

Case Study: Ag/Si(100) interface

Reproducing the results of '*General atomistic approach for modeling metal-semiconductor interfaces using density functional theory and non-equilibrium Green's function*'

Daniele Stradi, Umberto Martinez, Anders Blom, Mads Brandbyge, and Kurt Stokbro. Physical Review B, 93:155302, 2016.
URL: <http://arxiv.org/abs/1601.04651>, doi:10.1103/PhysRevB.93.155302.

Line Jelver



Set up initial guesses for the structure



1. Relax bulk Si and Ag from database using LDA

- 11x11x11 k-point grid
- 0.01 eV/Å force tolerance
- 0.001 eV/Å³ stress tolerance

Run locally

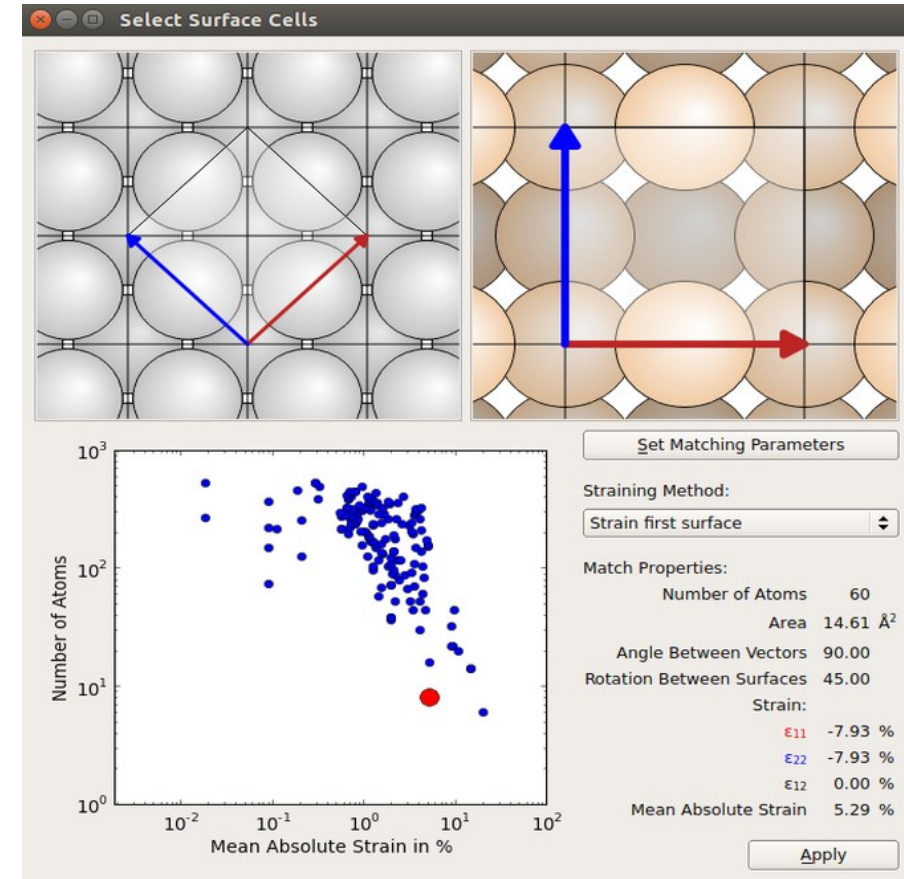
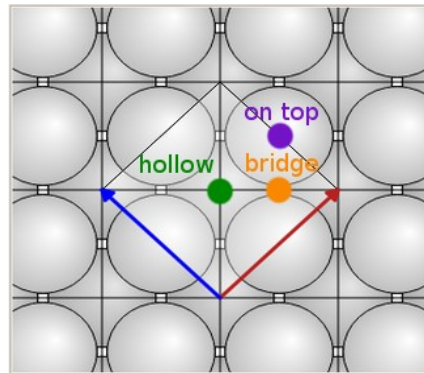
2. Create (100) surfaces of Si and Ag*

3. Create interface*

- 6 layers of Ag, 9 layers of Si
- Ag is strained 7.93 %

4. Make 3 guesses for the structure using the 'Move' tool

- Si dangling bond positioned at:
 - hollow site of Ag
 - bridge site of Ag
 - on top site of Ag



* See tutorial: Building an interface between Ag(100) and Au(111), http://docs.quantumwise.com/tutorials/ag_au_interface.html



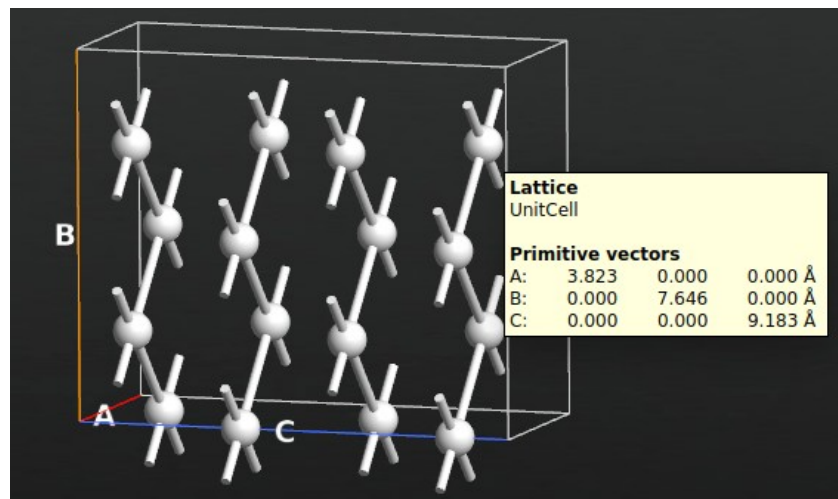
1. Relax the strained Ag electrode using LDA*

6x3x1 k-point grid
0.01 eV/Å force tolerance
0.001 eV/Å³ stress tolerance

Run locally

2. Apply the Ag elongation to the interface*

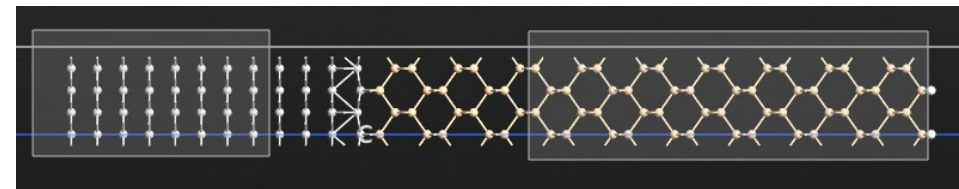
c-vector of the Ag electrode changes from 8.304 Å to 9.183 Å
which gives a elongation of **9.57%**



• Optimize the 3 interface slabs*

- Hydrogen passivation on Si side
- Add 35 Å of vacuum
- Constrains: 4 layers of Ag is fixed, 6½ layers of Si as rigid body - necessary for making the electrodes match bulk Ag and Si

20h on a 16-core node with 64 GB



- 6x3x1 k-point grid
- 0.02 eV/Å force tolerance

1. Create and optimize device with hollow site

- Use suggested central region and electrode size
- Constrains: 4 layers of Ag is fixed, 6½ layers of Si as rigid body
- 6x3x401 k-point grid
- 0.02 eV/Å force tolerance

2h on a 16-core node with 64 GB

* See tutorial: Guide on how to relax a device system, http://docs.quantumwise.com/tutorials/device_relaxation.html

Fit MGGA c-parameter to band gap of silicon



Run locally

For details, see tutorial on Meta-GGA and 2D confined InAs.*

1. Calculate bandstructure using MGGA

$$E_{\text{gap}} = 1.161 \text{ eV}$$
$$c = 1.0841$$

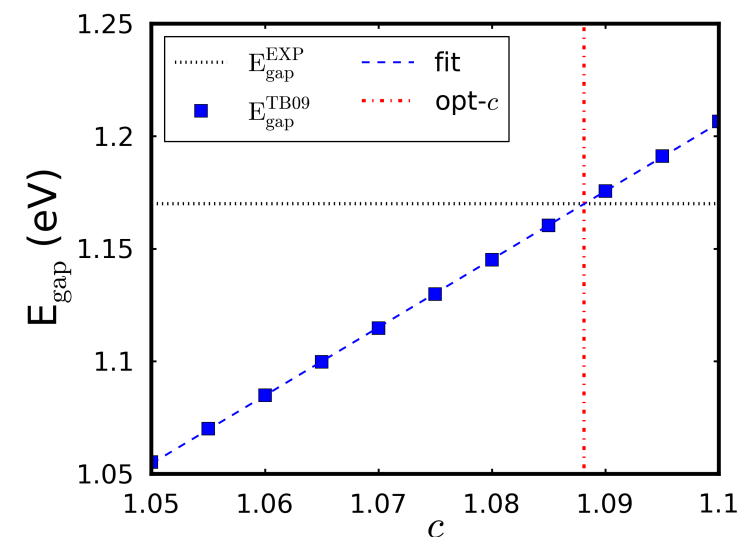
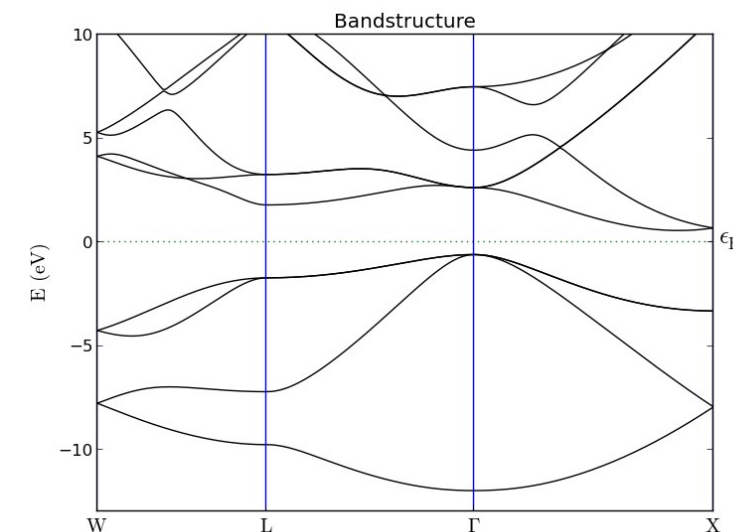
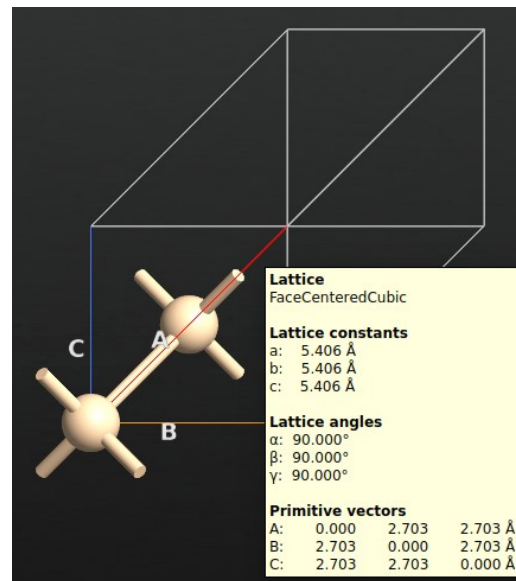
2. Do MGGA calculation of different fixed c-values

- $c = [1.05, 1.055, 1.06, 1.065, 1.07, 1.075, 1.08, 1.085, 1.09, 1.095, 1.1]$

3. Fit the c-value to the experimental band gap of

$$E_{\text{gap}} = 1.17 \text{ eV (0 K)** using } \text{c_fit.py}$$

$$\text{opt-c} = 1.0881$$



* http://docs.quantumwise.com/tutorials/inas_2d_mgga.html

** C. Kittel. Introduction to Solid State Physics. Wiley, 8th edition (2004)

Converge the doped device w.r.t. the silicon width



1. Add doping of $n_d = 10^{19} \text{ cm}^{-3}$ to the relaxed device
2. Do an HartreeDifferencePotential (HDP) analysis for different lengths (80 Å, 100 Å, 120 Å, 140 Å) of the silicon layer region using MGGA with c-opt.

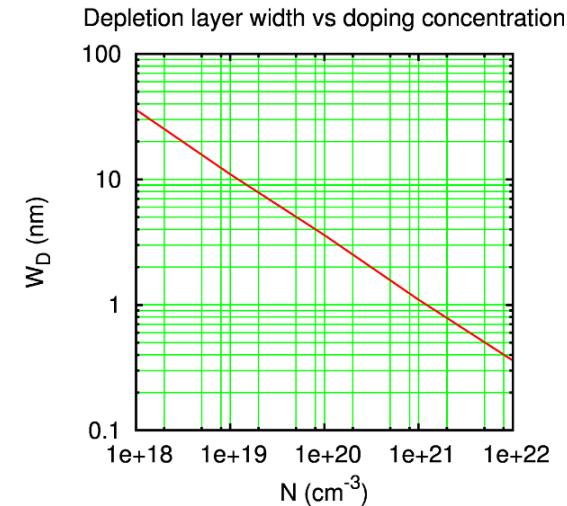
13h (for 140 Å) on a 16-core node with 64 GB

3. Calculate the average of the HDP using [hdp.py](#)

- Use the .nc file with the HDP analysis as input
- You will need:
 - z-distance between atomic layers in the left and right electrode material in Å (Determines the width of the Gaussian kernels used to calculate the average potential)
 - z-coordinate of the interface position in Å calculated as the midpoint between the last atom of the left material and the first atom in the right material

The potential is converged at 120 Å, fairly consistent with the expected depletion layer length of 100 Å*

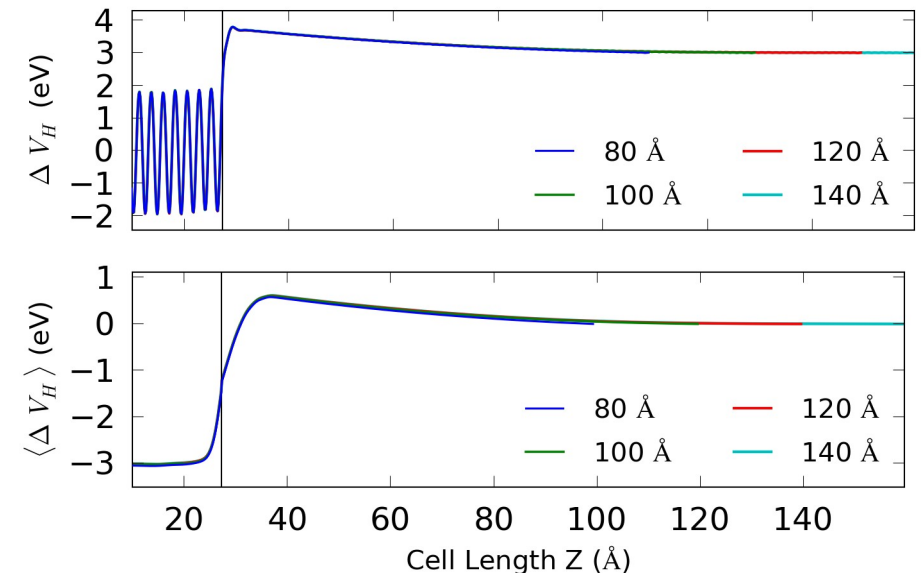
- You can compare the potentials of different lengths using [hdp_plot.py](#)



Notes:

- In order to converge the DFT calculation of the 120 Å device use:
damping_factor=0.05,
number_of_history_steps=12,
max_steps=400

- The kink at the interface border is due to the averaging procedure where Gaussians of different widths are being used.



* E. Kasper and D. J. Paul. Silicon Quantum Integrated Circuits. Springer-Verlag Berlin Heidelberg (2005)

Produce PLDOS + aligned EDP plot



1. Do a ProjectedLocalDensityOfStates (PLDOS) analysis from the converged 120 Å calculation

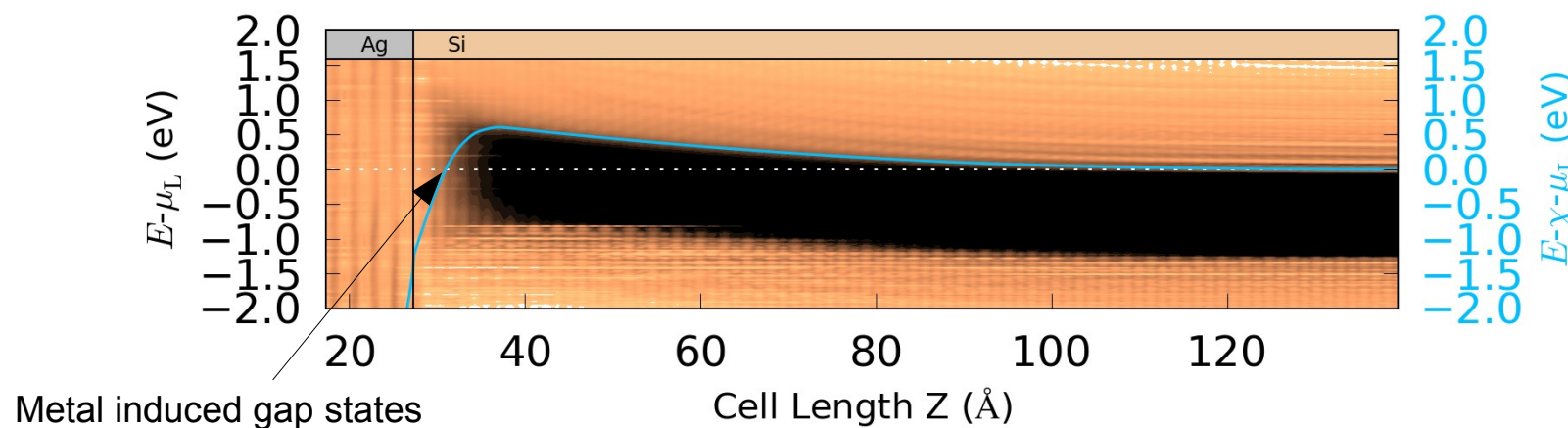
- [-2 eV: 2 eV] energy range with 401 energy points
- 18x9x401 k-point sampling

7h on a 16-core
node with 64 GB

2. Create the PLDOS + aligned HDP plot using `pldos_hdp.py`

- Uses the .nc file with the HDP and PLDOS analysis as input
- Requires same parameters as `hdp.py` along with some plotting preferences
- Returns the estimated Schottky barrier as the difference between the chemical potential of the left electrode and the maximum of the averaged Hartree difference potential and the position of the conduction band minimum w.r.t. the chemical potential of the left electrode (CB_{\min})

$$\Phi^{\text{pot}} = 606 \text{ meV (agrees with experimental results*)}$$



* Robert Balsano, Akitomo Matsubayashi, and Vincent P. LaBella. AIP Advances 3, 112110 (2013)

J. J. Garramone, J. R. Abel, I. L. Sitnitsky, and V. P. LaBella. Journal of Vacuum Science & Technology A 28, 643 (2010)

Calculate I-V curves and ideality factor



1. Do IVCurve analysis

- [-0.3 V: 0.3 V] voltage range with 15 points
- [-2 eV: 2 eV] energy range with 401 points
- 18x9 k-point sampling

15h on a 16-core
node with 64 GB

2. Use IV-Plot plugin or **IV.py** to plot the current vs. bias or current density vs. bias

NB: The current is much lower than the current calculated in the article*.
See the last part of the ppt file for an explanation.

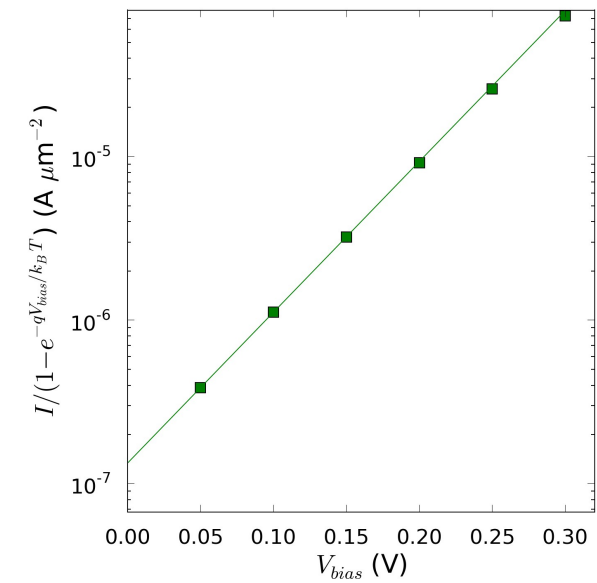
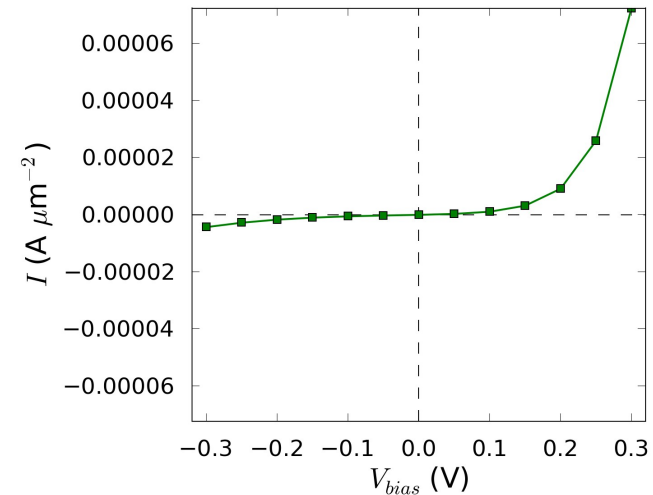
3. Thermionic emission theory IV characteristics of Schottky diode**:

$$I = I_0 e^{\frac{qV_{bias}}{nk_B T}} \left(1 - e^{-\frac{qV_{bias}}{k_B T}} \right)$$

- n is the ideality factor describing how much the system resembles an ideal Schottky diode

4. Use **IV-n-log.py** to extract the ideality factor from the slope of a log plot $I/(1 - e^{-qV_{bias}/k_B T})$

$$n = 1.8208$$



* Daniele Stradi, Umberto Martinez, Anders Blom, Mads Brandbyge, and Kurt Stokbro. Physical Review B 93, 155302 (2016)

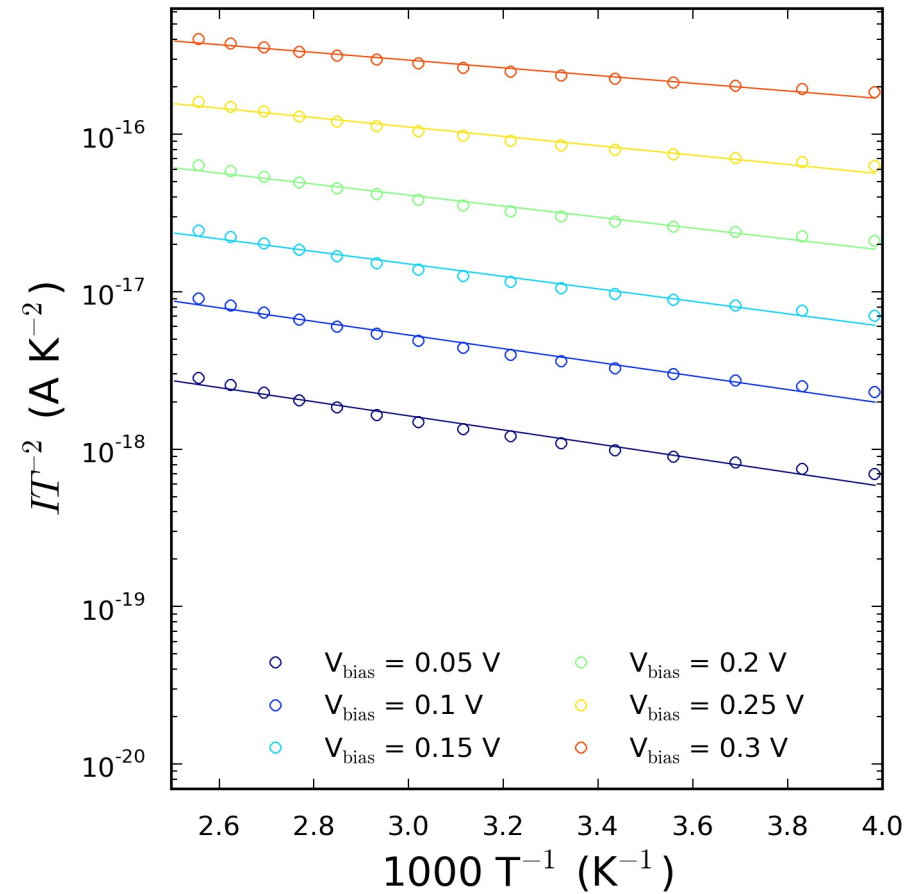
** S.M. Sze and K. N. Kwok. Physics of Semiconductor Devices. Wiley, 3rd edition (2006)



1. Use `lvsT.py` to calculate the linear-response transmission for each positive applied bias in a range of temperatures between 250 K and 400 K.

- You will need the .nc file containing the IVCurve analysis
- The script will calculate the transmission current from the Landauer-Büttiker expression:

$$I = \frac{2q}{h} \int T(E, \mu_L, \mu_R) \left[f\left(\frac{E - \mu_L}{k_B T}\right) - f\left(\frac{E - \mu_R}{k_B T}\right) \right] dE$$



Make Arrhenius plot and calculate Schottky barrier height



The Activation-Energy method* describes the I-T characteristics as

$$IT^{-2} = AA^* e^{-\frac{q\Phi^{AE}}{k_B T}} e^{\frac{q(V_{bias}/n)}{k_B T}}$$

It assumes that the Richardson constant A^* and Φ^{AE} are constant

1. Use **arrhenius.py** to make Arrhenius plot of $\ln(I/T^2)$ vs. $1/T$

You will need:

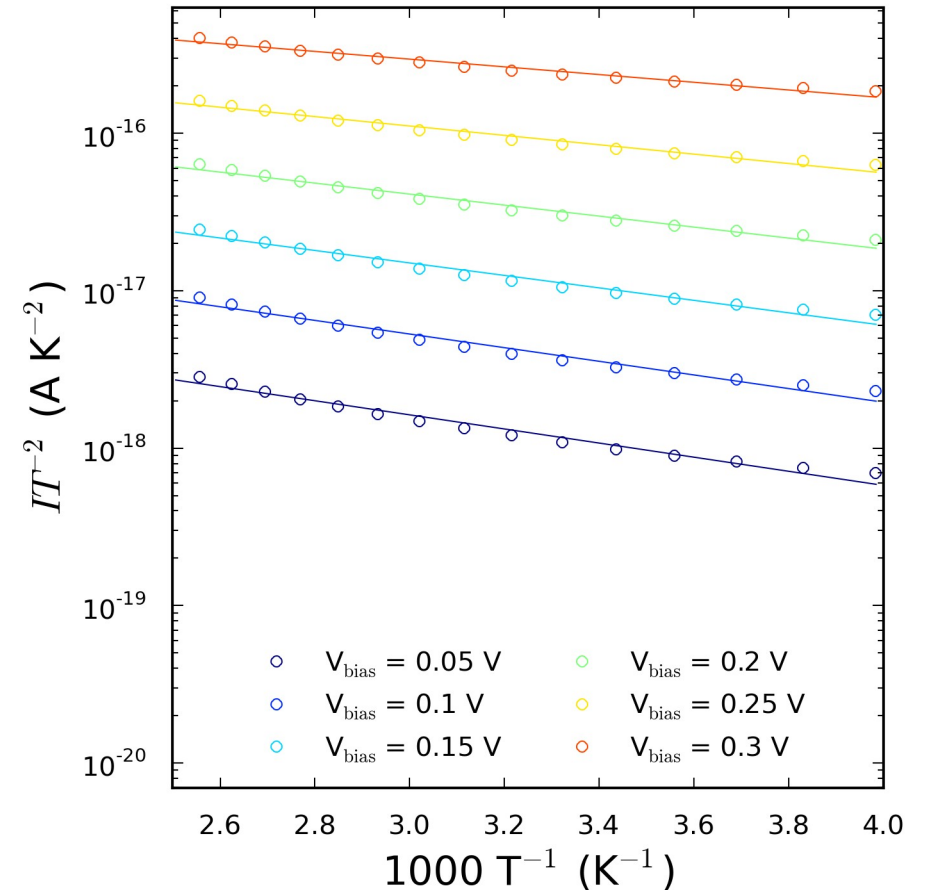
- The output files from **lvsT.py**
- The calculated ideality factor from **IV-n-log.py**

The script returns:

- The Schottky barrier height for each applied bias Φ^{AE} calculated from the slope of the plot.
- The effective Schottky barrier height for each applied bias, $\phi^{AE}(V_{bias})$, exported as a .dat file

$$\ln(I/T^2) = \ln(AA^*) - \frac{q}{k_B} \left(\Phi^{AE} - \frac{V_{bias}}{n} \right) \frac{1}{T} \Rightarrow$$

$$\phi^{AE}(V_{bias}) = \Phi^{AE} - \frac{V_{bias}}{n} = -\frac{k_B}{q} \frac{d[\ln(I/T^2)]}{d(1/T)}$$



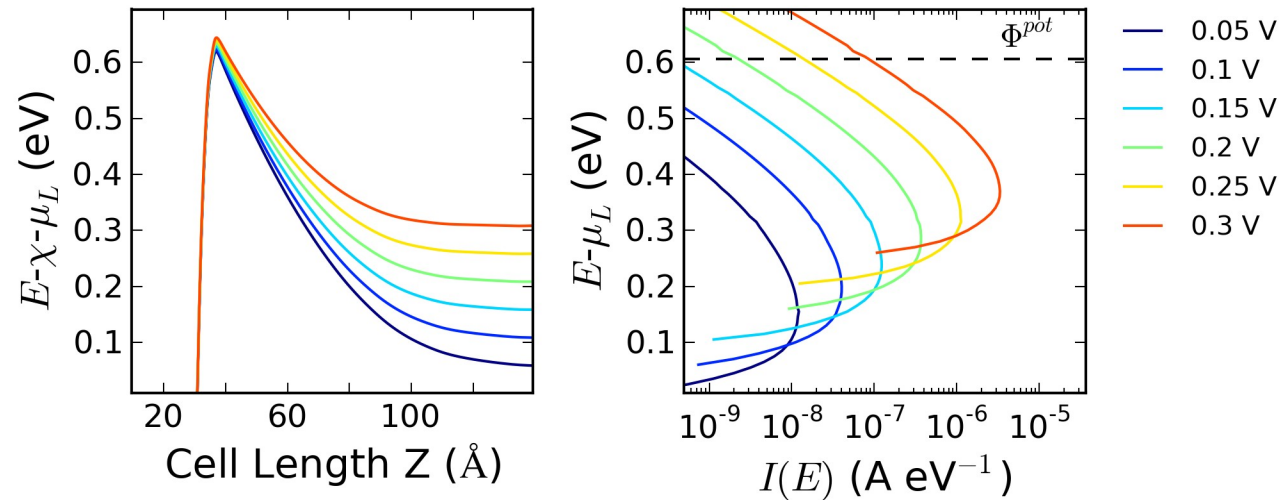
* S.M. Sze and K. N. Kwok. Physics of Semiconductor Devices. Wiley, 3rd edition (2006)

Do a spectral current analysis



Run locally

1. Do an HartreeDifferencePotential analysis for each positive applied voltage using the `ivcurve_selfconsitent_configurations.nc` file
2. Average the potentials using [hdp.py](#)



Do a spectral current analysis



1. Use `spectral_current.py` to plot the HDP and spectral current

The spectral current is defined as

$$I(E) = \frac{2q}{h} T(E, \mu_L, \mu_R) \left[f\left(\frac{E - \mu_L}{k_B T}\right) - f\left(\frac{E - \mu_R}{k_B T}\right) \right]$$

You will need:

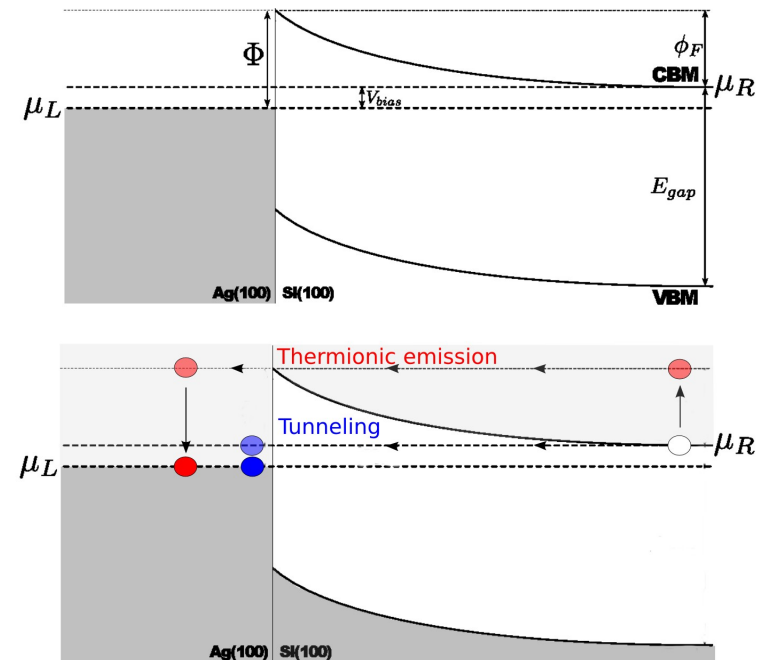
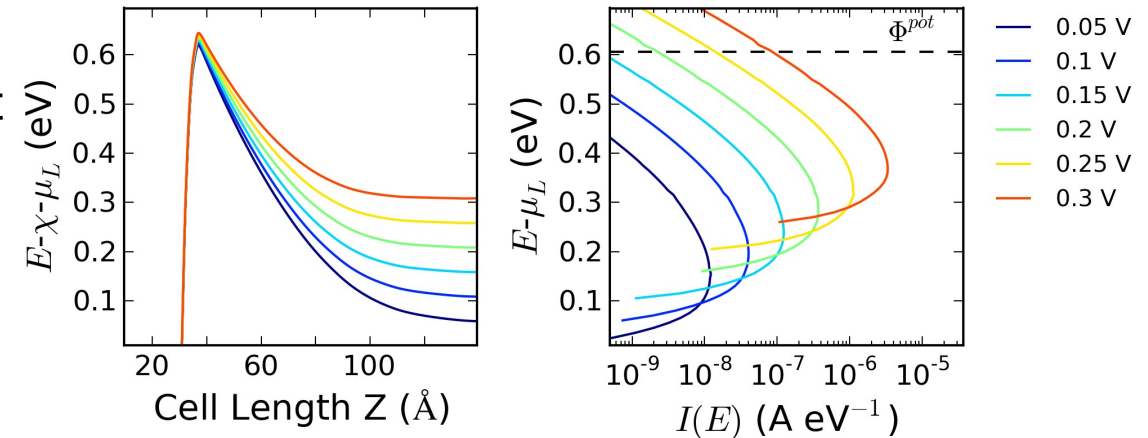
- .dat files containing the averaged EDP from `hdp.py`
- .nc file containing the IVCurve analysis
- The estimated Schottky barrier at zero bias calculated by `pldos_hdp.py`
- CB_{\min} from `pldos_hdp.py`

The script returns:

- The barrier of thermionic emission for each voltage, ϕ_F
- The energy of maximum spectral current for each voltage

2. Analysis

The energy of maximum spectral current is below the thermionic emission barrier which shows a significant part of the transmission occurs through tunneling.





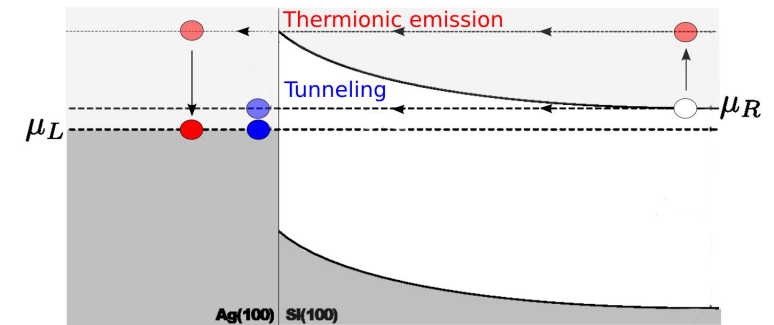
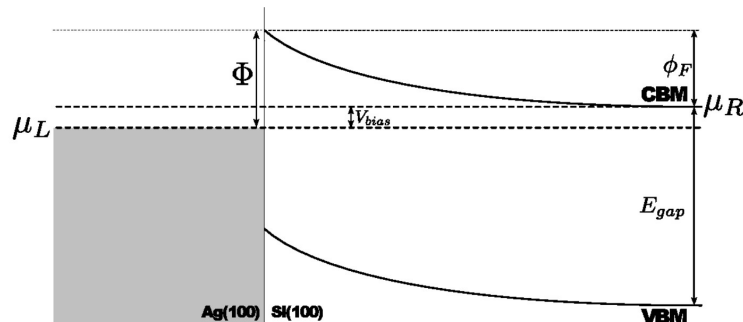
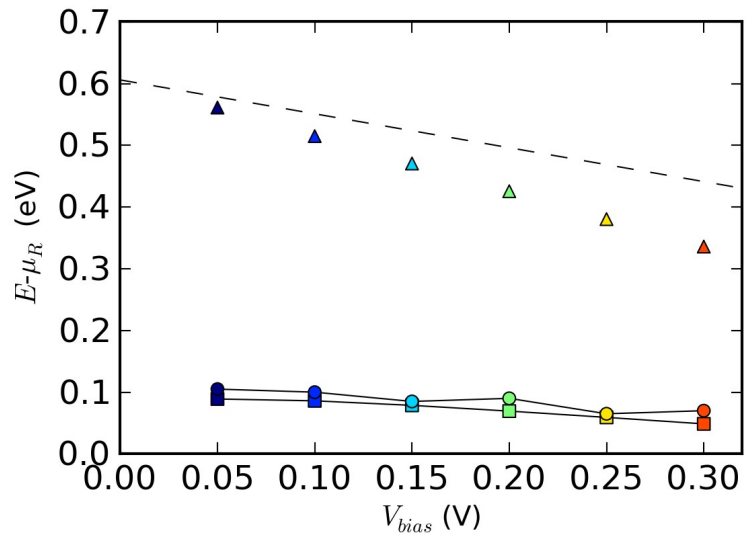
1. Use `barrier_compare.py` to create the plot

You will need:

- File containing the energy of maximum spectral current from `spectral_current.py` (circles)
- File containing the calculated thermionic emission barrier from `spectral_current.py` (triangles)
- File containing the AE Schottky barrier height from `arrhenius.py` (squares)
- The calculated ideality factor from `IV-n-log.py`
- The calculated AE Schottky barrier at zero voltage from `pldos_hdp.py`

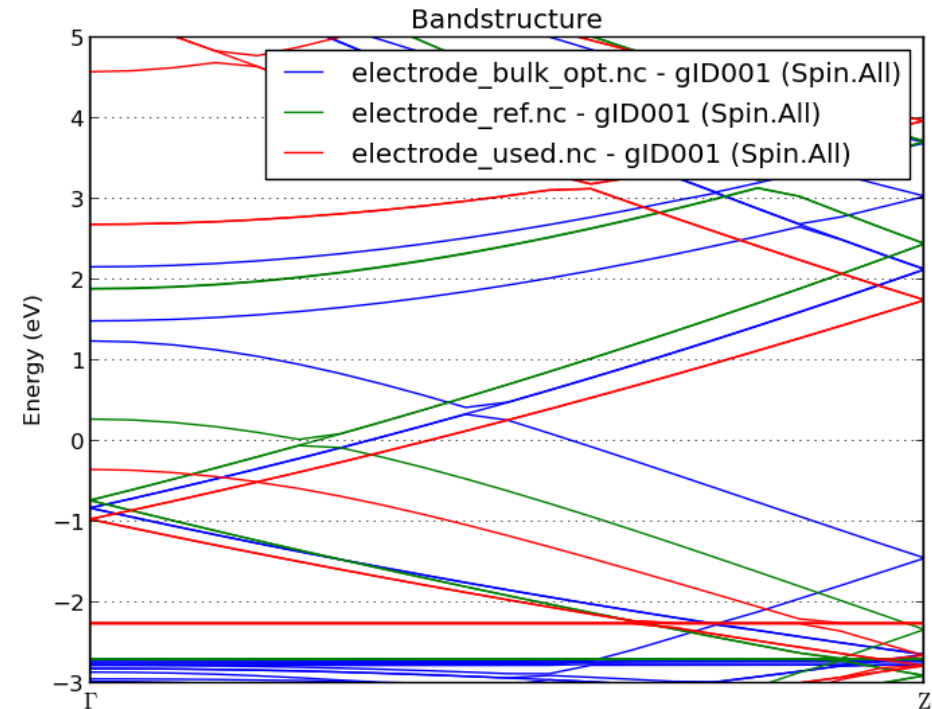
2. Analysis

Using the AE method, we get a Schottky barrier much lower in energy compared to the thermionic emission barrier since the AE method neglects tunneling contributions.





1. The electrode in the article* was not relaxed
2. The electronic structure of the unrelaxed (electrode_ref) and relaxed (electrode_used) electrodes are calculated along with an electrode in the bulk configuration of silver (electrode_bulk_opt)
 - The elongation of the electrode results in lower energy levels
 - Fewer k-points (and thereby states) are available in the bias window
3. A calculation using the structure from the article* with a silicon central region of 120 Å results in higher current as well



I (0.1 V) unrelaxed	I (0.1 V) relaxed
5.537 e-02 nA	3.196 e-04 nA

* Daniele Stradi, Umberto Martinez, Anders Blom, Mads Brandbyge, and Kurt Stokbro. Physical Review B 93, 155302 (2016)



c_fit.py

hdp.py

hdp_plot.py

pldos_hdp.py

IV.py

IV-n-log.py

lvsT.py

arrhenius.py

spectral_current.py

barrier_compare.py