contained within each orbital (which will be addressed later on). This can be done by describing the orbitals as a Linear Combination of Atomic Orbitals (LCAO). The solution to the Schrödinger equation is then:

$$\Psi_{\mathrm{MO}} = \sum_{\alpha,R} c_{\alpha,R}\phi_\alpha(R) \tag{II.1}$$

where $\phi_\alpha(R)$ is an atomic orbital at position R, with α denoting the valence of the orbital ($2s$,$2p_x$,$2p_y$,$2p_z$). In electron transport the states close to the Fermi level is of interest. These are namely the highest occupied molecular orbitals (HOMO), or the lowest unoccupied molecular orbitals (LUMO). As stated earlier only the π-electrons is then of interest. The electrons' motion can be described with the hopping matrix of elements:

$$V_{pp\pi} = \langle \phi_\pi(1) | \hat{H} | \phi_\pi(2) \rangle \tag{II.2}$$

Physically this means that there is a potential ($V_{pp\pi}$) between the π orbitals of neighbouring atoms 1 and 2. In our tight-binding approximation we consider only hop between nearest neighbours. Furthermore we do not take account for out of plane carbon atoms. The element

$$\epsilon_0 = \langle \phi_\pi(1) | \hat{H} | \phi_\pi(1) \rangle \tag{II.3}$$

is the average energy of the electron on atom 1 and, it is common to define the hopping energy relative to this, i.e. $\epsilon_0 = 0$. If the atoms or their environment differs, so does the on-site potential.
Appendix A contains an illuminating example of how the tight-binding approximation can be used to describe simple carbon systems.

III. HAMILTONIAN FOR PERIODIC SYSTEMS

In the following sections, the focus will be to present and explain how to calculate band structures, local density of states and transmission through Python-programming, using the tight-binding approximation for any periodic structure. For simplicity, all initial examples and calculations will be done on a simple system, to make sure the different steps are easy to follow. The system can be seen in Fig. 5. Note how the atoms have indices according to their position. The device consist of a left (red) contact, a right (blue) contact and the atoms in between (yellow). Around the device one finds the semi-infinite chain of atoms.

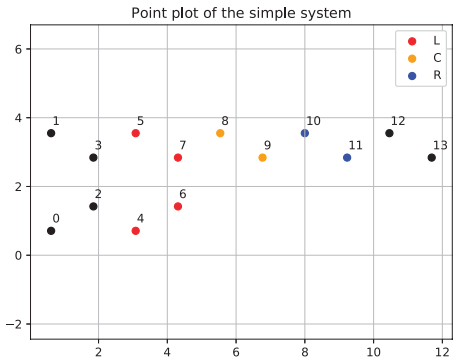


Figure 5: *Figure showing the simple system. The atoms 4-11 is the device itself while the black atoms are the semi-infinite chain of atoms around the device.*

A. Creating the on-site Hamiltonian and hopping matrices

The first and most essential parts needed for calculations is the *on-site* Hamiltonian $\mathbf{h_0}$ and the *hopping* matrices $\mathbf{V}$, $\mathbf{V}^\dagger$. The starting point is a matrix, containing a set of coordinates x_0,y_0,z_0, representing atom positions and a set of unit vectors $\mathbf{u}_x$, $\mathbf{u}_y$, $\mathbf{u}_z$. The unit vectors will be the basis of the unit cell containing all atom coordinates. From now on, only the x and y coordinates will be considered as graphene is considered purely a 2D material. The on-site Hamiltonian represents the interaction of atoms within the unit cell and the hopping matrices represents the interaction of atoms between periodically repeated unit cells. In Fig. 6 a visual representation of the simple systems with periodically repeated unit cells can be seen. For the rest of the report, the on-site Hamiltonian will all ways be the centre cell while the hopping matrices will be the cells surrounding the on-site Hamiltonian

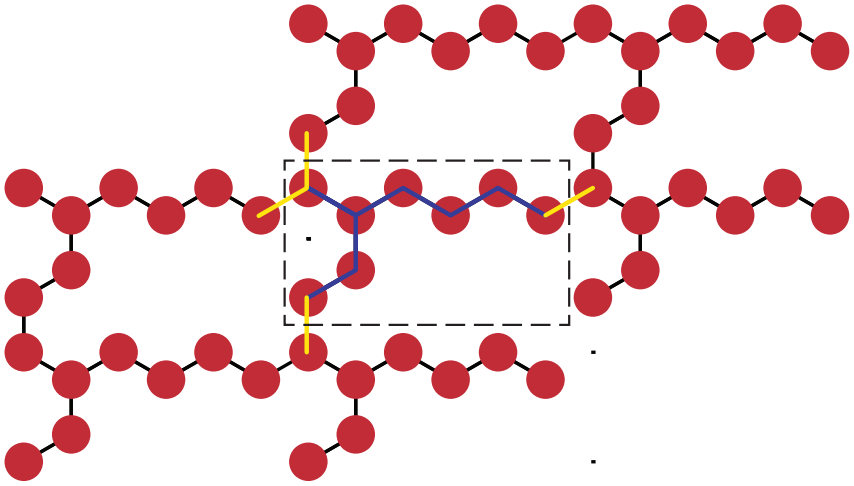


Figure 6: *Visual representation of the periodic simple system. The atoms surrounded by the black box in the centre represents the unit cell. The neighbouring boxes are unit cells repeated periodically once in every direction. Blue connections are onsite hops. Yellow are offsite hops.*

(See Fig. 7 for a generalised visual representation of the concept).
In order to find the hops within the cell, we start off with a set of coordinates for the unit cell atoms and their lattice unit vectors. In order to get the on-site hop matrix we use what is called the vector outer product. This means that every coordinate set (x,y) is compared to every other coordinate set. In our case, we want to find the nearest neighbours. Every two atoms within a distance of 1.60 Å is considered nearest neighbours. A function called *Onsite* have been developed to do just this. In Listing 1 the function can be seen.

```
33     h = np.zeros((xyz.shape[0], xyz.shape[0]))
34     for i in range(xyz.shape[0]):
35         for j in range(xyz.shape[0]):
36             h[i, j] = LA.norm(np.subtract(xyz[i], xyz[j]))
37     h = np.where(h < 1.6, Vppi, 0)
38     h = np.subtract(h, Vppi * np.identity(xyz.shape[0]))
```

Listing 1: *The outer operator in numpy is manifested as two nested loops. On lines 34-36 each atomic distance is calculated. Line 37 replaces all nearest neighbour distances with an input potential, leaving the rest as zero. Lastly the diagonal is subtracted from the matrix on line 38.*

The function produces a matrix which contain all distances between all atoms in the unit cell. The tight-binding model dictates that only atoms with a specific inter-atomic distance interact. Therefore the function has implemented a threshold (Listing 1 line 37) to determine whether a given distance is too great for interaction or small enough for interaction. All distances above the threshold will be changed to a 0-element in the on-site Hamiltonian matrix, representing zero interaction and all distances below the threshold will be changed to 1 to represent interaction between atoms. Finally The on-site Hamiltonian is multiplied with a on-site potential (scalar). The on-site potential $V_{pp\pi}$ differs depending on the system. Lastly the diagonal is subtracted as these elements represent the electrons average energy on each site (The atoms does not interact with themselves). Remember that $\epsilon_0 = 0$. Now the on-site Hamiltonian is complete and the product is a matrix containing 0's and 1's to represent interaction between atoms in a unit cell.
Moving on to the hopping matrices one first has to realise that the interactions are happening in a 2D plane. This has to be kept in mind when describing interaction between unit cells

repeated in all directions in the plane. Effectively this means that six hopping matrices should be created. One in the x-direction, one in the y-direction, one in the xy-direction and their hermitian conjugates. Graphically this corresponds to a structure of this kind:

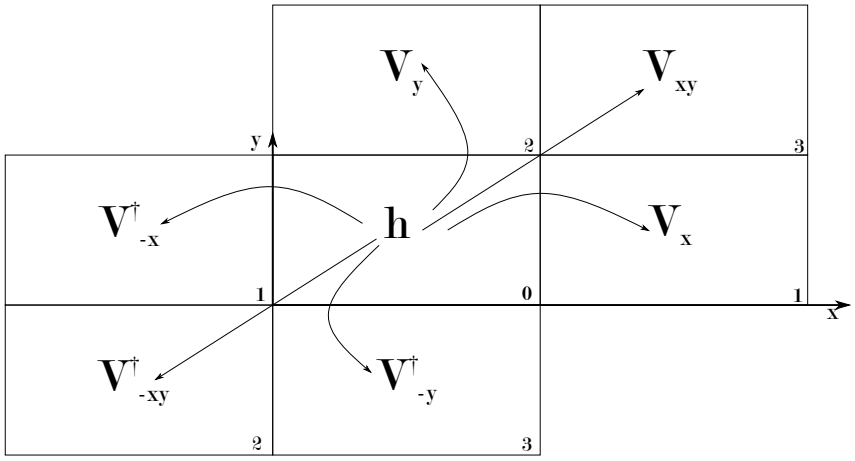


Figure 7: *Representative figure of how the on-site Hamiltonian along with its hopping matrices are structured*

In practice this is done by shifting the original x_0,y_0 coordinates by the given unit vectors $\mathbf{u}_x$,$\mathbf{u}_y$. By addition of the unit vectors to the original coordinate matrix one can get three new coordinate matrices $\mathbf{xy}_{shift-x} = x_0, y_0 + \mathbf{u}_x$, $\mathbf{xy}_{shift-y} = x_0, y_0 + \mathbf{u}_y$ and $\mathbf{xy}_{shift-xy} = x_0$,$y_0$+$\mathbf{u}_x$+$\mathbf{u}_y$. With these three matrices what follows is basically the same method used to get the on-site Hamiltonian. The only difference being that it will be distances between atoms in the on-site Hamiltonian and the shifted matrices respectively. That way it is the distance, and thus the interaction, between the on-site Hamiltonian and the repeated unit cells that is calculated. The three resulting hopping matrices are denoted $\mathbf{V}_x$, $\mathbf{V}_y$ and $\mathbf{V}_{xy}$. They represent interaction (hopping) in the "forward" direction (left-to-right) (See Fig. 7). To create the hopping matrices hopping in the "backwards" (right-to-left) direction (See Fig. 7) one simply has to transpose the hopping matrices. These matrices are denoted with a dagger: $\mathbf{V}_x^\dagger$, $\mathbf{V}_y^\dagger$ and $\mathbf{V}_{xy}^\dagger$. To get an idea of how such matrices looks like, see Fig. 8, where the resulting matrix-maps from a small 12-atom graphene unit cell are shown. It has been stitched together like in Fig. 7.

B. Defining the full Hamiltonian and solving the Schrödinger equation

Now that the on-site Hamiltonian along with its hopping matrices have been created, the next step is to create the full Hamiltonian in order to solve the Schrödinger equation for the system as a whole. This is an eigen-value/vector problem. The Schrödinger equation needs to be solved to get the eigen energies for the system as they will be used to produce band structure plots later on. In essence the full Hamiltonian denoted $\mathbf{H}$ is a sum of the on-site Hamiltonian and its corresponding hopping matrices multiplied by a complex exponential function that has the appropriate phase relative to the hopping matrix. Remember that the electrons moves as waves in the periodic system. Because of the periodicity it is practical to use the reciprocal space. Here the electron can move from k-point to k-point, where k represents a continuous variable between 0 and π along the $\frac{1}{x}$- and $\frac{1}{y}$-axis (hence the term "reciprocal space"). This means that when moving from one k-point to another, one needs to

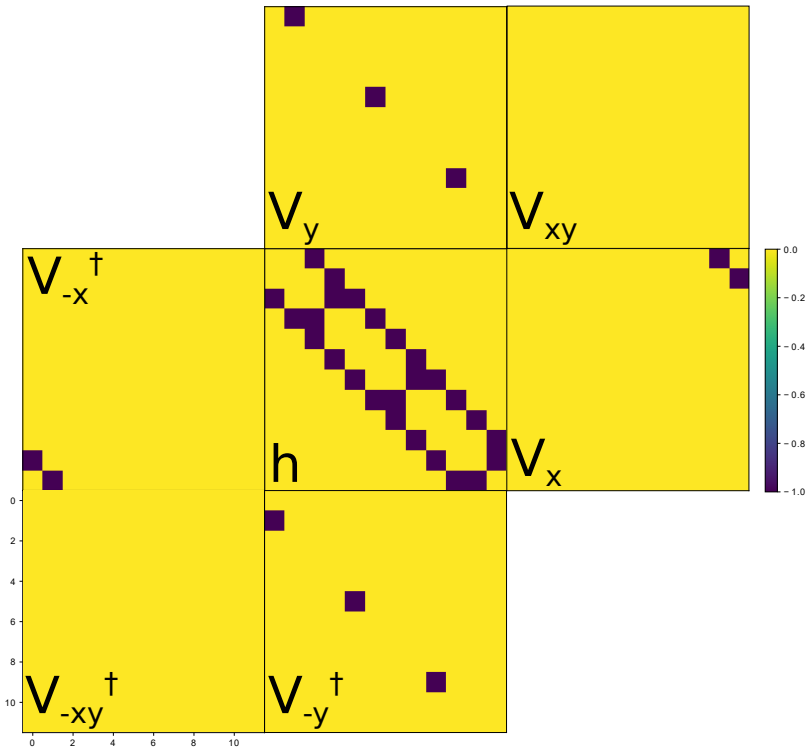


Figure 8: *Matrix maps from calculation on an arbitrary graphene system with a unit cell of 12 atoms. The on-site Hamiltonian along with all its hopping matrices are stitched together like in figure Fig. 7. All the dark spots represent a hopping of an electron to its nearest neighbour i.e. a 1 element and yellow represents a 0 element*

multiply the appropriate parts with a Bloch phase[5]:

$$\begin{aligned}\mathbf{H}(k_x,k_y) = \mathbf{h}_0 + (&\mathbf{V}_x e^{-ik_x} + \mathbf{V}_x^\dagger e^{ik_x} \\ &+ \mathbf{V}_y e^{-ik_y} + \mathbf{V}_y^\dagger e^{ik_y} \\ &+ \mathbf{V}_{xy} e^{-ik_x} e^{-ik_y} + \mathbf{V}_{xy}^\dagger e^{ik_x} e^{ik_y})\end{aligned} \tag{III.1}$$

Using the full Hamiltonian, the Schrödinger equation can be solved

$$\mathbf{H}(k_x,k_y)\boldsymbol{\phi}_k = \boldsymbol{\epsilon}_n(k_x,k_y)\boldsymbol{\phi}_k \tag{III.2}$$

Where $\boldsymbol{\phi}_k$ is the electron wavefunctions and $\boldsymbol{\epsilon}(k_x,k_y)$ is the eigenenergies.
In practice this is done by defining a function, here called *Hkay*, that takes the on-site Hamiltonian, the hopping matrices, and $k_{x/y}$ as inputs and outputs the eigenvalues. Using numpy's *numpy.linalg.eigh* the Hamiltonian matrix is diagonalised. The number of eigenvalues in the output corresponds to the dimension of the full Hamiltonian. In Listing 2 the code for the function is shown.

```
73    Ham = Ham + (V1 * np.exp(-1.0j * x)
74             + np.transpose(V1) * np.exp(1.0j * x)
75             + V2 * np.exp(-1.0j * y)
76             + np.transpose(V2) * np.exp(1.0j * y)
77             + V3 * np.exp(-1.0j * x) * np.exp(-1.0j * y)
78             + np.transpose(V3) * np.exp(1.0j * x) * np.exp(1.0j * y))
```

```
79    e = LA.eigh(Ham)[0]
80    v = LA.eigh(Ham)[1]
```

Listing 2: *Function producing the full Hamiltonian, corresponding to Eq. (III.1) the inputs x and y corresponds to the k_x,k_y .*

C. Producing band structures

In order to calculate and visualise the band structure of the simple system, one need to define the full Hamiltonian $\mathbf{H}$ in two directions. When working with band structures and periodic systems it is common to note points in space with respect to the *Brillouin Zone* which is a primitive cell in reciprocal space. Therefor a continuous variable k is introduced. it extends in two directions in $(-k_x)$ and (k_y), which correspond to lengths between the symmetry points X, Γ and Y in the Brillouin zone. Here Γ is the origin (0,0). Practically this corresponds to making two plots, one for each pair of symmetry points. The y-values in each plot correspond to the eigenvalues obtained by the *Hkay* function described in Section III B. The number of eigen energies, and effectively the number of bands in the plot is dictated by the dimensions of the Hamiltonian, which again is dictated by the number of unit cell atoms. n-atoms $\rightarrow n \times n$-matrix $\rightarrow n$ eigenenergies (bands). However plots produced in this report will only show a few of these bands in a small energy range. In the case of the simple system, the full Hamiltonian for obtaining the eigen energies that corresponds to directions X and Y are:

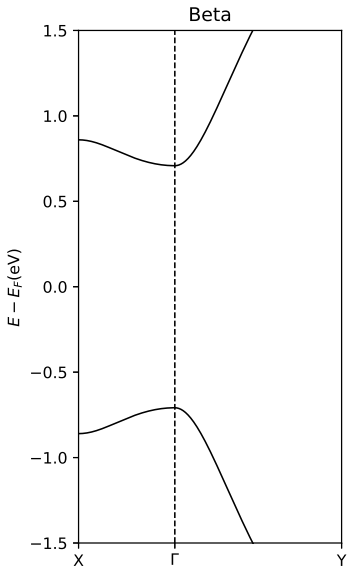


Figure 9: *Figure showing the band structure of the simple system.*

$$X: \mathbf{H}_X = \mathbf{h}_0 + (\mathbf{V}_x e^{ik_x} + \mathbf{V}_x^\dagger e^{-ik_x} + \mathbf{V}_y + \mathbf{V}_y^\dagger + \mathbf{V}_{xy} e^{ik_x} + \mathbf{V}_{xy}^\dagger e^{-ik_x}) \tag{III.3}$$

$$Y: \mathbf{H}_Y = \mathbf{h}_0 + (\mathbf{V}_x + \mathbf{V}_x^\dagger + \mathbf{V}_y e^{-ik_y} + \mathbf{V}_y^\dagger e^{ik_y} + \mathbf{V}_{xy} e^{-ik_y} + \mathbf{V}_{xy}^\dagger e^{ik_y}) \tag{III.4}$$

Using the eigenvalues as y-values in the two plots, putting the two plots together will yield a final plot of the band structure shown in Fig. 9.

IV. GREENS FUNCTIONS, SELF-ENERGY AND THE RECURSION ROUTINE

The Green's function and self-energies play the central role when it comes to obtaining the Local Density of States (LDOS) as well as electron transport in a system. In fact, the imaginary part of the Green's function is the LDOS for a specific site in a system. What the Green's function and self-energy actually is and how they come about will here be explained formally, to motivate the practical use in the following sections.

A. Green's functions and self-energy

Some of the concepts in this section will be explained using the simle system as an example (Fig. 5). Imagine a system like the one in Fig. 6. It contains a unit cell in the centre, marked

by a black border, surrounded by repeated unit cells in all directions. The aim is to explain how electrons move through this region. Suppose all cells surrounding the centre cell are considered "contacts" in the sense that they represent a semi-infinite chain of molecules and that they are the source of electrons (or states) that is injected in to the centre cell. What the Green's function is doing is that it "takes the states through" the centre region. It propagates the states in this particular area. In other words, the Green's function is the solution to the Schrödinger Equation in this area and the equation has the form

$$[(E+i\eta)\mathbf{1}-\mathbf{H}]\mathbf{G}(E)=\mathbf{1} \tag{IV.1}$$

Where η is a small number ensuring that the equation does not diverge when we use the numerical recursion routine described later on. E is the energy for which we are probing the system. From this equation one can also get the Green's function as

$$\mathbf{G}(E)=\mathbf{1}([(E+i\eta)\mathbf{1}-\mathbf{H}])^{-1} \tag{IV.2}$$
$$=[(E+i\eta)\mathbf{1}-\mathbf{H}]^{-1} \tag{IV.3}$$

The Green's functions in these equations are represented as matrices that contain all the individual Green's functions for the unit cell as well a the Green's functions for the rest of the chain. As seen in the equations, all that is needed to get the Green's function for a unit cell, in theory, is an energy and the Hamiltonian of the unit cell. Note that the solution to the Green's function matrix is a diagonal matrix with the two first off diagonals. This is because of rules for nearest neighbour interaction dictated by the tight-binding approximation. As the Green's functions for all unit cells in a potentially semi-infinite system are needed, in practice, one has to turn to more sophisticated methods to obtain all the Green's functions, namely recursion. More on that shortly. For now this is the introduction to the Green's function. How it relates to a unit cell in a system and that it is the source of the LDOS in a unit cell.

As described one can use the Green's functions to get the propagation of states through a specific on-site Hamiltonian. However, if the system contains a range of cells, possibly infinitely many, the Hamiltonian would be of infinite size and the inversion in Eq. (IV.2) would be impossible to do practically. The solution to this, is to model a semi-infinite tight binding chain of atom/molecules and then use *recursion* on this chain. The way the recursion is done is to remove every second cell in the chain. Because the chain is semi-infinite, the yield would just be a new semi-infinite chain. Continuing this way the system can be reduced to a finite size which can actually be inverted. Say one continues to remove every second element in the chain, then in the end, the cells would be too far apart to interact an no hopping between cells would occur. At this point the recursion should stop. More on how this is done practically later. For now one just have to keep in mind that the removing cells in the chain effectively changes to coupling between them and this is where *self-energy* comes in. The self-energy is what describes the effective coupling between a cell and the rest of the semi-infinite chain. And it can be derived by looking at a cell at the very end of the semi-infinite chain and see how it couples to the rest. First one needs the Green's functions. The Green's matrix for this single cell would be given by the equation in Eq. (IV.2). This is before when only one cell and thus one matrix had to be considered. But now, there is an semi-infinite amount of cells and an semi-infinite amount of matrices to consider. However, the cell in the end of the chain only interacts with the cell next to it and so on. Considering this one can write op an equation equivalent to that of Eq. (IV.1) but as system of matrix

equations for the chain.

$$\begin{pmatrix} z\mathbf{1}-\mathbf{H}_c & -\mathbf{V}^\dagger \\ -\mathbf{V} & (z-\varepsilon')\mathbf{1} \end{pmatrix} \begin{pmatrix} \mathbf{X} & \mathbf{G}_{0c} \\ \mathbf{G}_{c0} & \mathbf{G}_{00} \end{pmatrix} = \begin{pmatrix} \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \end{pmatrix} \tag{IV.4}$$

where ε' is the on-site Hamiltonian of the first cell, z is $E+i\eta$, $\mathbf{G}_{0c/c0}$ is the Green's matrices coupling the cell to the rest of the chain and $\mathbf{X}$ is the Green's matrices for the rest of the chain. This is also assuming one knows the Green's function within the chain $\mathbf{G}_c$ and that the chain has constant hopping and on-site elements $\mathbf{H}_c$,$\mathbf{V}$,$\mathbf{V}^\dagger$. Solving this system for $\mathbf{G}_{00}$ and eliminating $\mathbf{G}_{0c}$, which is unknown, one gets

$$\mathbf{G}_{00}(z) = \left[(z-\varepsilon')-\mathbf{V}(z\mathbf{1}-\mathbf{H}_c)\mathbf{V}^\dagger\right]^{-1} \tag{IV.5}$$
$$= (z-\varepsilon'-\Sigma(z))^{-1} \tag{IV.6}$$

where $\Sigma(z)$ is the self-energy. One can isolate the self energy from the equations above to

$$\mathbf{\Sigma}(z) = \mathbf{V}[z\mathbf{1}-\mathbf{H}_c]^{-1}\mathbf{V}^\dagger \tag{IV.7}$$

So from Eq. (IV.5) it can be seen that the solution to the system of matrices for the Green's matrix **G00** is the same as in Eq. (IV.2) but with a correction (Eq. (IV.7)), which is the self-energy. And that is what describes the coupling for the first cell to the rest of the chain. This concludes the formal introduction to Green's functions and self-energy.

B. Obtaining first cell self-energy and Green's matrix through programming

For simplicity and in order to check whether the routine would yield the expected results, the system in Fig. 5 is use as an example. The goal is to get the Green's functions for the centre unit cell in the semi-infinite chain and the self-energies coupling to rest of the chain right and left. Specifically for the simple system one should imagine first having one centre unit cell like Fig. 5 and then repeating it infinitely in the left and right direction. The fact that there is a left *and* right self energy is that the unit cell lies within the semi-infinite chain and not at the very end as described in Section IV A. To be assured, this does not conflict with any of the preciously mentioned formalism and the left and right self-energies are quite easily obtained as one shall see shortly. As mentioned the goal is to get the Green's functions of a specific unit cell and the self-energies related to it. If the Green's matrix $\mathbf{G}$ represents the whole chain, then the equation of the whole system would be equivalent to that of Eq. (IV.1). Considering the Green's functions for specific unit cell in question, it would correspond to one column in the system of equations, say the first. One can define the on-site Hamiltonian $\mathbf{h}_0$ for the specific unit cell and its hopping matrices $\mathbf{V}$,$\mathbf{V}^\dagger$. The two hopping matrices correspond to hopping left or right in the chain respectively. These can be obtained using the functions already developed in Section III. Throughout this section they will be named $a_0 = \mathbf{V}^\dagger$, $b_0 = \mathbf{V}$, $e_{s0} = \mathbf{h}_s$. The recursion is an iterative process and so the zero index indicates the starting point of the iterations and the s index indicates that it is the Hamiltonian of the specific wanted cell. One can also define a Green's function for a single unit cell as $g_0 = (z-e_0)^{-1}$ just like Eq. (IV.1) where $e_0 = \mathbf{h}$ which is the on-site Hamiltonian of the other cells. With these elements a system of equations, similar to Eq. (IV.1) can be setup. The first difference being that the identity matrix is replaced by its first column, because the solution of interest is that one first column in the Green's matrix. The second is that the first element in the Hamiltonian matrix $\mathbf{H}$ is related to the specific single unit cell $\mathbf{h}_s$. Next a range of multiplications of the different elements stated so far will be shown, and

afterwards it will be explained how these affect the system of equations to give recursion. The multiplications are:

$$\begin{aligned} a_1 &= a_0 \times g_0 \times a_0 \\ b_1 &= b_0 \times g_0 \times b_0 \\ e_1 &= e_0 + a_0 \times g_0 \times b_0 + b_0 \times g_0 \times a_0 \\ e_{1s} &= e1_{0s} + a_0 \times g_0 \times b_0 \\ g_1 &= (z - e_1)^{-1} \end{aligned} \tag{IV.8}$$

These equations constitutes the first iteration in the recursion and they can be repeated indefinitely. In the matrix system of equations these multiplications effectively shifts all elements in the matrix containing on-site Hamiltonians and hopping matrices by one column. Because the matrix is diagonal, it will leave the first column of the matrix empty. The column can then be removed and this is exactly what corresponds to removing a cell in the semi-infinite chain. Keeping on doing these multiplications, raising the index by +1 every time, one can move through the system as a whole, removing of columns (cells) in the system of equations, thus reducing it to a finite size. The loop is shown in Listing 3.

```
92   while np.max(np.abs(a0)) > 1e-6:
93       ag = a0 @ g0
94       a1 = ag @ a0
95       bg = b0 @ g0
96       b1 = bg @ b0
97       e1 = e0 + ag @ b0 + bg @ a0
98       es1 = es0 + ag @ b0
99       g1 = LA.inv(z - e1)
100      a0 = a1
101      b0 = b1
102      e0 = e1
103      es0 = es1
104      g0 = g1
```

Listing 3: *The while loop in the recursion routine. The matrix elements are overwritten with the new variables until the resulting matrix is small enough to invert*

Note that some intermediate multiplications are made e.g. *ag = a0 @ b0*. This is for run-time optimisation only, as these products are used multiple times per iteration. The recursion is run until a threshold is met. The threshold is determined by the value of the hopping matrix a_0. As it reaches a value close to zero, there is no longer any effective interacting (hopping) between the cells because of removal of cells and the recursion should stop. In the end one will obtain re-normalised Hamiltonians and hopping matrices which is then used to get the Green's functions and self-energies through these simple equations:

$$\begin{aligned} \mathbf{\Sigma}_R &= e_s - h \\ \mathbf{\Sigma}_L &= e - h - \mathbf{\Sigma}_R \\ \mathbf{G00} &= (z - e_s)^{-1} \end{aligned} \tag{IV.9}$$

These are calculated out of the Python function's while loop as such:

```
106  e, es = e0, es0
```

```
107     SelfER = es - h
108     SelfEL = e - h - SelfER
109     G00 = LA.inv(z - es)
```

Listing 4: *Self-energies from left and right as well as a normalised Green's functions-matrix are calculated at the end of the recursion loop.*

This concludes how recursion works and how the first cell Green's function as well as the self-energies are obtained.

C. Plotting the real and imaginary part of the first cell Green's function

One of the results possible to obtain via the recursion routine is the Green's function of the centre unit cell in relation to the rest of the chain. As mentioned the imaginary part of the elements Green's matrix is the LDOS of the different sites in the unit cell. With a relatively simple approach, the Green's matrix elements can be obtained as a function of energy, using a *for loop*, looping over a range of energies which is then used as input in the *RecursionRoutine* function (Listing 3), see Listing 5:

```
64  G00 = np.zeros((En.shape[0]), dtype=complex)
65  for i in range(En.shape[0]):
66      G, SelfER, SelfEL = RecursionRoutine(En[i], h, V, eta)
67      G = np.diag(G)
68      G00[i] = G[4]
```

Listing 5: *Code showing the loop which produces the complex Green's function (or y-values) for a range of energies used in the plot.Fig. 10*

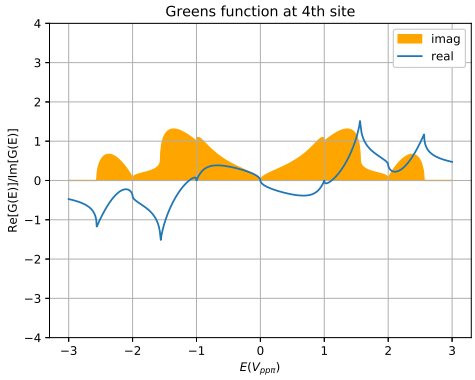


Figure 10: *A plot showing the real and imaginary part of Green's function at the zeroth site resulting from the recursion routine on the simple system. Note that the yellow imaginary part is the representation of the local density of states.*

Taking the imaginary part of the output in Listing 5 gives information about the LDOS at a specific energy and place in space, namely a specific atom in the unit cell. The resulting plot for the simple system (at atom index 4) can be seen in Fig. 10. Note that the plot only represents the LDOS for a specific site on the molecule and that they may change radically from site to site (see Appendix B, Fig. 26 for an example using the same system as Fig. 5). The site can be changed by choosing another index in Listing 5 line 68, which corresponds to the atom indices in Fig. 5.

V. TRANSMISSION ROUTINE

This section will mark the conclusion of the preliminary work done in Sections II to IV. All the functions producing on-site Hamiltonians, hopping matrices, full Hamiltonians, band structures as well as self-energy and Green's functions by recursion, will be used to get the transmission through the material.

A. Left-right device geometry, rate matrices and spectral functions

First of all a sentence as to what transmission is: Transmission is the probability of an electron being transported trough a specific region for a specific range of energies and thus how the region affects the overall current flow of electrons through the system as a whole. Below is an equation stating it formally

$$P(t)_{mn} = \left|\langle m|e^{i\mathbf{H}t/\hbar}|n\rangle\right|^2 \tag{V.1}$$

where m,n is the density of states in each side of the region of interest (states going in/out) and $e^{i\mathbf{H}t/\hbar}$ is the solution to the Green's function.
To give an overview and explain the different concepts of transmission this section will rely heavily on Fig. 11 where all the different parts of the system have been translated from the actual material into mathematical formalism in the shape of matrices. The first and central piece is the so-called "Device Region" with the on-site Hamiltonian $\mathbf{H}_D$ (Green area in Fig. 11). The device region contains at least one central unit cell as well as a "left" and "right" unit cell (Red and blue area in Fig. 11). The left and right unit cells represent the contact region of the device i.e. the two parts that connects to the rest of the system/molecule. They have on-site Hamiltonians $\mathbf{H}_L$,$\mathbf{H}_R$ and they interact with rest of the system via hopping matrices $\mathbf{V}_{L,R}$,$\mathbf{V}^\dagger_{L,R}$. As $\mathbf{H}_D$ contains $\mathbf{H}_{L,R}$ they can be picked out of $\mathbf{H}_D$, without further calculation (See Fig. 11) once the $\mathbf{H}_D$ has been calculated. The cells next to the contact region can be reduced into a single Hamiltonian by recursion to have same dimension as $\mathbf{H}_{L,R}$. Note that $\mathbf{H}_{L,R}$ need not be the same dimensions. Related to $\mathbf{H}_{L,R}$ is the left and right self-energies $\mathbf{\Sigma}_{L,R}$ and on-site Green's matrices $\mathbf{g}_L$, $\mathbf{g}_R$. These can be obtained using the theory and developed methods from Section IV. However the aim is to obtain the Green's matrix for the device region, $\mathbf{G}_D$, as it is the one needed to fully describe the transmission in the region of interest. In other words, propagation of the states in the green area in Fig. 11. To obtain it, one simply has to keep in mind that the effective coupling of the device region to the rest of the system is determined by the two contact regions ($\mathbf{H}_{L,R}$) and thus the correction to the device Green's functions will be determined by the self-energy of those contact regions. So the Green's matrix will be given by the same equation as Eq. (IV.2) but with a self-energy correction going left ad right:

$$\mathbf{G}_D = [\mathbf{1}(E+i\eta) - \mathbf{H}_D - \mathbf{\Sigma}_L(E) - \mathbf{\Sigma}_R(E)]^{-1} \tag{V.2}$$

Looking back at Eq. (V.1) the Green's function needed has been obtained, but what about the states going in and out of the device region $\langle m|$ and $|n\rangle$? As states travel, their corresponding self-energies change in time. Rate operators $\mathbf{\Gamma}$ are defined as the change in imaginary self-energy. They describe the 'rate' by which the self-energies of the states moving in and out, change in time. The rate matrices are given by

$$\mathbf{\Gamma}_{L,R} = i(\mathbf{\Sigma}_{L,R} - \mathbf{\Sigma}^\dagger_{L,R}) \tag{V.3}$$

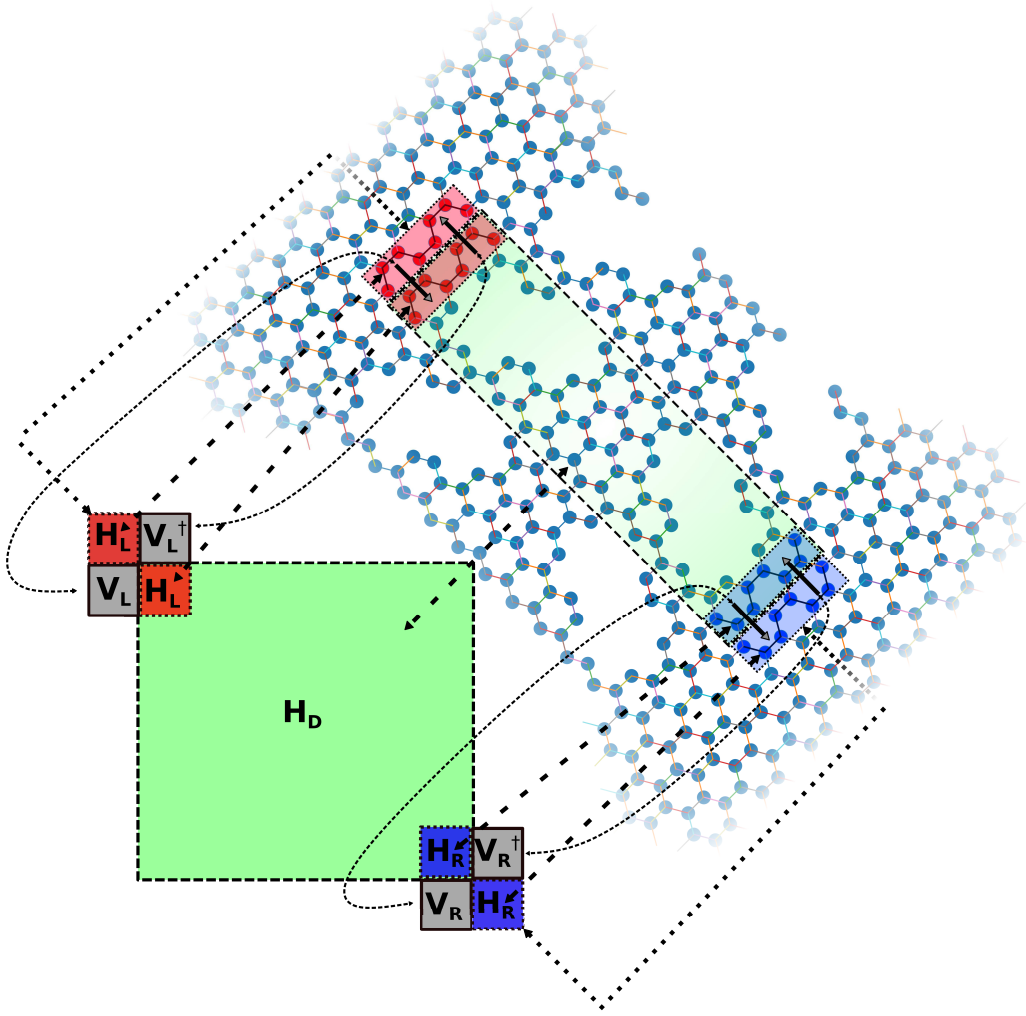


Figure 11: *Illustration showing how the different parts of the system are translated into matrix blocks in NPG. The green box is the unit cell of the device with the Hamiltonian $\boldsymbol{H}_D$. It includes one red and blue box which themselves are unit cells of the left and right contacts and have the Hamiltonians $\boldsymbol{H}_L$, $\boldsymbol{H}_R$. The two other unit cells lying outside the device region represents what could be an infinite contact region reducible by recursion. Finally the two fat black arrows (not dotted) on each side of the device represents the hopping between the device- and contact region. Note that the direction of hopping corresponds to a specific hopping matrix. F.ex. left-to-right is the ordinary hopping matrix ($\mathbf{V}$) while right-to-left is its conjugate ($\mathbf{V}^\dagger$) (for both left and right side of the device).*

Similarly the change in the imaginary Green's matrix is defined as

$$\begin{aligned}\mathbf{A} &= i(\mathbf{G}-\mathbf{G}^\dagger) && \text{(V.4)}\\ &= i(\mathbf{G}^\dagger\left[\mathbf{G}^\dagger-\mathbf{G}\right]^{-1}\mathbf{G})\\ &= i(\mathbf{G}^\dagger[E-\mathbf{H}^\dagger-\boldsymbol{\Sigma}^\dagger-(E-\mathbf{H}-\boldsymbol{\Sigma})]^{-1}\mathbf{G})\\ &= i(\mathbf{G}^\dagger[-\boldsymbol{\Sigma}^\dagger+\boldsymbol{\Sigma}]^{-1}\mathbf{G})\\ &= i(\mathbf{G}^\dagger\boldsymbol{\Gamma}\mathbf{G}) && \text{(V.5)}\end{aligned}$$

Here it have been utilised that the Green's matrix is symmetric in time ($\mathbf{G}=\mathbf{G}^T$, $\mathbf{G}^T\mathbf{G}=\mathbf{I}$, $\mathbf{G}^T=\mathbf{G}^{-1}$) as well as the fact that $\mathbf{G}=[z\mathbf{1}-\mathbf{H}-\boldsymbol{\Sigma}]^{-1}$. Additionally the Hamiltonians

cancel out as they are unitary. The definition of $\mathbf{A}$ is that it is a *spectral function*. It is the change of the imaginary Green's functions. As seen in the equations above it can be described by the Green's matrix and the rate matrix. From the previous sections it is know that the imaginary part of the Green's functions represent the LDOS. The spectral function is thus describing the density of states, changing by the rate $\mathbf{\Gamma}$. The spectral function can also be described in 'left/right' terminology as

$$\mathbf{A}_{L,R} = \mathbf{G}_D^\dagger \mathbf{\Gamma}_{L,R} \mathbf{G}_D \tag{V.6}$$

Taking the left spectral function as an example, it represents the density of states of a wave coming from the left entering the device. To describe how the density of states pass through the device region from left-to-right or right-to-left one just have to multiply the spectral function by the appropriate rate matrix. Again using the left spectral function as an example

$$\mathrm{Tr}[\mathbf{A}_L \mathbf{\Gamma}_R] = \mathrm{Tr}[\mathbf{G}_D^\dagger \mathbf{\Gamma}_L \mathbf{G}_D \mathbf{\Gamma}_R] \tag{V.7}$$

Here the trace of the product is taken as all states besides the one in the left contact region are zero. This corresponds to the density of states coming from the left, which then pass on through the right electrode by the rate $\mathbf{\Gamma}_R(E)$. Additionally the trace is cyclic so

$$\mathrm{Tr}[\mathbf{A}_L \mathbf{\Gamma}_R] = \mathrm{Tr}[\mathbf{A}_R \mathbf{\Gamma}_L] \tag{V.8}$$

This ultimately leads to transmission because transmission is in essence an expression of how much of the density of states passes trough the device and as explained, Eq. (V.7) is exactly the density of states, coming from the left (or right) and then passes through the right (left) by rate $\mathbf{\Gamma}_{L,R}(E)$. So using the Green's function, left/right self-energies as well as the left/right rate matrices, the transmission, as a function of energy, can be obtained via the following equation:

$$T(E) = \mathrm{Tr}[\mathbf{\Gamma}_R \mathbf{G}_D \mathbf{\Gamma}_L \mathbf{G}_D^\dagger](E) \tag{V.9}$$

B. Transmission in 1D

Again the developed routine in this section will be used on the simple system (Fig. 5) as an example and in order to make sure that the obtained results are as expected. Thereafter it will be generalised to suit all kinds and sizes of system. First thing is to define the device in the same manner as Fig. 11 so that the device Hamiltonian $\mathbf{H}_D$ can be obtained through the already defined function *Onsite*. The left and right Hamiltonian $\mathbf{H}_{L,R}$ are thus picked out as described earlier. An implementation has been made to the script to allow the user to see the left and right contact cells graphically as red and blue marked atom indices in the plot of the unit cell (See Fig. 5). This allows the user to get an overview of dimensions of $\mathbf{H}_{L,R}$. From $\mathbf{H}_{L,R}$ the corresponding hopping matrices are defined using the *Hop* function. Then the developed *EnergyRecursion* function is used to obtain the device Green's matrix. This function is a more elaborate version of the earlier mentioned *RecursionRoutine* and it takes $\mathbf{H}_D$,$\mathbf{H}_L$,$\mathbf{H}_R$,$\mathbf{V}_L$,$\mathbf{V}_R$, a range of energies and η as inputs. As these matrices are sparse, it is memory efficient to convert these to compressed sparse row matrices as shown in Listing 6. This is, however, a tradeoff as numerical computations will take more time.

```
199  HD = scp.csr_matrix(HD)
200  HL = scp.csr_matrix(HL)
201  HR = scp.csr_matrix(HR)
```

```
202    VL = scp.csr_matrix(VL)
203    VR = scp.csr_matrix(VR)
```

Listing 6: *By using SciPy, EnergyRecursion converts all matrices to the more memory efficient "Compressed Sparse Row matrix".*

The next step is to calculate the self-energies for the left and right cells ($\mathbf{\Sigma}_{L,R}$) and the Green's functions for the left and right cells ($\mathbf{g}_{L,R}$) using the *RecursionRoutine* as shown in Listing 7.

```
210    for i in En:
211        gl, scrap, SEL = RecursionRoutine(i, HL, VL, eta=eta)
212        gr, SER, scrap = RecursionRoutine(i, HR, VR, eta=eta)
```

Listing 7: *The Green's functions and self-energies are calculated, first from left and then from right. This is done for each energy in a selected energy range (En)*

The device Green's function $\mathbf{G}_D$ as well as the left and right rate matrices $\mathbf{\Gamma}_{L,R}$, are then calculated using Eqs. (V.2) and (V.3), as shown in Listing 8.

```
225        GD["GD{:d}".format(q)] = scp.linalg.inv(
226            scp.identity(HD.shape[0]) * (i + eta) - HD - SEL - SER)
227        GammaL["GammaL{:d}".format(q)] = 1j * (SEL - SEL.conj().transpose())
228        GammaR["GammaR{:d}".format(q)] = 1j * (SER - SER.conj().transpose())
```

Listing 8: *The device' Green's functions and the left and right rate matrices are calculated in EnergyRecursion as shown.*

The output of the *EnergyRecursion* function is the two rate matrices $\mathbf{\Gamma}_{L/R}$ as well as the device Green's function $\mathbf{G}_D$ and as per Eq. (V.9) the matrices needed for transmission have been obtained. As seen in Listing 9 the function *Transmission* simply carries out the matrix product and subsequent trace of the matrices resulting from *EnergyRecursion* and outputs a range of transmission probabilities which is then plotted against an energy range. Do mind that this is still just 1D in the sense that the transmission only moves in one direction. A plot of the transmission for the simple 1D system (the one in Fig. 5) can be seen in Fig. 12.

```
240    for i in range(En.shape[0]):
241        T[i] = np.trace((GammaR["GammaR{:d}".format(i)] @ GD["GD{:d}".format(
242            i)] @ GammaL["GammaL{:d}".format(i)] @ GD["GD{:d}".format(i)].conj(
243        ).transpose()).todense())
```

Listing 9: *Code showing how the transmission probabilities are calculated. Taking rate matrices, device Green's function and a range of energies as inputs, it takes the trace of the matrix product for a range of energies as in Eq.* (V.9).

The transmission-function also transform the transmission and Green's function to normal "dense" matrices, as theese are no longer exessively large, nor sparse.

C. Development of transmission to 2D

Lastly the transmission routine needs to be generalised so it can handle transport in two directions. The most convenient approach is to work with five real space unit cells as a

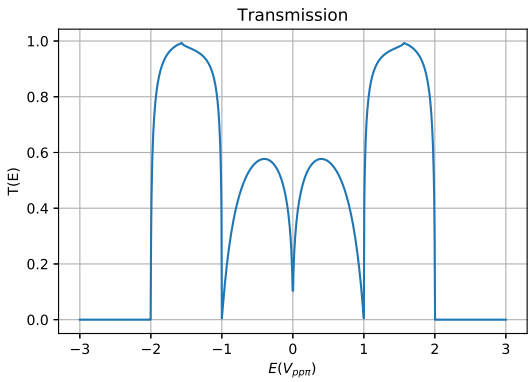


Figure 12: *Figure showing the resulting transmission through the simple system*

starting point. One centre cell, a right and a left cell representing the contacts and then two additional cells on the left and right, representing the rest of the semi-infinite chain. This would be the minimum amount of cells needed to generalise the transmission. One might have more centre cells if the structure changes from one of those cells to the other. First these five unit cells will be defined using already developed tools. Again working with the simple system, Fig. 13 shows such a 2D system. This system contains the 1D semi-infinite chain of

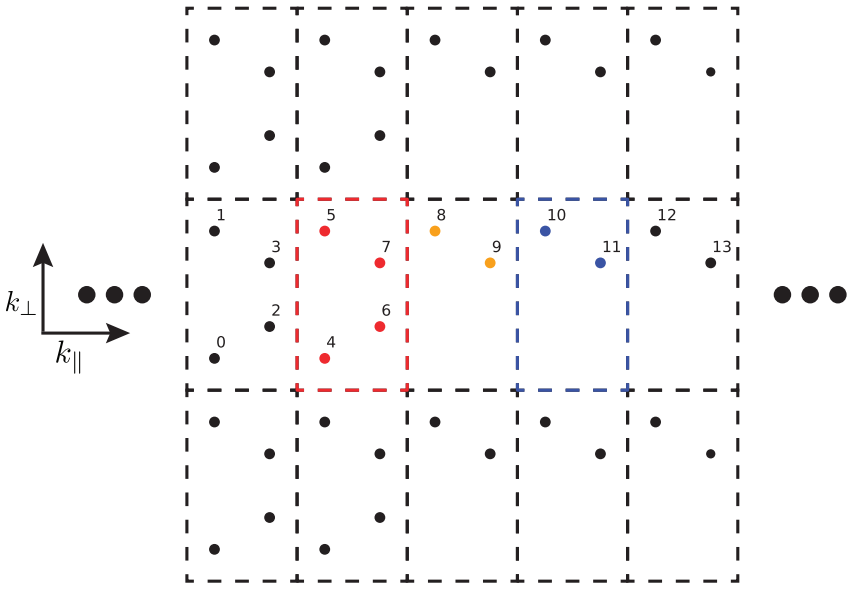


Figure 13: *The simple system repeated in 2D. The k-points perpendicular to the transport direction are shifted with a Bloch phase. This practically packs the system to just one row of cells, which in turn is dealt with using the EnergyRecursion routine.*

cells however, when describing cells in the added direction, one simply needs to shift the system by a phase shift. There are two considerations: Firstly the range of k-points exists between $-\pi$ and π. Secondly these are periodic boundraries. One really needs to pick a range of perpendicular k-points of interest and then the idea is to construct a Hamiltonian for

the 1D chain, located at that point. The recusion routine then calculates Green's functions and transmission. The Hamiltonian is constructed using the function *PeriodicHamiltonian*. This function uses the equation:

$$\mathbf{H} = \mathbf{h} + \mathbf{V}e^{1ik_\perp} + \mathbf{V}^\dagger e^{1i-k_\perp} \tag{V.10}$$

The function is shown in Listing 10.

```
250    h, p = Onsite(xyz=xyz, Vppi=-1, f=1)
251    V = Hop(xyz=xyz, xyz1=xyz + np.array([0, UY, 0]), Vppi=-1)
252    print('Number of hopping elements: {}'.format(np.sum(np.abs(V))))
253    Ham = h + V * np.exp(1j * i) + np.transpose(V) * np.exp(-1j * i)
```

Listing 10: *The function calculating the periodic Hamiltonian for a given k-point using Eq. (V.10).*

As seen in Listing 10, the onsite and hop matrices are calculated in the usual way but for all of the atoms in the 5 cells. In order to understand why, first consider Fig. 14. Because of

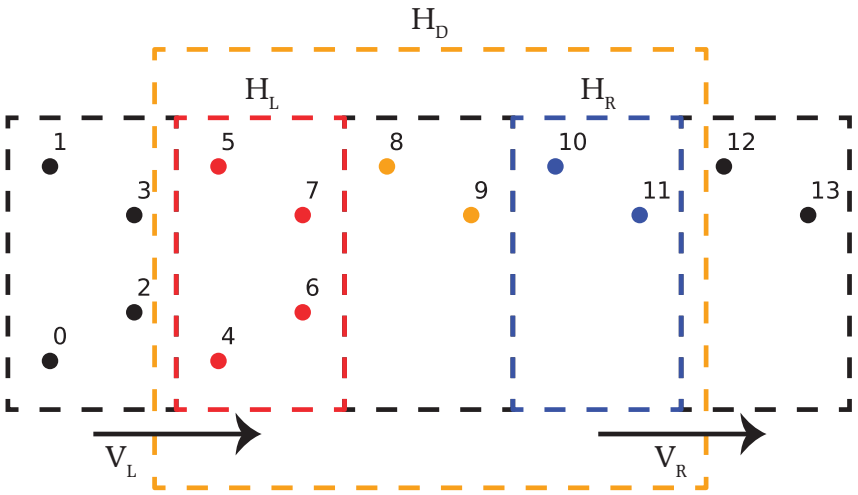


Figure 14: *All onsite on offsite hop elements and contact Hamiltonians can be extracted from the matrix produced by PeriodicHamiltonian because of the 5-cell setup. H_D is the device Hamiltonian, H_L and H_R are contact Hamiltonians; V_L and V_R are hopping matrices.*

the 5-cell setup it is possible to extract all hop elements and Hamiltonians from the output matrix of *PeriodicHamiltonian*. The contact Hamiltonians exists in the ends of the diagonal and some off diagonal elements contains the hop elements. These are extracted in the 2D transmission routine as shown in Listing 11.

```
39    Ham = PeriodicHamiltonian(xyz, UY, i)
40    HL = Ham[L]
41    HL = HL[:, L]
42    HR = Ham[R]
43    HR = HR[:, R]
44    VL = Ham[L]
45    VL = VL[:, RestL]
```

```
46    VR = Ham[RestR]
47    VR = VR[:, R]
```

Listing 11: *All required matrices for the recursion routine can be picked from the periodic Hamiltonian.*

To get the transmission in 2D the function *PeriodicHamiltonian* is nested in a for loop, looping over transverse k-points. Still within the loop the *EnergyRecursion* is used to get the device Greens function, and left/right rate matrices. Lastly the transmission is calculated with the function *Transmission* using Eq. (V.9). Plots for transmission per k-point will be shown for NPG in Section V E.

D. Summary of developed scripts

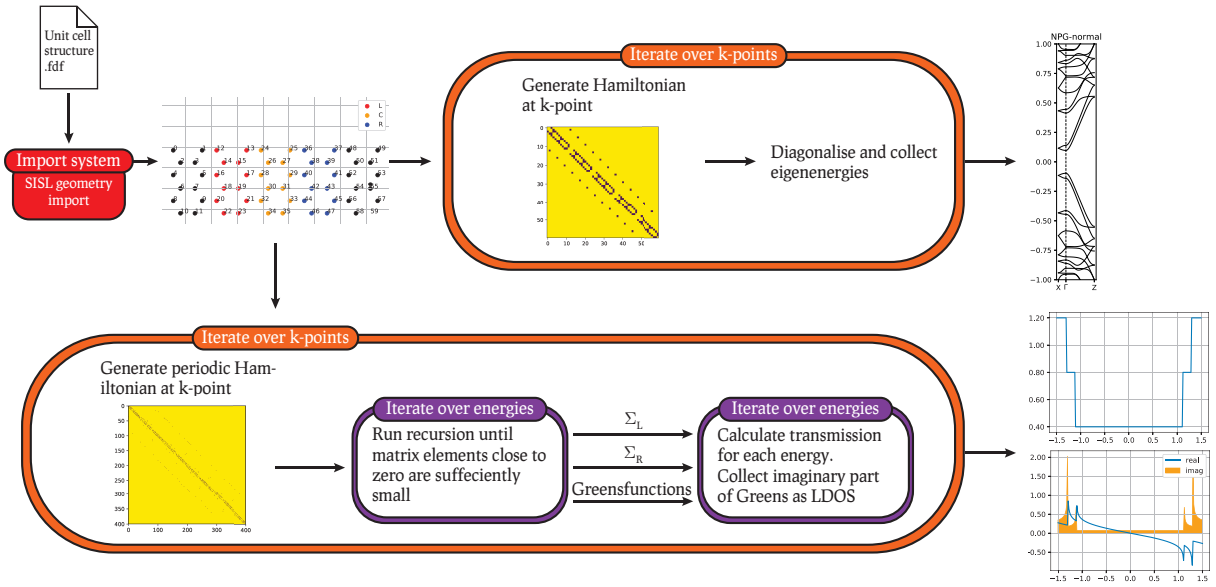


Figure 15: *Flowchart depicting the routines run in python. SISL is used to import the geometry. Afterwards the coordinates are either used for band structure plots or for transmission plots. For the band structures, a Hamiltonian at each desired k-point is generated and diagonalised in order to get the eigen energies. These energies are then plotted. With transmission, a periodic Hamiltonian at various transverse k-points are generated and reduced to self-energies in the transport direction (using the recursion algorithm). The self-energies and the Green's functions retrieved here are then multiplied as to get the transmission.*

The final algorithm is shown in Fig. 15. It is laid our here:

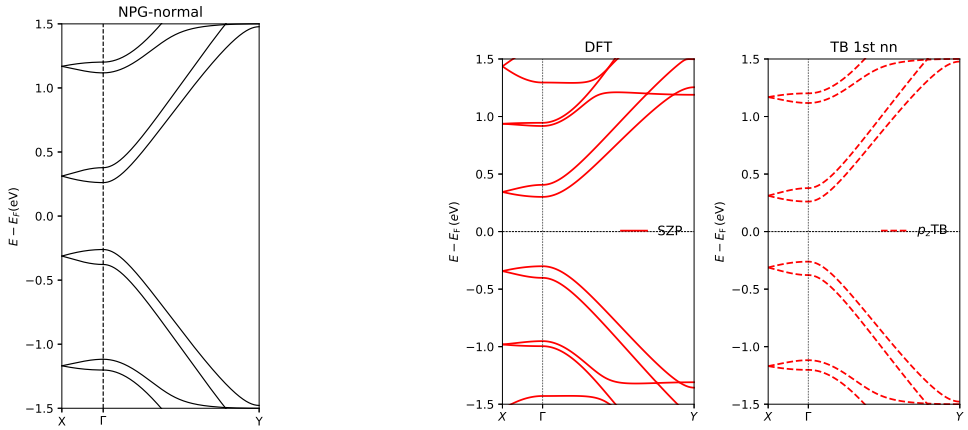
1. Import geometry and unit vectors with SISL.
2. For bandstructure plots
   (a) Find the Hamiltonian described in Section III
   (b) Calculate the eigen energies, iterating over the desired k-points.
   (c) Plot the gathered k-points.
3. For transmission plots.
   (a) Hamiltonian at a specific $k_\perp$-point is generated.
   (b) Iterating over the energies, the (at least 5-cell) system is reduced using the recursion routine.

(c) Iterating over the energies, the self-energies and Green's functions are used to calculate transmission.

(d) Transmission, Green's functions and LDOS are plottet.

E. Comparing Tight Binding with DFT and TBtrans for transmission and band structure calculations in NPG

All the scripts necessary for calculation have been developed and they can now be used on a system of NPG. The following results are based on this structure similar to that of Fig. 1. Firstly the band structure obtained using the script described in Section III B is shown together with band plots obtained from DTF[] and TBtrans calculations.

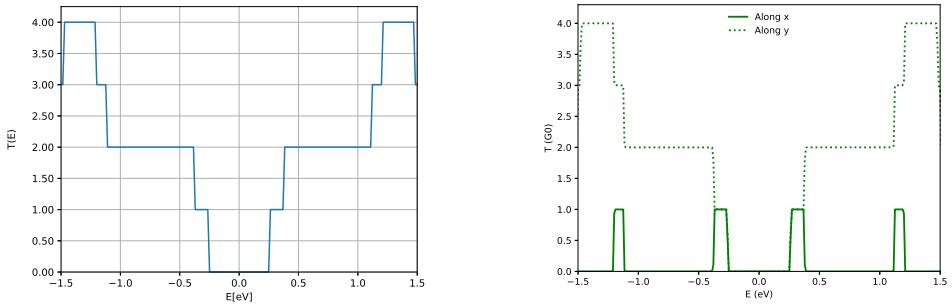


(a) *Figure showing the band structure for NPG with normal bridges obtained with the script described in Section III B*

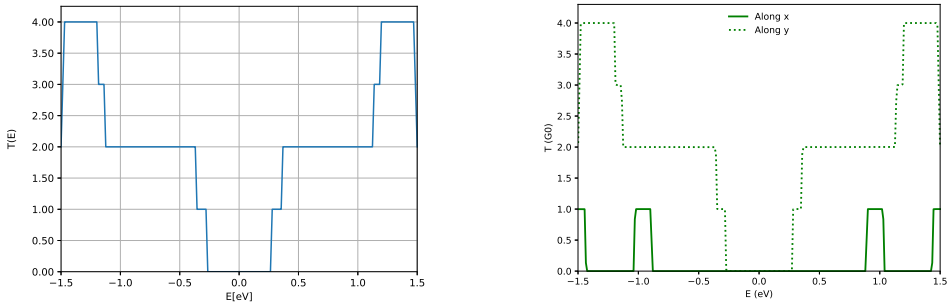
(b) *Figure showing the bands structures obtained from DFT and TBtrans calculations.*

Figure 16: *Figure showing how the band plots compare for DFT, TBtrans and the developed script.*

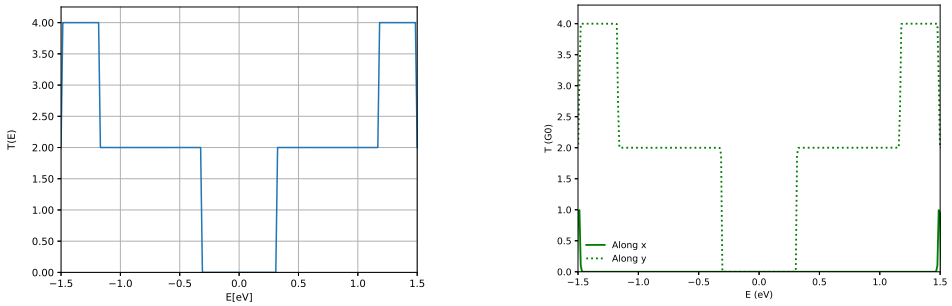
The band plot obtained in Fig. 16a shows almost 1-to-1 correspondence with the plot obtained using TBtrans Fig. 16b (right) and is also very similar to the plot obtained from DFT Fig. 16b (left), proving that the script is capable of creating band structures for NPG-systems. Next is a comparison of the transmission in NPG for different k-points in reciprocal space. The plots are made for transmission through NPG in real space (x-direction) for each three different k-points in the reciprocal space ($\frac{1}{y}$-direction). The three k-points are 0, $\frac{\pi}{2}$, π. Additionally an average over these k-points is plotted as well. In Fig. 17 transmission plots obtained with the script described in Section V C is compared with transmission plot obtained through DTF, using the same k-points.



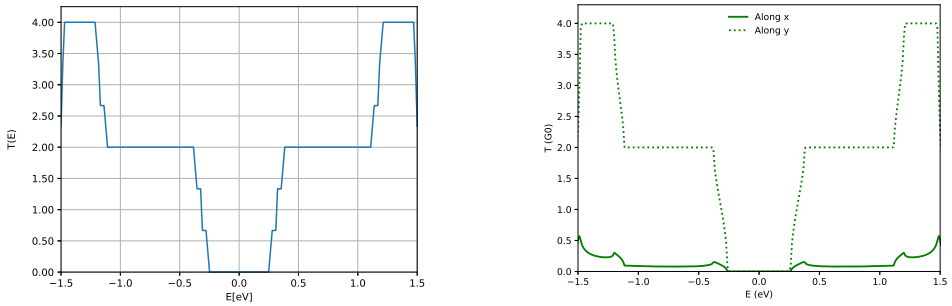
(a) *Transverse k-point 0*



(b) *Transverse k-point* $\frac{\pi}{2}$



(c) *Transverse k-point* π



(d) *Average over k-points* $0, \frac{\pi}{2}, \pi$

Figure 17: *Figure showing the comparison between plots of transmission through NPG obtained using the developed scripts (left/blue) and obtained using DFT (right/green).*

As one can see in Fig. 17 The plots show very good correspondence with the DFT calculation. Again proving that the scripts developed on the basis of the Tight Binding approximation

can produce valid results, when it comes to band structure and transmission through NPG. This concludes the preliminary work. The tools for calculation of band structures, LDOS and transmission has been successfully developed, using *python*-programming and the Tight Binding approximation. Following will be a range of tests on different NPG systems, using the developed tools.

VI. EXPLORING FUNCTIONALITY OF GNR BRIDGES

In this section a range of tests will be conducted on different NPG structures in order to uncover the effect of chemical modification of the bridges between the Graphene Nano Ribbons (GNR's) in the NPG. From an applied perspective, one of the main motivations is to find out how these bridges can be chemically modified in order to control the current through the material. A recent study[6], submitted for publication in JACS[7], has shown how one could possibly confine current flow to a single GNR channel by modification of bridges between GNR's in NPG utilising *Quantum Interference* (QI) effects (See Fig. 18). This provides a solution to an important requirement in carbon-based nanocircuitry design, namely nano confinement of electron flow. The study was focused on the difference in effect of having *meta* and *para* bridges between GNR's. Meta and para bridges are essentially benzene rings connected in two different ways (See Fig. 19). The meta and para bridges are 'static' cases in the sense that once made through organic synthesis, they are not interchangeable. However, if the bridges are functionalised with oxygen they become sensitive to f.ex. hydrogenation in basic/acidic environments, which tends to affect QI. Thus by hydrogenation it will be possible to tune the electrical properties of the material and make it sensitive to external environments. Studies[8] show that hydrogenation of specific sites on nano meter scale is possible experimentally. By the use of the functions developed in previous sections, this section will try to uncover what happens when oxygen is bonded to the benzene rings in the meta and para bridges. Subsequently hydrogenation of the oxygen will be tested. Following is a section introducing para and meta NPG in detail.

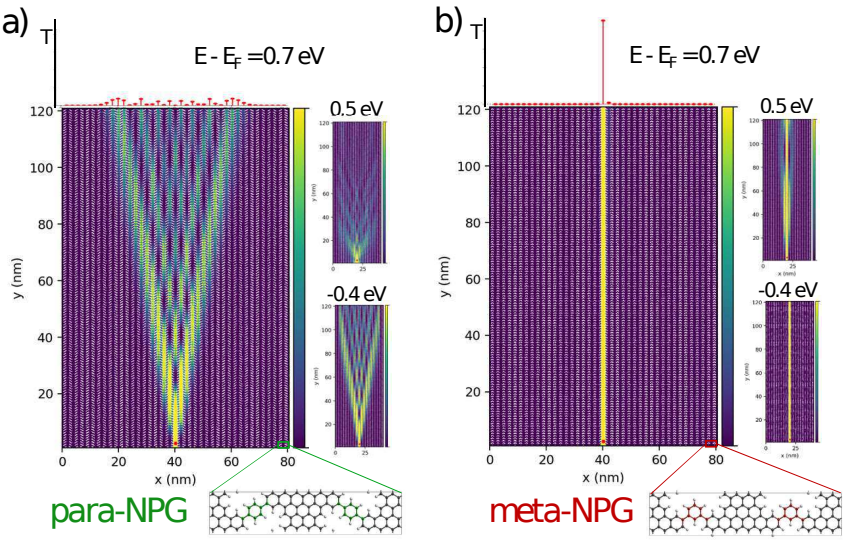


Figure 18: *Figure showing the currents through the GNR's in a large peace of NPG. Note how the current is spread across GNR's with the para bridge (a) and confined to a single GNR with meta bridge (b).* ***Used with permission of ISAAC***

A. Differences in para and meta bridges

In broad terms the difference in the *meta* and *para* structures lies in the path an electron will travel through the benzene ring to get across the bridge between GNR's in NPG. In the para bridge, the path across the aromatic ring is symmetric and so the electron will pass above or below with equal probability. Since the para bridge has three bonds in each direction across the ring, the path length in the para bridge is the same on each side (See Fig. 19 a)). This will cause constructive QI of the states once the waves meet on the other side of the ring. For para NPG this causes electronic coupling between the GNR channels. In the meta bridge, the way across the aromatic ring is not symmetric in the sense that there is two bonds across the path below and four bonds across on the path above (See Fig. 19 b)). This will cause a shift by half a wavelength between the two paths and thus create destructive QI between waves meeting on the other side of the ring. This causes electronic decoupling between GNR channels, allowing for confinement of injected currents in a single GNR channel[6]. In Fig. 20 band plots as well as transmission plots from the study[6] are shown for para and meta NPG. In the band plots the two sets of valence/conduction bands around the fermi level shows a shift for para and interference for meta NPG. Looking at the transmission plots next to the band plots one can see there is transmission and thus coupling of the states between the GNR's for para. The area between the two peaks at 0.500 eV and 1.20 eV show transport between the GNR's. In the transmission plot for meta the transmission is confined (Fig. 20 right). The transmission plot the area between the two peaks is now pretty much 0. This means GNR's are electrically decoupled in meta system. To summarise these results qualitatively: Shifts of the bands in the band plots, corresponds to effective coupling of the GNR's. Bands on top of each other, showing QI, corresponds to decoupling of the GNR's and thus electric confinement. In the following sections, band plots will mainly be used to show results. Therefore one should keep in mind how the band structures, qualitatively, relate to transmission. In Appendix B, Fig. 27 a figure obtained from the developed functions with band plots of normal, para and meta NPG can be seen.

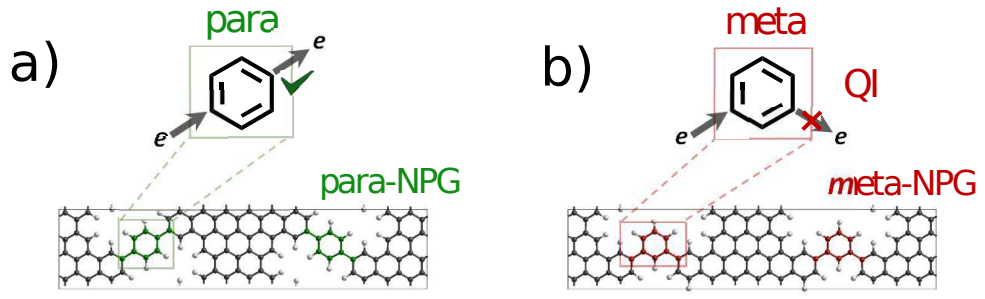


Figure 19: *a) showing the para NPG, b) showing the meta NPG. Used with permission from* ***Isaac***

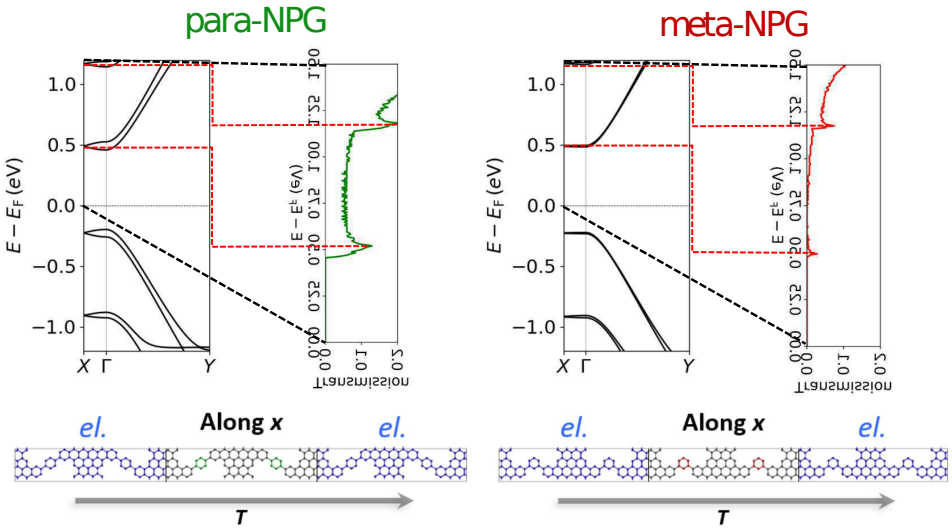


Figure 20: *Figure showing band plots and transmission plots for para and meta NPG. In the bottom of each plot is a schematic showing which way the transmission occurs in relation to the NPG.* ***Used with permission of ISAAC***

B. Tests with modified meta and para NPG

The tests consist of two kinds of chemical modification. Firstly it will be bonding oxygen sites to the meta and para NPG (Addition of Oxygen will be simulated by addition of another carbon site to the structure, effectively adding another pi-electron). Secondly simulations of hydrogenation will be done by bonding hydroxide groups to the bridges. These modifications will be carried out separately. Because of the complex nature of DFT the band plots obtained via DFT makes it hard to interpret what is actually going on in the system chemically. Therefore the aim is to show that using the methods developed, based on Tight Binding models, it is possible to reproduce bands structures calculated from DFT. If successful, it will provide a much easier way to understand the effect of the implemented chemical modification compared to the DFT approach. Band plots, as well as some on-site potential maps, calculated with DFT has been provided by *Isaac* for comparison. The geometries, used for calculations, are provided by *ISAAC* and have been optimised by DFT.

C. Test 1: Para-O_4-NPG

The first test is considering the Para-O_4-NPG. The basis structure is para-NPG where 4 oxygen sites are bonded, two on each benzene ring (See Fig. 21a). Starting by looking at the resulting DFT plot in Fig. 21b the fist thing to notice is that the valence bands show QI and thus decouples the GNR's. So in spite having a para bridge, which normally gives coupling, decoupling occurs when oxygen is bonded to the bridges. To simulate this practically, the developed method considers the bonded oxygen as carbon atoms. It does not differentiate between the type of atoms only the distance between them. As a baseline the potential of these atoms are thus the same. Moreover, the hydrogen, included in the provided optimised geometries, will be removed before any calculations take place. In Fig. 21c one can see the resulting band structure plot, where calculations have been carried out, using only the provided geometry. No modifications where made. The band plot does not show much resemblance with the one obtained, using DFT. It is pretty much symmetric around the fermi level and it is hard to make out whether there is coupling or decoupling between GNR's. This

is expected. As seen in the potential map of Fig. 21e, the sites where the oxygen is bonded, have a potential much different from that of the rest of the system (the four dark blue spots). As mentioned, the baseline of the method does not take into account differentiating on-site potentials within a cell. So to compensate for this, the on-site potential of the four bonded oxygen is changed. In Fig. 21d one can see the resulting band plot after the on-site potential of the bonded oxygen has been change to $-0.500\,\text{eV}$. The resulting band plot resembles the DFT calculations much more. There is now QI in the valence bands which means decoupling of the GNR's. This means that the bonded oxygen effectively lowers the potential of the GNR's thus creating decoupling. The reason behind being that oxygen adds extra electrons to the the pi-conjugated system. So far the simple model have effectively managed to explain what happens when oxygen is bonded to para NPG.

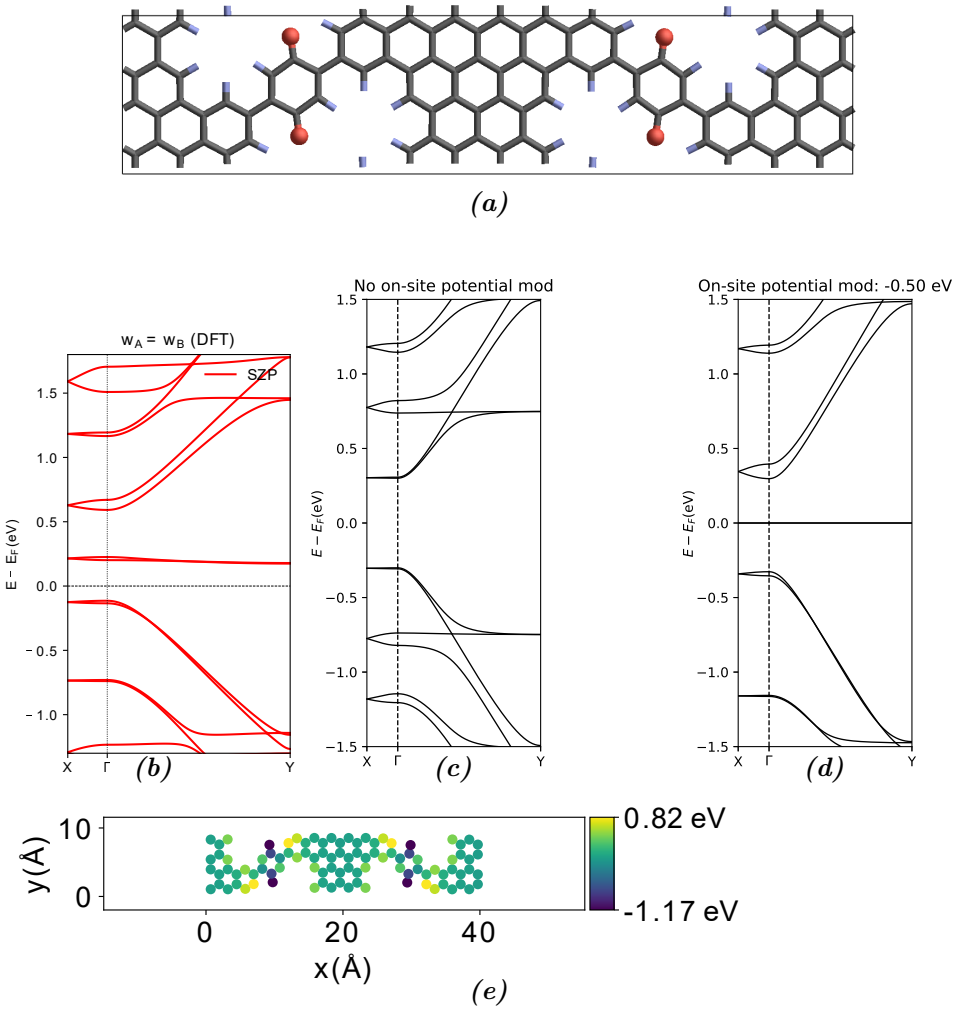


Figure 21: *a) Overview of the system with carbon atoms (grey sticks), oxygen atoms (red balls) and hydrogen atoms (blue sticks). b) Band structure obtained using DFT. c) Band structures obtained using developed program with no on-site potential mods. c) Band structures obtained using developed method with the on-site potential changed to* $-0.500\,\text{eV}$. *d). Potential map of the system.*

D. Test 2: P-$(OH)_4$-NPG

Next test will be with the P-$(OH)_4$-NPG system. Again the basis structure is para-NPG with bonded oxygen, but here the aim is to simulate hydrogenation, so therefore the oxygen

atoms added will become hydroxide groups (See Fig. 22a). The DFT plot in Fig. 22b shows a shift in the valence/conduction bands, so the GNR's have been coupled again. Here the first approach is to try to use the applied on-site potential for the bonded oxygen from the previous test (Section VI B) as a baseline. This is on the assumption that oxygen still contribute to the pi-conjugated system. In Fig. 22c the resulting band plot can be seen. It shows exactly the same as Fig. 21d. What one can conclude for this is that even though the geometries provided are optimised with DFT specifically for each system, the developed method does not distinguish the small differences in distance between atoms, which leads to hopping. The result is that, seen from a tight binding point of view, the geometries are the same initially even though they have been optimised individually with DFT. Therefore the resulting plots are identical. A natural way to compensate for this is to change the on-site potential by a value significantly lower, so that hopping can occur. Using the potential plot in Fig. 22f as a guide the on-sties of the oxygen are changed to $-2\,\mathrm{eV}$. The resulting plot can be seen in Fig. 22d. The plot show good agreement with the DFT. The bands have shifted again and the GNR's are coupled. This means hydrogenation decouples the oxygen, so by lowering the on-site potential of one can get effective decoupling of oxygen. As seen this gives a satisfying result. Now that it is known that hydrogenation decouples the oxygen, a last hypothesis is that by removing the hydroxide group entirely, before calculation, the result will be similar to Fig. 22d. In Fig. 22e the resulting plot can be seen. As hypothesised, the resulting band plot show resemblance with both the DFT and Fig. 22d. So the decoupling of oxygen by hydrogenation can be simulated simply by removing the hydroxide group entirely. Again there is shifts in the bands and thus coupling of the GNR's. This means that the developed method can uncover what happens chemically when the NPG is hydrogenated and it can do so in a simple manner.

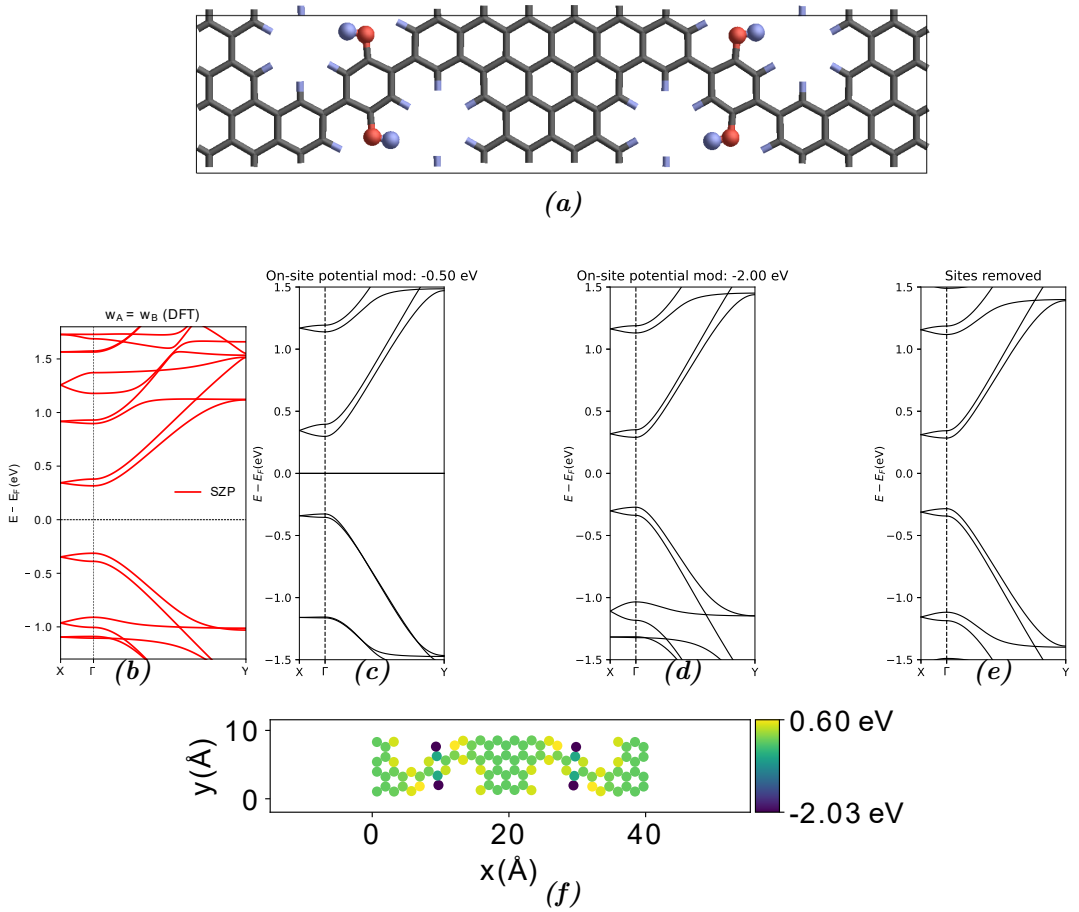


Figure 22: *a) Overview of the system with carbon atoms (grey sticks), oxygen atoms (red balls) and hydrogen atoms (blue balls). b) Band structure obtained using DFT. c) Band structures obtained using developed program with with on-site potential changed to* $-0.500\,\mathrm{eV}$*. d) Band structures obtained using developed method with on-site potential changed to* $-2\,\mathrm{eV}$*. e) Band structures obtained using the developed method with sites removed. f) Potential map of the system.*

E. Test 3: Meta-NPG with oxygen added symmetrically

Moving on to the meta NPG, the test is going to be exactly the same as Section VI B. To see the structure of the system look in Fig. 23a. Recall from Section VI A that the meta NPG gives rise to QI and thus decoupling of the GNR's. However as seen in Section VI B, functionalising the bridges with oxygen caused some quantum interference in the para and thus decoupling of the GNR's in para NPG. The expected outcome of this test is that by functionalising the meta NPG bridges with oxygen will give rise to shifts in the bands and coupling of the GNR's. First the DFT plot in Fig. 23b show that there is indeed a shift in the valence and that it is bigger than the shift in the conduction band. So by functionalising with oxygen the effect is most prevalent at the valence band for both para and meta. In Fig. 23c the blot for the system with no modification is shown. Here one can also so see shifts in the valence and conduction bands. However, the plot is totally symmetric around 0 so it is not entirely accurate. Again the potential map in Fig. 23e is used to get the on-site potential of the oxygen, so that they can be modified with the developed method. The result is seen Fig. 23d. Now the band plot shows a shift of the bands downwards, just like the DFT and the valence bands have a bigger shift compared to the conduction bands. So effectively

the GNR's are coupled.

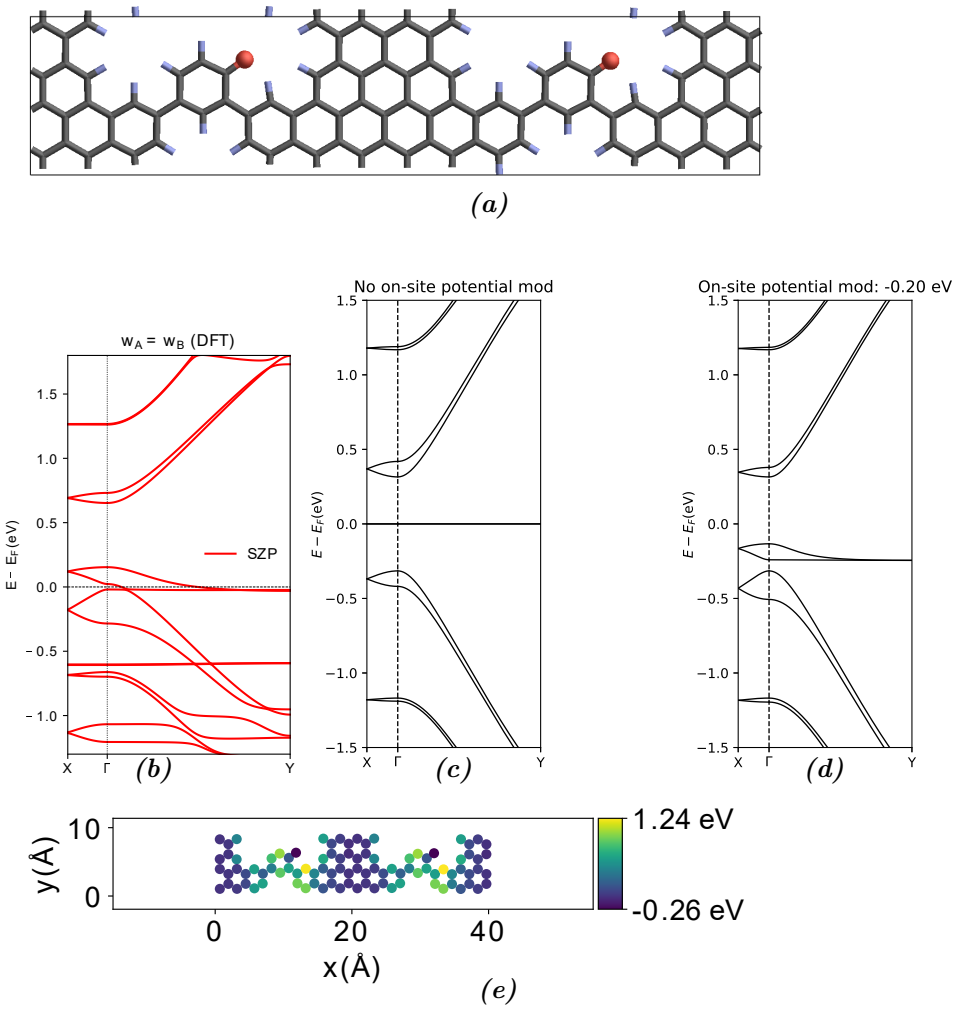


Figure 23: *a) Overview of the system with carbon atoms (grey sticks), oxygen atoms (red balls) and hydrogen atoms (blue sticks). b) Band structure obtained using DFT. c) Band structures obtained using developed program with no on-site potential mods. c) Band structures obtained using developed method with on-site potential changed to* $-0.200\,\mathrm{eV}$*. d) Potential map of the system.*

F. Test 4: Simulating hydrogenation of meta-NPG with symmetrically added oxygen

The fourth and final test will be with the same approach as Section VI D only here the hydrogenation is of the meta-NPG with oxygen bonded symmetrically (See Fig. 24a). Here one would expect to see QI and decoupling of the GNR's again as the system is hydrogenated, following the logic of the other tests. The DFT plot in Fig. 24b shows QI for both valence and conduction bands. Again the first plot produced, using the developed method, will be using the on-site modification from Fig. 23d as a baseline. The resulting plot can be seen in Fig. 24d. As expected it is identical to that of Fig. 23d. There is a bigger shift in the valence band and the GNR's are coupled. So the on-site potential is lowered with a value according to Fig. 24f and the result can be seen in Fig. 24d. As expected the plot is in good agreement with DFT, showing effective decoupling of oxygen and thus decoupling of GNR's just like the unmodified meta NPG discussed in Section VI A. Finally by removing the hydroxide

group entirely, using the developed method afterwards, the resulting band plot in Fig. 24e is obtained. The result is also in good agreement with DFT. This again proves that the developed method can uncover the chemical effects of functionalisation with oxygen and subsequent hydrogenation in a simple manner.

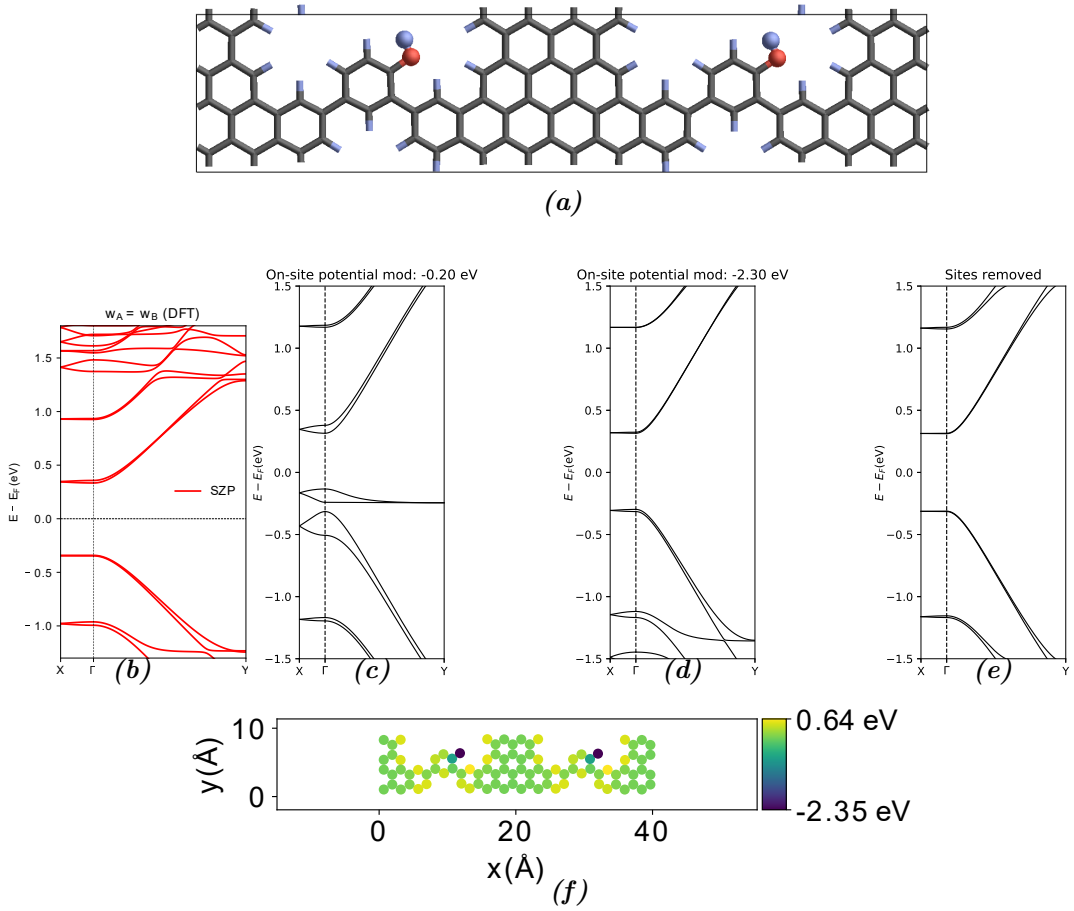


Figure 24: *a) Overview of the system with carbon atoms (grey sticks), oxygen atoms (red balls) and hydrogen atoms (blue balls). b) Band structure obtained using DFT. c) Band structures obtained using developed program with with on-site potential changed to* $-0.200\,\mathrm{eV}$*. d) Band structures obtained using developed method with on-site potential changed to* $-2.30\,\mathrm{eV}$*. e) Band structures obtained using the developed method with sites removed. f) Potential map of the system.*

G. Test summary

For the para systems, the added oxygen introduced some QI to the system. It was possible to show that the developed program could reproduce the results obtained from DFT. This did require some adjustment to the on-site potential of the added oxygen. By hydrogenation of the oxygen it was shown that the system again returned to a shift in the bands. Again by adjusting the on-site potential it was possible to obtain results very close to the DFT calculation. The results are also comparable with the ones mentioned in Section VI A. This shows that hydrogenation of oxygen can change the system to resemble regular para-NPG, even if it has oxygen bonded. The same can be said for the meta tests. All in all these results give a better, intuitive understanding of what happens with chemically with the systems and what effect it has on the electron transport in NPG. This is because all manipulations can be followed with ease and the results are easy to interpret. The tests have shown that it

is possible to tune the electrical properties of chemically modified meta and para NPG by hydrogenation and thus providing a novel approach in designing sensitive nanocircuitries, for f.ex. PH-sensors. This rounds off the summary of the test section. The section following will conclude the project as a whole.

VII. CONCLUSION

ACKNOWLEDGMENTS

The authors would like to thank...

---

[1] C. Moreno, M. Vilas-Varela, B. Kretz, A. Garcia-Lekue, M. V. Costache, M. Paradinas, M. Panighel, G. Ceballos, S. O. Valenzuela, D. Peña, and A. Mugarza, Bottom-up synthesis of multifunctional nanoporous graphene, Science **360**, 199 (2018), https://science.sciencemag.org/content/360/6385/199.full.pdf.

[2] N. R. Papior, sisl: v0.9.5 (2018).

[3] G. Calogero, N. R. Papior, B. Kretz, A. Garcia-Lekue, T. Frederiksen, and M. Brandbyge, Electron Transport in Nanoporous Graphene: Probing the Talbot Effect, Nano Letters **19**, 576 (2019).

[4] T. Markussen, R. Stadler, and K. S. Thygesen, The Relation between Structure and Quantum Interference in Single Molecule Junctions, Nano Letters **10**, 4260 (2010).

[5] S. Simon, *The Oxford Solid State Basics* (OUP Oxford, 2013).

[6] G. Calogero, I. Alcón, N. Papior, A.-P. Jauho, and M. Brandbyge, Quantum inteference engineering of nanoporous graphene for carbon nanocircuitry (2019), submitted.

[7] Journal of American Chemical Society.

[8] J. Li, S. Sanz, M. Corso, D. J. Choi, D. Peña, T. Frederiksen, and J. I. Pascual, Single spin localization and manipulation in graphene open-shell nanostructures, Nature Communications **10**, 10.1038/s41467-018-08060-6 (2019).

List of Figures

List of Tables

Listings

Appendices

A. THE BENZENE MOLECULE

As an example the Hamiltonian of benzene is considered. In Fig. 25 one can see the indices of a benzene molecule. Remember that $\langle\phi_\pi(1)|\hat{H}|\phi_\pi(1)\rangle = 0$ and Eq. (II.2), the Hamiltonian reads:

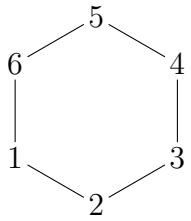


Figure 25: *Indices of a benzene molecule*

$$\mathbf{H} = V_{pp\pi} \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \end{array} \tag{A.1}$$

As a helping aid, Eq. (A.1) shows the atomic indices of the atom on the top and to the left of the matrix. This will give an understanding of how to work with such matrices. The structure of the benzene molecule is rotationally symmetric and rotating the indices one sixth must yield the same Hamiltonian. Consider the energy eigenvector:

$$\phi = \begin{pmatrix} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 \end{pmatrix} \tag{A.2}$$

There must exist an operator that rotates the indices as such:

$$C_6\phi = \begin{pmatrix} c_2 & c_3 & c_4 & c_5 & c_6 & c_1 \end{pmatrix} \tag{A.3}$$

The rotated Hamiltonian is the same, and thus C_6 and $\mathbf{H}$ commute. The rotated vector must be an eigenvector with the same energy and it should be possible to find simultaneous eigenvectors to C_6 and $\mathbf{H}$.

$$C_6\phi = \begin{pmatrix} c_2 & c_3 & c_4 & c_5 & c_6 & c_1 \end{pmatrix} = \lambda \begin{pmatrix} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 \end{pmatrix} \tag{A.4}$$

This operator C_6 is represented with the matrix:

$$\mathbf{C}_6 = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{A.5}$$

It can quickly be shown that the normalised eigenvectors to C_6 are

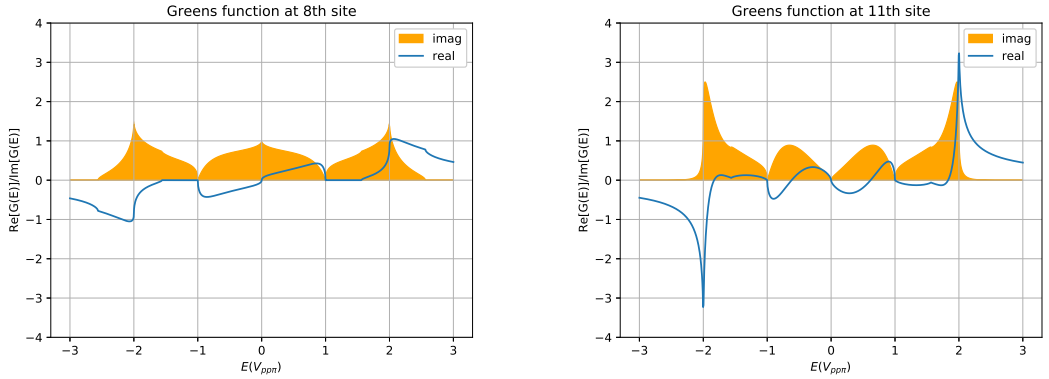
$$\phi_n = \frac{1}{\sqrt{6}} \begin{pmatrix} \lambda_n^0 & \lambda_n^1 & \lambda_n^2 & \lambda_n^3 & \lambda_n^4 & \lambda_n^5 \end{pmatrix}, \quad \lambda_n = \exp\{-i2\pi n/6\}, \quad n = 0,1,2,3,4,5 \tag{A.6}$$

These eigenvectors are also eigenvectors for $\mathbf{H}$ with the eigenvalues:

$$\varepsilon_n = \lambda_n + \lambda_{n-1} = 2\cos n\pi/3 \tag{A.7}$$

Thus thanks to the rotational symmetry it was possible to find the eigenvectors and eigenenergies for the Hamiltonian.

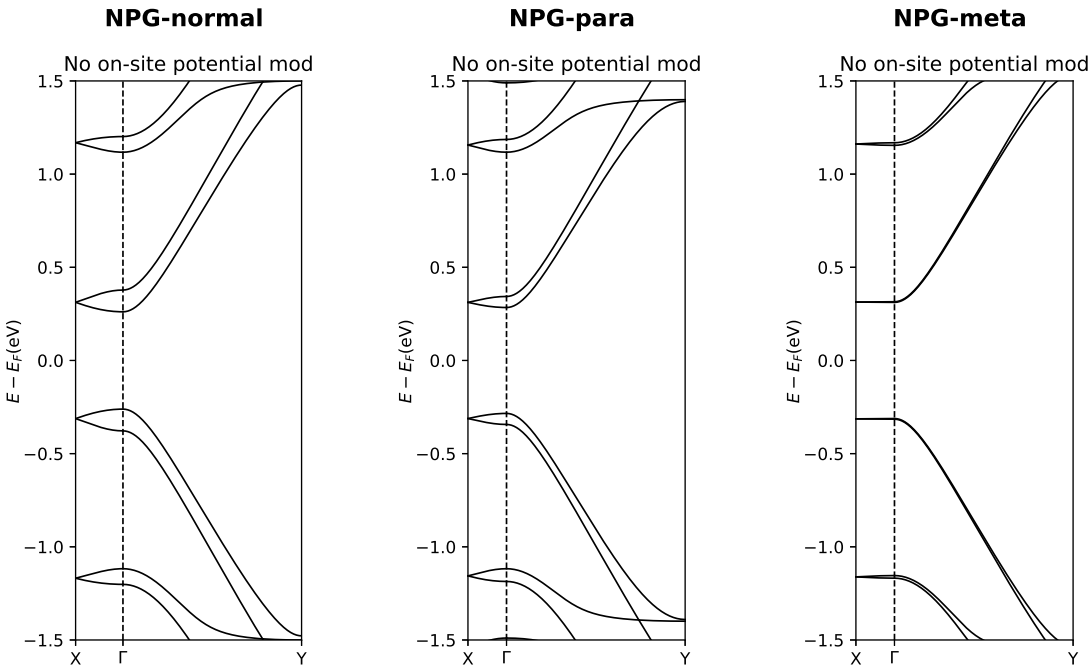
B. ADDITIONAL FIGURES



(a) *Figure showing a plot of the Green's function at the 8^{th} site*

(b) *Figure showing a plot of the Green's function at the 11^{th} site*

Figure 26: *Two plots showing how the Green's function changes as the site is changed. The 8^{th} and 11^{th} sites are corresponding to atoms of those indices (8, 11) in Fig. 5. Note how the LDOS changes (imaginary part) for the different sites.*



(a) *Normal NPG*

(b) *Para NPG*

(c) *Meta NPG*

Figure 27: *Plot showing band structures in the energy range $-1.50\,\mathrm{eV}$ to $1.50\,\mathrm{eV}$ for normal, para and meta NPG. The are plotted between symmetry points X and Y with respect to the origin Γ*