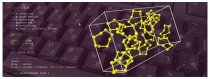
Convergence properties of TranSIESTA/TBtrans

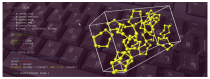
Nick Papior Andersen & Pablo Ordejón

Tel Aviv SIESTA/TranSIESTA workshop

10. September 2014



Outline



Calculating the density matrix

Integration in $\mathbf{k}$ and energy space

General formalism of Non-Equilibrium Green's functions

An integration over $\mathbf{k}$ space and energy space

$$\rho = \frac{1}{\pi}\iint_{-\infty,\mathrm{BZ}}^{\infty} \mathrm{d}\epsilon \mathrm{d}\mathbf{k}\, \mathbf{G}_{\mathbf{k}}(z)\Big[\boldsymbol{\Gamma}_{L,\mathbf{k}}(z) n_{F,L}(\epsilon) + \boldsymbol{\Gamma}_{R,\mathbf{k}}(z) n_{F,R}(\epsilon)\Big]\mathbf{G}^{\dagger}_{\mathbf{k}}(z), \qquad z = \epsilon + i\eta$$

$$\mathbf{G}_{\mathbf{k}}(z) = \frac{1}{z\mathbf{S}_{\mathbf{k}} - \mathbf{H}_{\mathbf{k}} - \Sigma_{L,\mathbf{k}}(z) - \Sigma_{R,\mathbf{k}}(z)}$$

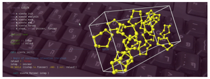
$$\boldsymbol{\Gamma}_{j,\mathbf{k}}(z) = i\Big[\Sigma_{j,\mathbf{k}}(z) - \Sigma^{\dagger}_{j,\mathbf{k}}(z)\Big]/2$$

$$n_{F,j} = \frac{1}{1 + \exp\Big[(\epsilon - \mu_j)/(k_B T)\Big]}$$

η broadens the density contribution

Inverting a *huge* matrix is extremely expensive, scales with N^3!

For those interested: You can derive the following equation using the above 3 equations!
We will leave that as an exercise!



Arriving at the governing formulas

See Brandbyge et al., DOI: 10.1103/PhysRevB.65.165401 for more details

The full density can be calculated in N_μ ways, where $N_\mu \in \{1,2\}$.

$$\rho_L = \frac{i}{\pi}\iint_{-\infty,\mathrm{BZ}}^{\infty} \mathrm{d}\epsilon \mathrm{d}\mathbf{k} \left[\mathbf{G}_\mathbf{k}(z) - \mathbf{G}_\mathbf{k}^\dagger(z)\right] n_{F,L}(\epsilon) + \frac{1}{\pi}\iint_{-\infty,\mathrm{BZ}}^{\infty} \mathrm{d}\epsilon \mathrm{d}\mathbf{k}\, \mathbf{G}_\mathbf{k}(z)\mathbf{\Gamma}_{R,\mathbf{k}}(\epsilon)\mathbf{G}_\mathbf{k}^\dagger(z)\left[n_{F,R}(\epsilon) - n_{F,L}(\epsilon)\right],$$

$$\rho_R = \frac{i}{\pi}\iint_{-\infty,\mathrm{BZ}}^{\infty} \mathrm{d}\epsilon \mathrm{d}\mathbf{k} \left[\mathbf{G}_\mathbf{k}(z) - \mathbf{G}_\mathbf{k}^\dagger(z)\right] n_{F,R}(\epsilon) + \frac{1}{\pi}\iint_{-\infty,\mathrm{BZ}}^{\infty} \mathrm{d}\epsilon \mathrm{d}\mathbf{k}\, \mathbf{G}_\mathbf{k}(z)\mathbf{\Gamma}_{L,\mathbf{k}}(\epsilon)\mathbf{G}_\mathbf{k}^\dagger(z)\left[n_{F,L}(\epsilon) - n_{F,R}(\epsilon)\right].$$

This is split in two terms

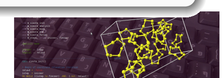
Equilibrium

$$\rho_{j,\mathrm{eq}} = \frac{i}{\pi}\iint_{-\infty,\mathrm{BZ}}^{\infty} \mathrm{d}\epsilon \mathrm{d}\mathbf{k} \left[\mathbf{G}_\mathbf{k}(z) - \mathbf{G}_\mathbf{k}^\dagger(z)\right] n_{F,j}(\epsilon)$$

Non-equilibrium

$$\Delta_{j,\mathrm{neq}} = \frac{1}{\pi}\iint_{-\infty,\mathrm{BZ}}^{\infty} \mathrm{d}\epsilon \mathrm{d}\mathbf{k}\, \mathbf{G}_\mathbf{k}(z)\mathbf{\Gamma}_{j'\neq j,\mathbf{k}}(\epsilon)\mathbf{G}_\mathbf{k}^\dagger(z)\left[n_{F,j'}(\epsilon) - n_{F,j}(\epsilon)\right].$$

Precision comes in how well we calculate both terms



k-point sampling

TranSIESTA ≠ TBtrans

$$\int \mathrm{d}\mathbf{k} \approx \sum_{\mathbf{k}}$$

TranSIESTA

In TranSIESTA the $\mathbf{k}$-point sampling is the same as for transverse directions in SIESTA

FDF-file:

```
%block kgrid_Monkhorst_Pack
  <A1>    0      0    0.
    0   <A2>     0    0.
    0     0    <A3>   0.
%endblock kgrid_Monkhorst_Pack
```

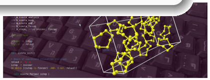
TranSIESTA perception of FDF-file:

```
%block kgrid_Monkhorst_Pack
  <A1>    0      0    0.
    0   <A2>     0    0.
    0     0      1    0.
%endblock kgrid_Monkhorst_Pack
```

TranSIESTA will truncate number of $\mathbf{k}$-points in A3 direction to 1

Converge $\mathbf{k}$-points for SIESTA and utilise that for your simulations

Note, this is not so for TBtrans, we will return to this!



Integration in energy space

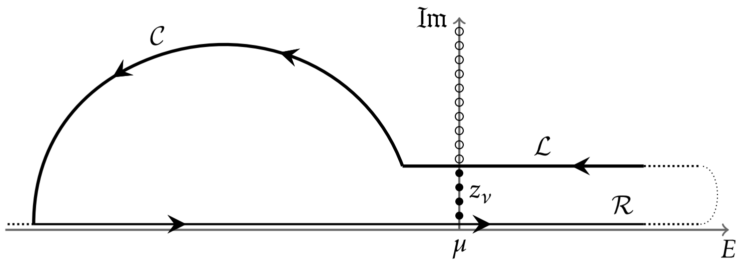
Equilibrium density

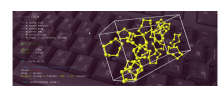
$$\rho_{\mathrm{eq},\mathbf{k}} = \frac{i}{\pi}\int_{-\infty}^{\infty}\mathrm{d}\epsilon\left[\mathbf{G}_{\mathbf{k}}(z) - \mathbf{G}_{\mathbf{k}}^{\dagger}(z)\right]n_F(\epsilon)$$

- The Green's function *only* has poles on the real axis (the energy eigenvalues) and on the imaginary axis (the Fermi function poles)
- We employ a complex contour method based on the residue theorem

$$\oint\mathrm{d}\epsilon\left[\mathbf{G}_{\mathbf{k}}(z) - \mathbf{G}_{\mathbf{k}}^{\dagger}(z)\right]n_F(z-\mu) = -i2\pi k_B T\sum_{z_\nu}\left[\mathbf{G}_{\mathbf{k}}(z_\nu) - \mathbf{G}_{\mathbf{k}}^{\dagger}(z_\nu)\right], \qquad z_\nu = ik_BT\pi(2\nu+1)$$

Partition the LHS to arrive at the expression in Brandbyge et al., DOI: 10.1103/PhysRevB.65.165401





$$\oint\mathrm{d}\epsilon = \int_{\mathcal{R}}\mathrm{d}\epsilon + \int_{\mathcal{L}}\mathrm{d}\epsilon + \int_{\mathcal{C}}\mathrm{d}\epsilon$$

Integration in energy space

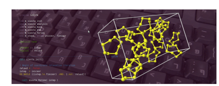
Equilibrium density

- The Green's function *only* has poles on the real axis (the energy eigenvalues) and on the imaginary axis (the Fermi function poles)
- We employ a complex contour method based on the residue theorem

$$\oint \mathrm{d}\epsilon \Big[\mathbf{G}_{\mathbf{k}}(z) - \mathbf{G}^{\dagger}_{\mathbf{k}}(z)\Big] n_F(z-\mu) = -i2\pi k_B T \sum_{z_\nu} \Big[\mathbf{G}_{\mathbf{k}}(z_\nu) - \mathbf{G}^{\dagger}_{\mathbf{k}}(z_\nu)\Big], \qquad z_\nu = i k_B T \pi (2\nu + 1)$$

  Partition the LHS to arrive at the expression in Brandbyge et al., DOI: 10.1103/PhysRevB.65.165401

- The Green's function is smooth far in the complex plane, whereas it is non-smooth on the real-axis



Integration in energy space

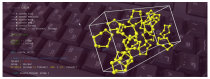
Equilibrium density

- The Green's function *only* has poles on the real axis (the energy eigenvalues) and on the imaginary axis (the Fermi function poles)
- We employ a complex contour method based on the residue theorem

$$\oint \mathrm{d}\epsilon \Big[\mathbf{G}_{\mathbf{k}}(z) - \mathbf{G}^{\dagger}_{\mathbf{k}}(z)\Big] n_F(z-\mu) = -i2\pi k_B T \sum_{z_\nu} \Big[\mathbf{G}_{\mathbf{k}}(z_\nu) - \mathbf{G}^{\dagger}_{\mathbf{k}}(z_\nu)\Big], \qquad z_\nu = i k_B T \pi(2\nu+1)$$

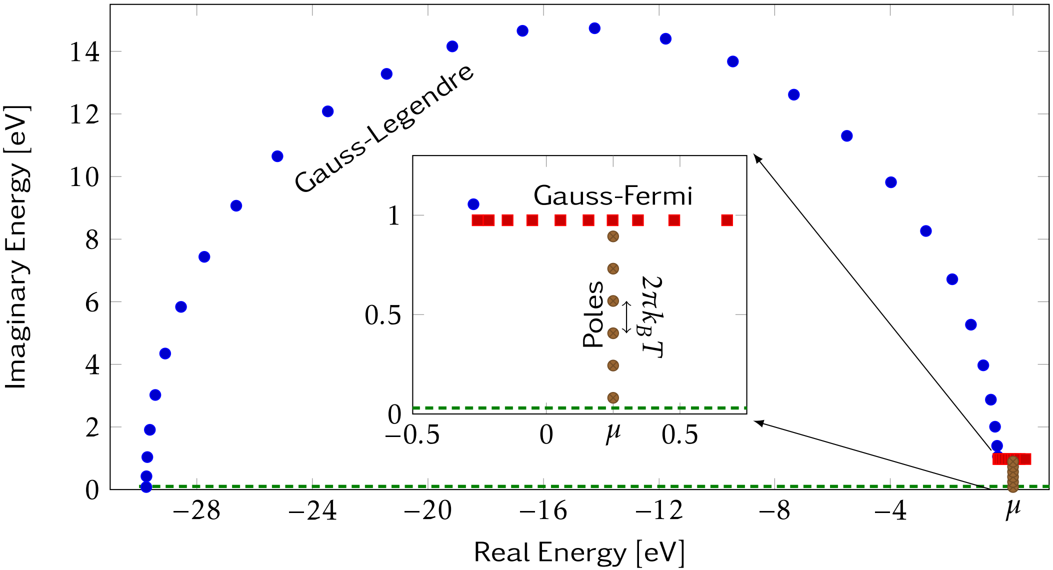
Partition the LHS to arrive at the expression in Brandbyge et al., DOI: 10.1103/PhysRevB.65.165401

- The Green's function is smooth far in the complex plane, whereas it is non-smooth on the real-axis
- We are forced to do numerical integration and resort to Gaussian quadrature methods

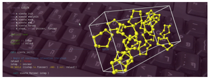


Integration in energy space

Equilibrium density — an example



Lower bound	`TS.ComplexContourEmin`
Gauss-Legendre	`TS.ComplexContour.NCircle`
Gauss-Fermi	`TS.ComplexContour.NLine`
Poles	`TS.ComplexContour.NPoles`



Integration in energy space

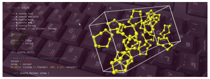
Non-equilibrium density

This is where trouble enters

This triple product is the culprit:

$$\Delta_{j,\mathrm{neq},\mathbf{k}} = \int_{-\infty}^{\infty} \mathrm{d}\epsilon\, \mathbf{G}_{\mathbf{k}}(z)\boldsymbol{\Gamma}_{j,\mathbf{k}}\mathbf{G}_{\mathbf{k}}^{\dagger}(z)(n_{F,j}(\epsilon) - n_{F,j'}(\epsilon)), \quad z = \epsilon + i\eta$$

- Along the real axis the triple-product is non-smooth
- We cannot use Gaussian quadrature methods
- We must resort to fine grained numerical integration



Integration in energy space

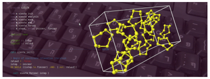
Non-equilibrium density

This is where trouble enters

This triple product is the culprit:

$$\Delta_{j,\mathrm{neq},\mathbf{k}} = \int_{-\infty}^{\infty} \mathrm{d}\epsilon\, \mathbf{G}_{\mathbf{k}}(z)\boldsymbol{\Gamma}_{j,\mathbf{k}}\mathbf{G}^{\dagger}_{\mathbf{k}}(z)(n_{F,j}(\epsilon) - n_{F,j'}(\epsilon)), \quad z = \epsilon + i\eta$$

- Along the real axis the triple-product is non-smooth
- We cannot use Gaussian quadrature methods
- We must resort to fine grained numerical integration
- The bias window is governed by the difference $n_{F,i}(\epsilon) - n_{F,i'}(\epsilon)$, above and below the corresponding chemical potentials will the Fermi-functions limit the contribution



Integration in energy space

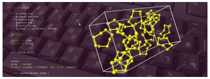
Non-equilibrium density

This is where trouble enters

This triple product is the culprit:

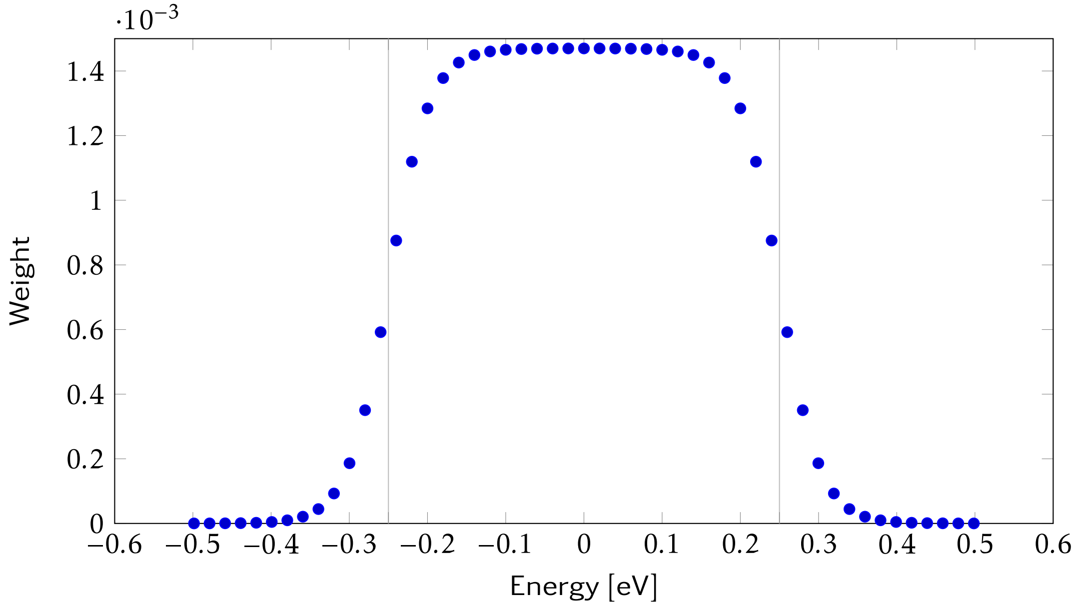
$$\Delta_{j,\mathrm{neq},\mathbf{k}} = \int_{-\infty}^{\infty} \mathrm{d}\epsilon\, \mathbf{G}_{\mathbf{k}}(z)\mathbf{\Gamma}_{j,\mathbf{k}}\mathbf{G}^{\dagger}_{\mathbf{k}}(z)(n_{F,j}(\epsilon) - n_{F,j'}(\epsilon)), \quad z = \epsilon + i\eta$$

- Along the real axis the triple-product is non-smooth
- We cannot use Gaussian quadrature methods
- We must resort to fine grained numerical integration
- The bias window is governed by the difference $n_{F,i}(\epsilon) - n_{F,i'}(\epsilon)$, above and below the corresponding chemical potentials will the Fermi-functions limit the contribution
- Control *broadening* of DOS along real axis with imaginary part η, high $\Rightarrow$ broadening of levels and fewer points, low $\Rightarrow$ high accuracy and requires more points `TS.biasContour.Eta`

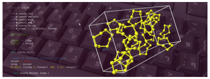


Integration in energy space

Non-equilibrium density — an example



- `TS.Voltage`
- `TS.biasContour.Eta <energy>`
- `TS.biasContour.NumPoints` $\sim V/0.01\mathrm{eV}$



Utility TBtrans

Calculating the transport

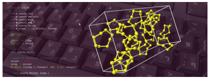
A utility to calculate the transport from a TranSIESTA calculation

```
tbtrans < RUN.fdf > RUNTBT.out
```

Calculating the current

$$I(V) = G_0 \iint_{-\infty,\mathrm{BZ}}^{\infty} \mathrm{d}\epsilon \mathrm{d}\mathbf{k}\ \mathrm{Tr}\left[\mathbf{\Gamma}_{L,\mathbf{k}} \mathbf{G}^{\dagger}_{\mathbf{k}}(z) \mathbf{\Gamma}_{R,\mathbf{k}} \mathbf{G}_{\mathbf{k}}(z)\right] (n_{F,L}(\epsilon) - n_{F,R}(\epsilon))$$

- Difference in Fermi functions makes window narrow (as for the non-equilibrium contribution)
- The full energy spectrum (outside of bias-window) is still interesting!
- Control energy window:
  ```
  TS.TBT.Emin <lower bound energy>
  TS.TBT.Emax <upper bound energy>
  TS.TBT.NPoints <number of separations>
  ```
- PDOS calculation from Green's function via
  ```
  TS.TBT.PDOSFrom <first atom>
  TS.TBT.PDOSTo <last atom>
  ```



Utility TBtrans

Example output

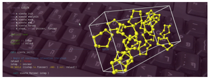
Several files:

- `LDOS` Bulk density of states for left electrode
- `RDOS` Bulk density of states for right electrode
- `TEIG` **k**-point resolved transmission eigenvalues, see *e.g.* Paulsson and Brandbyge, DOI: 10.1103/PhysRevB.76.115117
- `AVTEIG` **k**-point averaged transmission eigenvalues
- `TRANS` **k**-point resolved transmission
- `AVTRANS` **k**-point averaged transmission

AVTRANS:

```
# Averaged transmission, total DOS and projected DOS
#   E [eV]        Trans [G0]           TotDOS             PDOS
   -0.50000    0.52117304E+00    0.85934817E+00    0.35934817E+00
   -0.49000    0.51903380E+00    0.97680060E+00    0.47680060E+00
   -0.48000    0.51631594E+00    0.11658509E+01    0.80658509E+00
   ...
```

1. Energy
2. Transmission
3. Total DOS in central region
4. Projected DOS for denoted region (defaulted to entire central region)



Utility TBtrans

k-point sampling

Transmission is per surface area (double the electrode surface $\Rightarrow$ double the transmission, for bulk systems)

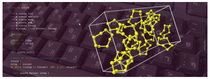
Important!

Transmission *highly* $\mathbf{k}$-point dependent. Even though the electronic structure is well explained we need a higher density of $\mathbf{k}$-points for TBtrans.

You can relate this to the bandstructure; you can recreate the bandstructure from an electronic structure calculation with few $\mathbf{k}$-points, yet you cannot obtain the full bandstructure by linear interpolation of the eigenvalues at the simulated $\mathbf{k}$-points

TBtrans $\mathbf{k}$-point sampling by (defaults to `kgrid_Monkhorst_Pack`):

```
%block TBT_kgrid_Monkhorst_Pack
  <A1>   0     0   0.
    0  <A2>    0   0.
    0    0     1   0.
%endblock TBT_kgrid_Monkhorst_Pack
```



Utility TBtrans

$\mathbf{k}$-point sampling

Transmission is per surface area (double the electrode surface $\Rightarrow$ double the transmission, for bulk systems)

Important!

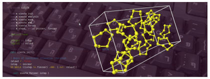
Transmission *highly* $\mathbf{k}$-point dependent. Even though the electronic structure is well explained we need a higher density of $\mathbf{k}$-points for TBtrans.

You can relate this to the bandstructure; you can recreate the bandstructure from an electronic structure calculation with few $\mathbf{k}$-points, yet you cannot obtain the full bandstructure by linear interpolation of the eigenvalues at the simulated $\mathbf{k}$-points

TBtrans $\mathbf{k}$-point sampling by (defaults to `kgrid_Monkhorst_Pack`):

```
%block TBT_kgrid_Monkhorst_Pack
  <A1>   0     0   0.
    0  <A2>    0   0.
    0    0     1   0.
%endblock TBT_kgrid_Monkhorst_Pack
```

A good example for this point is graphene



FDF-flags

Ensure charge neutrality, loosing/accumulating too much charge is erroneous

```
Qtot: 84.000
ts-charge:    1.461   14.528    1.479   50.436    1.447   14.547   83.897
```

TranSIESTA	kgrid_Monkhorst_Pack
TranSIESTA	TS.ComplexContourEmin
TranSIESTA	TS.ComplexContour.NCircle
TranSIESTA	TS.ComplexContour.NLine
TranSIESTA	TS.ComplexContour.NPoles
TranSIESTA	TS.biasContour.Eta
TranSIESTA	TS.biasContour.NumPoints
TranSIESTA/TBtrans	TS.Voltage
TBtrans	TS.TBT.Emin
TBtrans	TS.TBT.Emax
TBtrans	TS.TBT.NPoints
TBtrans	TS.TBT.PDOSFrom
TBtrans	TS.TBT.PDOSTo
TBtrans	TBT_kgrid_Monkhorst_Pack