Foundations of Second-Principles DFT

How to run SCALE-UP

Pablo García-Fernández

Santander July 2017

Universidad de Cantabria

What can we do with first-principles simulations?

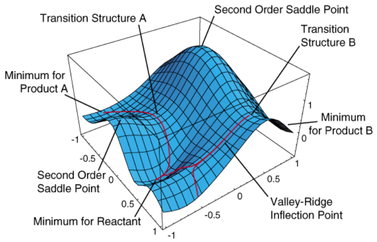
Predict material properties using just fundamental constants

What can we do with first-principles simulations?

Predict material properties using just fundamental constants

Solving the stationary Schrödinger equation (clamped nuclei):

$$\hat{H}\Psi = E\Psi \quad \text{or DFT}$$

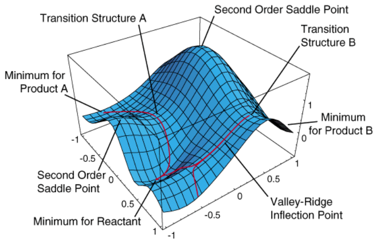


What can we do with first-principles simulations?

Predict material properties using just fundamental constants

Solving the stationary Schrödinger equation (clamped nuclei):

$$\hat{H}\Psi = E\Psi \quad \text{or DFT}$$



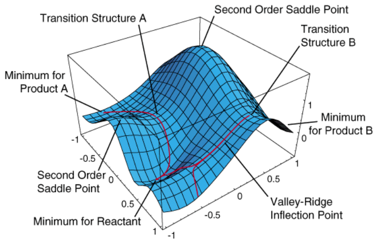
- ✓ Predictive
- ✓ Accurate energies
- ✓ Equilibrium geometries

What can we do with first-principles simulations?

Predict material properties using just fundamental constants

Solving the stationary Schrödinger equation (clamped nuclei):

$$\hat{H}\Psi = E\Psi \quad \text{or DFT}$$



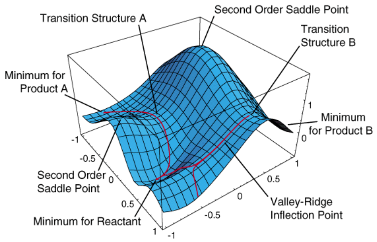
- ✓ Predictive
- ✓ Accurate energies
- ✓ Equilibrium geometries
- $\sim$ Excited states

What can we do with first-principles simulations?

Predict material properties using just fundamental constants

Solving the stationary Schrödinger equation (clamped nuclei):

$$\hat{H}\Psi = E\Psi \quad \text{or DFT}$$



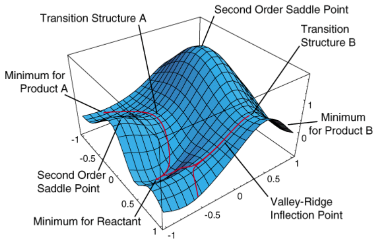
- ✓ Predictive
- ✓ Accurate energies
- ✓ Equilibrium geometries
- ∼ Excited states
- ✗ Temperature
- ✗ Non-adiabaticity
- ✗ Scaling (typically N^3)

What can we do with first-principles simulations?

Predict material properties using just fundamental constants

Solving the stationary Schrödinger equation (clamped nuclei):

$$\hat{H}\Psi = E\Psi \quad \text{or DFT}$$



- ✓ Predictive
- ✓ Accurate energies
- ✓ Equilibrium geometries
- $\sim$ Excited states
- ✗ Temperature
- ✗ Non-adiabaticity
- ✗ Scaling (typically N^3)

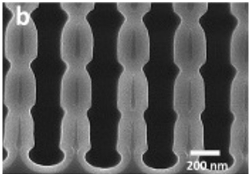
Wealth of information that could be difficult to obtain experimentally

The problem

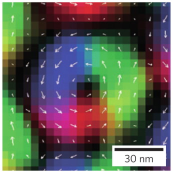
Scale of interest

✓ Nanoscale ($\approx 10 - 100 nm$)

✗ DFT $\approx$ 1 nm



Nanowires



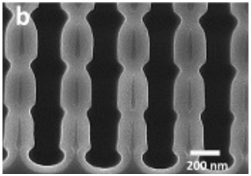
skyrmion in MnSi

The problem

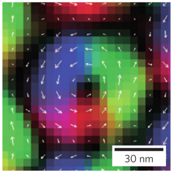
Scale of interest

✓ Nanoscale ($\approx 10-100nm$)

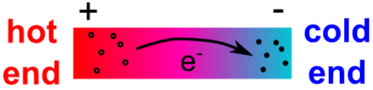
✗ DFT $\approx$ 1 nm



Nanowires



skyrmion in MnSi



Thermoelectrics, polarons...

Non-equilibrium states

✓ Resistivity

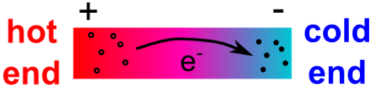
✓ Charge diffusion

The problem

Scale of interest

✓ Nanoscale ($\approx 10 - 100 nm$)

✗ DFT $\approx$ 1 nm



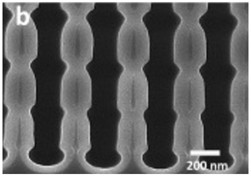
Thermoelectrics, polarons...

Disorder

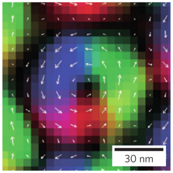
✓ Domains

✓ Thermal

✓ Defects (polarons, impurities...)



Nanowires



skyrmion in MnSi

Non-equilibrium states

✓ Resistivity

✓ Charge diffusion

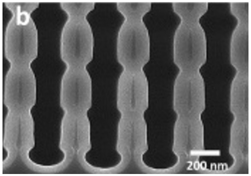
Ferroelectric domains in $BiFeO_3$

The problem

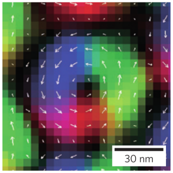
Scale of interest

✓ Nanoscale ($\approx 10 - 100 nm$)

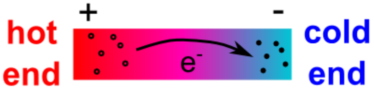
✗ DFT $\approx$ 1 nm



Nanowires



skyrmion in MnSi



Thermoelectrics, polarons...

Non-equilibrium states

✓ Resistivity

✓ Charge diffusion

Disorder

✓ Domains

✓ Thermal

✓ Defects (polarons, impurities...)

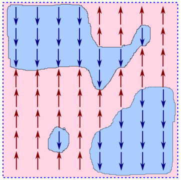
Ferroelectric domains in $BiFeO_3$

- Perturbations/disorder are key elements in experiments.
- Room temperature is fundamental for applications.

Model Hamiltonians

Simple Hamiltonians can be used to simulate large systems

$$E = -\sum_{\{i,j\}} J_{ij} \vec{S}_i \cdot \vec{S}_j$$

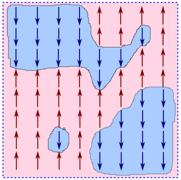


- ✓ Large systems
  We can treat 10^4-10^5 spins
- ✓ Finite temperature
  We can statistically sample our microstates

Model Hamiltonians

Simple Hamiltonians can be used to simulate large systems

$$E = -\sum_{\{i,j\}} J_{ij} \vec{S}_i \cdot \vec{S}_j$$



✓ Large systems

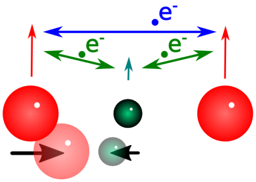
We can treat 10^4-10^5 spins

✓ Finite temperature

We can statistically sample our microstates

✗ Coarse grained

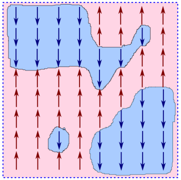
Electrons mediating magnetic interactions?



Model Hamiltonians

Simple Hamiltonians can be used to simulate large systems

$$E = -\sum_{\{i,j\}} J_{ij} \vec{S}_i \cdot \vec{S}_j$$



✓ Large systems

We can treat 10^4-10^5 spins

✓ Finite temperature

We can statistically sample our microstates

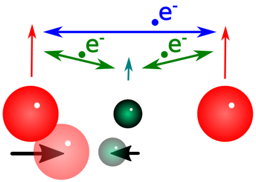
✗ Coarse grained

Electrons mediating magnetic interactions?

✗ Inaccurate

J_i constant with temperature?

Geometry dependence?



Model Hamiltonians: Complications

$$E = -\sum_{\{i,j\}} J_{ij}\vec{S}_i \cdot \vec{S}_j$$

1. What are the foundations of the model?

Model Hamiltonians: Complications

$$E = -\sum_{\{i,j\}} J_{ij} \vec{S}_i \cdot \vec{S}_j$$

1. What are the foundations of the model?
2. How are the parameters determined?

   Experimental $\rightarrow$ Semiempirical

   First-principles $\rightarrow$ Second-principles

Model Hamiltonians: Complications

$$E = -\sum_{\{i,j\}} J_{ij} \vec{S}_i \cdot \vec{S}_j$$

1. What are the foundations of the model?
2. How are the parameters determined?
   Experimental $\rightarrow$ Semiempirical
   First-principles $\rightarrow$ Second-principles
3. Which properties can it be applied to?

Model Hamiltonians: Complications

$$E = -\sum_{\{i,j\}} J_{ij} \vec{S}_i \cdot \vec{S}_j$$

1. What are the foundations of the model?
2. How are the parameters determined?

   Experimental $\rightarrow$ Semiempirical

   First-principles $\rightarrow$ Second-principles
3. Which properties can it be applied to?

Models can reproduce any set of data that they fit...

- They can store information or

Model Hamiltonians: Complications

$$E = -\sum_{\{i,j\}} J_{ij} \vec{S}_i \cdot \vec{S}_j$$

1. What are the foundations of the model?
2. How are the parameters determined?
   Experimental $\rightarrow$ Semiempirical
   First-principles $\rightarrow$ Second-principles
3. Which properties can it be applied to?

Models can reproduce any set of data that they fit...

- They can store information or
- They can predict

Model Hamiltonians: Complications

$$E = -\sum_{\{i,j\}} J_{ij} \vec{S}_i \cdot \vec{S}_j$$

1. What are the foundations of the model?
2. How are the parameters determined?
   Experimental $\rightarrow$ Semiempirical
   First-principles $\rightarrow$ Second-principles
3. Which properties can it be applied to?

Models can reproduce any set of data that they fit...

- They can store information or
- They can predict
- What accuracy?

Model Hamiltonians: Complications

$$E = -\sum_{\{i,j\}} J_{ij}\vec{S}_i \cdot \vec{S}_j$$

1. What are the foundations of the model?
2. How are the parameters determined?
   Experimental $\rightarrow$ Semiempirical
   First-principles $\rightarrow$ Second-principles
3. Which properties can it be applied to?

Models can reproduce any set of data that they fit...

- They can store information or
- They can predict
- What accuracy?

Main questions

- How should we build the model?
- How do we parameterize it?

Model Hamiltonians: Complications

$$E = -\sum_{\{i,j\}} J_{ij} \vec{S}_i \cdot \vec{S}_j$$

1. What are the foundations of the model?
2. How are the parameters determined?
   - Experimental $\rightarrow$ Semiempirical
   - First-principles $\rightarrow$ Second-principles
3. Which properties can it be applied to?

Models can reproduce any set of data that they fit...

- They can store information or
- They can predict
- What accuracy?

Main questions

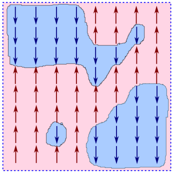
- How should we build the model?
- How do we parameterize it?

Predictive Model Hamiltonians $\rightarrow$ Based on fundamental equations
Well defined physics $\rightarrow$ Ab initio constants/good fitting procedures

Lattice Hamiltonians

Typically used in large-scale simulations of ferroelectrics

$$E = H_{\text{short}}(\{Q\}, \overleftrightarrow{\eta}) + H_{\text{elec}}(\{Q\})$$

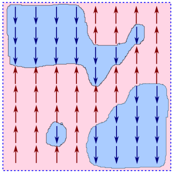


- Based on local polar distortions (Q)
  Distortions obtained from frequency simulation

Lattice Hamiltonians

Typically used in large-scale simulations of ferroelectrics

$$E = H_{\text{short}}(\{Q\}, \overleftrightarrow{\eta}) + H_{\text{elec}}(\{Q\})$$

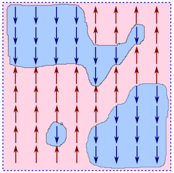


- Based on local polar distortions (Q)
  Distortions obtained from frequency simulation
- Dipole-dipole electrostatics
  Dipole obtained with FP Born charges

Lattice Hamiltonians

Typically used in large-scale simulations of ferroelectrics

$$E = H_{\text{short}}(\{Q\}, \overleftrightarrow{\eta}) + H_{\text{elec}}(\{Q\})$$

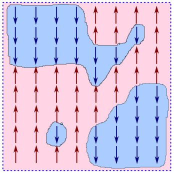


- Based on local polar distortions (Q)
  Distortions obtained from frequency simulation
- Dipole-dipole electrostatics
  Dipole obtained with FP Born charges
- Short-range interactions $\rightarrow$ polynomial expansion
  Parameters can be obtained from few FP calculations

Lattice Hamiltonians

Typically used in large-scale simulations of ferroelectrics

$$E = H_{\text{short}}(\{Q\}, \overleftrightarrow{\eta}) + H_{\text{elec}}(\{Q\})$$

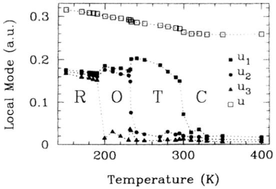


- Based on local polar distortions (Q)
  Distortions obtained from frequency simulation
- Dipole-dipole electrostatics
  Dipole obtained with FP Born charges
- Short-range interactions $\rightarrow$ polynomial expansion
  Parameters can be obtained from few FP calculations

✓ Large time/space scales

✓ Statistics

✓ Domains, superlattices...

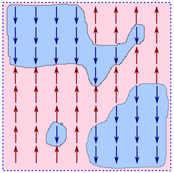


W. Zhong et al.,*PRL*, 94, 1861 (1994)

Lattice Hamiltonians

Typically used in large-scale simulations of ferroelectrics

$$E = H_{\text{short}}(\{Q\}, \overleftrightarrow{\eta}) + H_{\text{elec}}(\{Q\})$$



- Based on local polar distortions (Q)
  Distortions obtained from frequency simulation
- Dipole-dipole electrostatics
  Dipole obtained with FP Born charges
- Short-range interactions → polynomial expansion
  Parameters can be obtained from few FP calculations

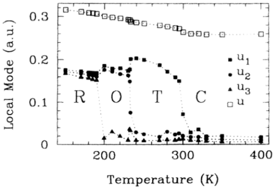
✓ Large time/space scales

✓ Statistics

✓ Domains, superlattices...

✗ Idealized degrees of freedom (not atomistic)

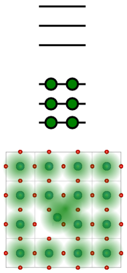
✗ No electrons!



W. Zhong et al.,*PRL*, 94, 1861 (1994)

Are larger/faster FP simulations possible?

First principles simulations deal with all electrons in the system:



Number of electrons grows fast

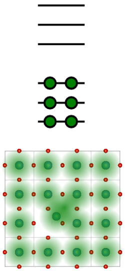
Hamiltonian$\sim N^2$

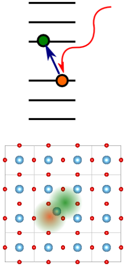
Diagonalization$\sim N^3$

Are larger/faster FP simulations possible?

First principles simulations deal with all electrons in the system:

Number of electrons grows fast
Hamiltonian$\sim N^2$
Diagonalization$\sim N^3$



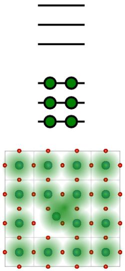


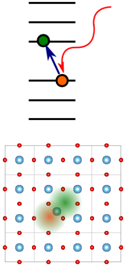
Response to perturbations usually involves a few active electron/holes

Are larger/faster FP simulations possible?

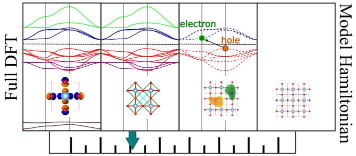
First principles simulations deal with all electrons in the system:

Number of electrons grows fast
Hamiltonian$\sim N^2$
Diagonalization$\sim N^3$





Response to perturbations usually involves a few active electron/holes



Can we select the level of fidelity of our calculations?

Can we make it efficient?

Can we reliably parameterize it?

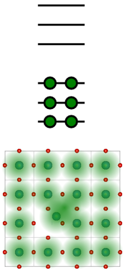
Are larger/faster FP simulations possible?

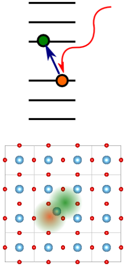
First principles simulations deal with all electrons in the system:

Number of electrons grows fast

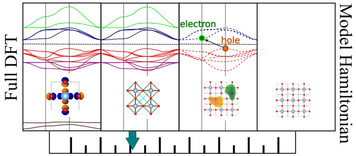
Hamiltonian$\sim N^2$

Diagonalization$\sim N^3$





Response to perturbations usually involves a few active electron/holes



Can we select the level of fidelity of our calculations?

Can we make it efficient?

Can we reliably parameterize it?

Second-principles Density Functional Methods

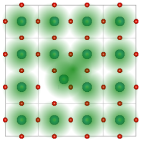
Basic concepts

We want to separate the active electrons that participate in physical properties from all others.

Basic concepts

We want to separate the active electrons that participate in physical properties from all others.

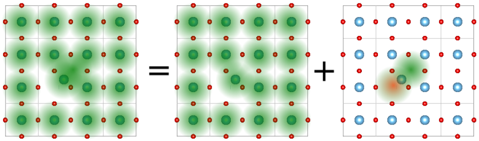
Suppose an insulator doped with electrons or holes:



Basic concepts

We want to separate the active electrons that participate in physical properties from all others.

Suppose an insulator doped with electrons or holes:



The total density is separated in reference and deformation densities:

$$n(\vec{r}) = n_0(\vec{r}) + \delta n(\vec{r})$$

n_0 = reference density
δn = deformation density

Approximating the DFT energy

Our starting point is the DFT energy

$$E_{\mathrm{DFT}} = \sum_{j\vec{k}} o_{j\vec{k}} \left\langle \psi_{j\vec{k}} \right| \hat{t} + v_{\mathrm{ext}} \left| \psi_{j\vec{k}} \right\rangle + \frac{1}{2} \iint \frac{n(\vec{r})n'(\vec{r}')}{|\vec{r}-\vec{r}'|} d^3r d^3r' + E_{\mathrm{xc}}[n] + E_{\mathrm{nn}}$$

We want to write the energy in terms of the reference and deformation densities.

$$n(\vec{r}) = n_0(\vec{r}) + \delta n(\vec{r})$$

Approximating the DFT energy

Our starting point is the DFT energy

$$E_{\mathrm{DFT}} = \sum_{j\vec{k}} o_{j\vec{k}} \left\langle \psi_{j\vec{k}} \right| \hat{t} + v_{\mathrm{ext}} \left| \psi_{j\vec{k}} \right\rangle + \frac{1}{2} \iint \frac{n(\vec{r})n'(\vec{r}')}{|\vec{r}-\vec{r}'|} d^3r d^3r' + E_{\mathrm{xc}}[n] + E_{\mathrm{nn}}$$

We want to write the energy in terms of the reference and deformation densities.

$$n(\vec{r}) = n_0(\vec{r}) + \delta n(\vec{r})$$

The only difficulty is the exchange-correlation energy that we expand in terms of δn
(see e. g. M. Elstner et al.,*Phys. Rev. B*, 58, 7260 (1998)):

$$E_{\mathrm{xc}}[n] = E_{\mathrm{xc}}[n_0] + \int \left. \frac{\delta E_{\mathrm{xc}}}{\delta n(\vec{r})} \right|_{n_0} \delta n(\vec{r}) d^3r + \frac{1}{2} \iint \left. \frac{\delta^2 E_{\mathrm{xc}}}{\delta n(\vec{r})\delta n(\vec{r}')} \right|_{n_0} \delta n(\vec{r})\delta n(\vec{r}') d^3r d^3r' + \cdots$$

Approximating the DFT energy

Our starting point is the DFT energy

$$E_{\mathrm{DFT}} = \sum_{j\vec{k}} o_{j\vec{k}} \left\langle \psi_{j\vec{k}} \right| \hat{t} + v_{\mathrm{ext}} \left| \psi_{j\vec{k}} \right\rangle + \frac{1}{2} \iint \frac{n(\vec{r}) n'(\vec{r}')}{|\vec{r} - \vec{r}'|} d^3r d^3r' + E_{\mathrm{xc}}[n] + E_{\mathrm{nn}}$$

We want to write the energy in terms of the reference and deformation densities.

$$n(\vec{r}) = n_0(\vec{r}) + \delta n(\vec{r})$$

The only difficulty is the exchange-correlation energy that we expand in terms of δn
(see e. g. M. Elstner et al.,*Phys. Rev. B*, 58, 7260 (1998)):

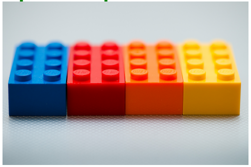
$$E_{\mathrm{xc}}[n] = E_{\mathrm{xc}}[n_0] + \int \left. \frac{\delta E_{\mathrm{xc}}}{\delta n(\vec{r})} \right|_{n_0} \delta n(\vec{r}) d^3r + \frac{1}{2} \iint \left. \frac{\delta^2 E_{\mathrm{xc}}}{\delta n(\vec{r}) \delta n(\vec{r}')} \right|_{n_0} \delta n(\vec{r}) \delta n(\vec{r}') d^3r d^3r' + \cdots$$

As usual in TB-DFT approximations we cut at second-order

$$E_{\mathrm{DFT}} \approx E = E^{(0)} + E^{(1)} + E^{(2)}$$

However, we group the terms in a different way to TB-DFT.

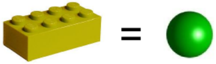
Second-principles DFT approach



Material simulations
allow for various approaches

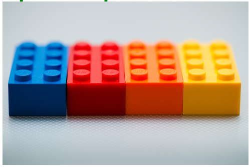
First principles methods are atomistic with flexible detailed bonding

FP or TB-DFT



Based on atoms

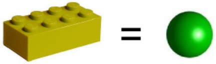
Second-principles DFT approach



Material simulations
allow for various approaches

First principles methods are atomistic with flexible detailed bonding

FP or TB-DFT

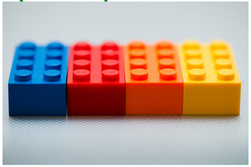


Based on atoms

$$E_{\mathrm{DFT}} \approx \underbrace{E_0}_{\text{atomic cores}} + \underbrace{E_1}_{\text{full 1e energy}} + \underbrace{E_2}_{\text{full 2e energy}} + \dots$$

Atoms $\Longrightarrow$ FP $\Longrightarrow$ Materials

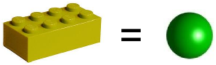
Second-principles DFT approach



Material simulations
allow for various approaches

First principles methods are atomistic with flexible detailed bonding

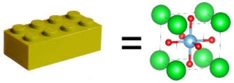
FP or TB-DFT



Based on atoms

$$E_{\mathrm{DFT}} \approx \underbrace{E_0}_{\text{atomic cores}} + \underbrace{E_1}_{\text{full 1e energy}} + \underbrace{E_2}_{\text{full 2e energy}} + \ldots$$

Second-principles



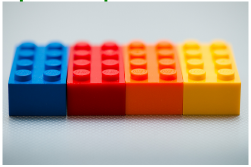
Based on materials

$$E_{\mathrm{DFT}} \approx \underbrace{E^{(0)}}_{\text{lattice}} + \underbrace{E^{(1)} + E^{(2)} + \ldots}_{\text{electron excitations}}$$

Atoms $\Longrightarrow$ FP $\Longrightarrow$ Materials $\Longrightarrow$ SP $\Longrightarrow$ Large-scale

Accurate properties do not require bond-breaking!

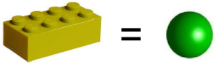
Second-principles DFT approach



Material simulations
allow for various approaches

First principles methods are atomistic with flexible detailed bonding

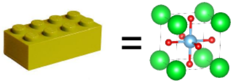
FP or TB-DFT



Based on atoms

$$E_{\mathrm{DFT}} \approx \underbrace{E_0}_{\text{atomic cores}} + \underbrace{E_1}_{\text{full 1e energy}} + \underbrace{E_2}_{\text{full 2e energy}} + \ldots$$

Second-principles



Based on materials

$$E_{\mathrm{DFT}} \approx \underbrace{E^{(0)}}_{\text{lattice}} + \underbrace{E^{(1)} + E^{(2)} + \ldots}_{\text{electron excitations}}$$

Atoms $\Longrightarrow$ FP $\Longrightarrow$ Materials $\Longrightarrow$ SP $\Longrightarrow$ Large-scale

Accurate properties do not require bond-breaking!

Main approximation: Fixed bond topology
A perovskite stays a perovskite, no surface reactivity, etc.

Energy terms: $E = E^{(0)} + E^{(1)} + E^{(2)}$ Reference

This term is the full DFT energy for the reference state:

$$E_0 = \sum_{j\vec{k}} o^0_{j\vec{k}} \left\langle \psi^0_{j\vec{k}} \right| \hat{t} + v_{\text{ext}} \left| \psi^0_{j\vec{k}} \right\rangle + \frac{1}{2} \iint \frac{n_0(\vec{r}) n_0'(\vec{r}')}{|\vec{r} - \vec{r}'|} d^3r d^3r' + E_{\text{xc}}[n_0] + E_{nn}$$

Energy terms: $E = E^{(0)} + E^{(1)} + E^{(2)}$ Reference

This term is the full DFT energy for the reference state:

$$E_0 = \sum_{j\vec{k}} o^0_{j\vec{k}} \left\langle \psi^0_{j\vec{k}} \right| \hat{t} + v_{\text{ext}} \left| \psi^0_{j\vec{k}} \right\rangle + \frac{1}{2} \iint \frac{n_0(\vec{r}) n_0'(\vec{r}')}{|\vec{r} - \vec{r}'|} d^3r d^3r' + E_{\text{xc}}[n_0] + E_{nn}$$

No approximations

At difference with usual TB-DFT this term is very large and contains most of the total energy. It can be made really accurate.

Energy terms: $E = E^{(0)} + E^{(1)} + E^{(2)}$ Reference

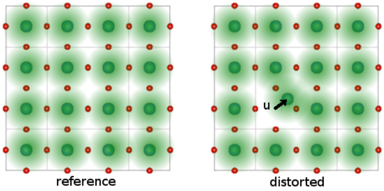
This term is the full DFT energy for the reference state:

$$E_0 = \sum_{j\vec{k}} o^0_{j\vec{k}} \left\langle \psi^0_{j\vec{k}} \middle| \hat{t} + v_{\text{ext}} \middle| \psi^0_{j\vec{k}} \right\rangle + \frac{1}{2} \iint \frac{n_0(\vec{r}) n'_0(\vec{r}')}{|\vec{r} - \vec{r}'|} d^3r d^3r' + E_{xc}[n_0] + E_{nn}$$

No approximations

At difference with usual TB-DFT this term is very large and contains most of the total energy. It can be made really accurate.

Energy is defined as difference from the reference geometry



The reference state is defined for all geometries

$E^{(0)}(\eta, \{\vec{u}\})$ is the energy surface for the reference state

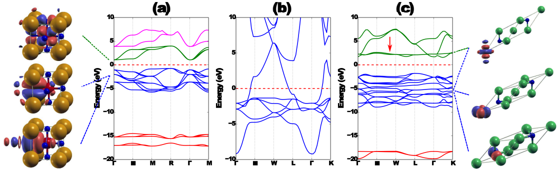
It can be represented by a high-quality model potential.

J. Wojdeł et al., *JPCM*, 25, 305401 (2013)

$E^{(1)} + E^{(2)}$ electron excitation basis

In order to express electron excitations we need basis:

Precise, small, material-adapted $\rightarrow$ Wannier-like functions



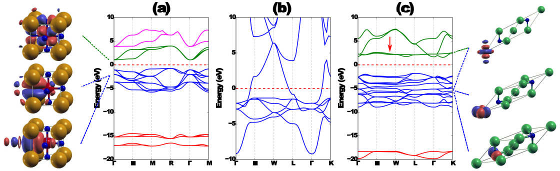
I. Souza et al.,*Phys. Rev. B*, 65, 035109 (2001)

$$|\chi_{\mathbf{a}}\rangle = \frac{V}{(2\pi)^3}\int_{\mathrm{BZ}} d\vec{k} e^{-i\vec{k}\cdot\vec{R}_A}\sum_{m}^{M} B_{a,m\vec{k}}|\psi_{m\vec{k}}\rangle$$

$E^{(1)} + E^{(2)}$ electron excitation basis

In order to express electron excitations we need basis:

Precise, small, material-adapted $\rightarrow$ Wannier-like functions



I. Souza et al.,*Phys. Rev. B*, 65, 035109 (2001)

$$|\chi_{\mathbf{a}}\rangle = \frac{V}{(2\pi)^3}\int_{\mathrm{BZ}} d\vec{k}e^{-i\vec{k}\cdot\vec{R}_A}\sum_{m}^{M} B_{a,m\vec{k}}|\psi_{m\vec{k}}\rangle$$

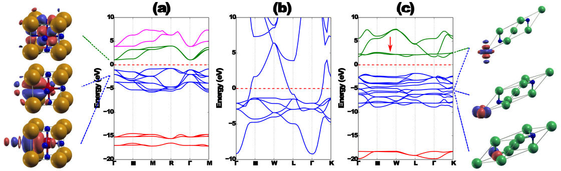
Wannier functions...

1. are localized and so are ideal to build range-limited models

$E^{(1)} + E^{(2)}$ electron excitation basis

In order to express electron excitations we need basis:

Precise, small, material-adapted $\rightarrow$ Wannier-like functions



I. Souza et al.,*Phys. Rev. B*, 65, 035109 (2001)

$$|\chi_{\mathbf{a}}\rangle = \frac{V}{(2\pi)^3}\int_{\mathrm{BZ}} d\vec{k} e^{-i\vec{k}\cdot\vec{R}_A} \sum_{m}^{M} B_{a,m\vec{k}} |\psi_{m\vec{k}}\rangle$$

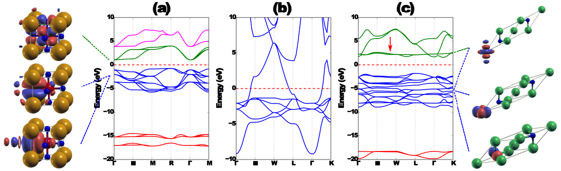
Wannier functions...

1. are localized and so are ideal to build range-limited models
2. allow disentanglement of electron degrees of freedom
   Reduced, meaningful physical models

$E^{(1)} + E^{(2)}$ electron excitation basis

In order to express electron excitations we need basis:

Precise, small, material-adapted $\rightarrow$ Wannier-like functions



I. Souza et al.,*Phys. Rev. B*, 65, 035109 (2001)

$$|\chi_{\mathbf{a}}\rangle = \frac{V}{(2\pi)^3}\int_{\mathrm{BZ}} d\vec{k}e^{-i\vec{k}\cdot\vec{R}_A}\sum_{m}^{M} B_{a,m\vec{k}}|\psi_{m\vec{k}}\rangle$$

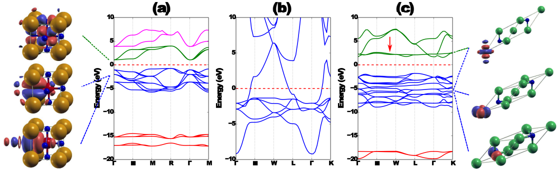
Wannier functions...

1. are localized and so are ideal to build range-limited models
2. allow disentanglement of electron degrees of freedom
   Reduced, meaningful physical models
3. are very accurate even with a small basis

$E^{(1)} + E^{(2)}$ electron excitation basis

In order to express electron excitations we need basis:

Precise, small, material-adapted $\rightarrow$ Wannier-like functions



I. Souza et al.,*Phys. Rev. B*, 65, 035109 (2001)

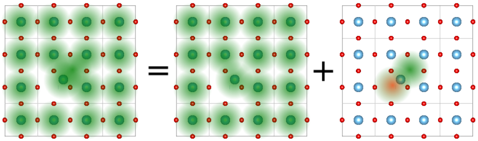
$$|\chi_{\mathbf{a}}\rangle = \frac{V}{(2\pi)^3}\int_{\mathrm{BZ}} d\vec{k}e^{-i\vec{k}\cdot\vec{R}_A}\sum_{m}^{M} B_{a,m\vec{k}}|\psi_{m\vec{k}}\rangle$$

Wannier functions...

1. are localized and so are ideal to build range-limited models
2. allow disentanglement of electron degrees of freedom
   Reduced, meaningful physical models
3. are very accurate even with a small basis
4. are orthogonal
5. are defined for all geometries

$E^{(1)} + E^{(2)}$: Density matrix in Wannier basis

In SP-DFT there are three kinds of densities:

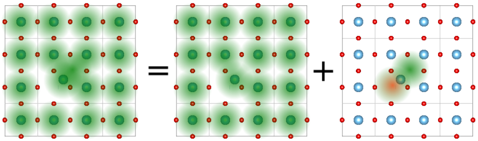


$$n(\vec{r}) = n_0(\vec{r}) + \delta n(\vec{r})$$

How is the density expressed in the WF basis?

$E^{(1)} + E^{(2)}$: Density matrix in Wannier basis

In SP-DFT there are three kinds of densities:



$$n(\vec{r}) = n_0(\vec{r}) + \delta n(\vec{r})$$

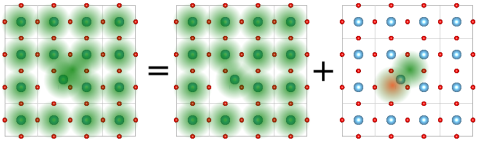
How is the density expressed in the WF basis?

Total	$n(\vec{r}) = \sum_{\mathbf{ab}} d_{\mathbf{ab}} \chi^*_{\mathbf{a}}(\vec{r}) \chi_{\mathbf{b}}(\vec{r})$	$d_{\mathbf{ab}} = \sum_{j\vec{k}} o_{j\vec{k}} e^{i\vec{k}(\vec{R}_B - \vec{R}_A)} c^*_{ja\vec{k}} c_{jb\vec{k}}$
Reference	$n_0(\vec{r}) = \sum_{\mathbf{ab}} d^0_{\mathbf{ab}} \chi^*_{\mathbf{a}}(\vec{r}) \chi_{\mathbf{b}}(\vec{r})$	$d^0_{ab} = \sum_{j\vec{k}} o^0_{j\vec{k}} e^{i\vec{k}(\vec{R}_B - \vec{R}_A)} \left(c^0_{ja\vec{k}}\right)^* c^0_{jb\vec{k}}$
Difference	$\delta n(\vec{r}) = \sum_{\mathbf{ab}} D_{\mathbf{ab}} \chi^*_{\mathbf{a}}(\vec{r}) \chi_{\mathbf{b}}(\vec{r})$	$D_{\mathbf{ab}} = d_{\mathbf{ab}} - d^0_{\mathbf{ab}}$

where $d_{\mathbf{ab}}$, $d^{(0)}_{\mathbf{ab}}$, $D_{\mathbf{ab}}$ are density matrixes (occupations)

$E^{(1)} + E^{(2)}$: Density matrix in Wannier basis

In SP-DFT there are three kinds of densities:



$$n(\vec{r}) = n_0(\vec{r}) + \delta n(\vec{r})$$

How is the density expressed in the WF basis?

Total	$n(\vec{r}) = \sum_{\mathbf{ab}} d_{\mathbf{ab}} \chi^*_{\mathbf{a}}(\vec{r}) \chi_{\mathbf{b}}(\vec{r})$	$d_{\mathbf{ab}} = \sum_{j\vec{k}} o_{j\vec{k}} e^{i\vec{k}(\vec{R}_B - \vec{R}_A)} c^*_{ja\vec{k}} c_{jb\vec{k}}$
Reference	$n_0(\vec{r}) = \sum_{\mathbf{ab}} d^0_{\mathbf{ab}} \chi^*_{\mathbf{a}}(\vec{r}) \chi_{\mathbf{b}}(\vec{r})$	$d^0_{ab} = \sum_{j\vec{k}} o^0_{j\vec{k}} e^{i\vec{k}(\vec{R}_B - \vec{R}_A)} \left(c^0_{ja\vec{k}}\right)^* c^0_{jb\vec{k}}$
Difference	$\delta n(\vec{r}) = \sum_{\mathbf{ab}} D_{\mathbf{ab}} \chi^*_{\mathbf{a}}(\vec{r}) \chi_{\mathbf{b}}(\vec{r})$	$D_{\mathbf{ab}} = d_{\mathbf{ab}} - d^0_{\mathbf{ab}}$

where $d_{\mathbf{ab}}$, $d^{(0)}_{\mathbf{ab}}$, $D_{\mathbf{ab}}$ are density matrixes (occupations)

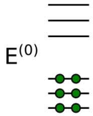
The main variable of the electron part is the Difference density

- $D_{\mathbf{ab}}$ is positive for excited electrons
- $D_{\mathbf{ab}}$ is negative for excited holes

Energy terms: $E = E^{(0)} + E^{(1)} + E^{(2)}$ One electron

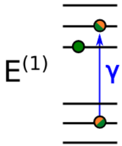
Reference

- Full DFT energy for n_0
- Force field



$E^{(1)}$ contains differences in one-electron energies

$$E^{(1)} = \sum_{j\vec{k}} \left[o_{j\vec{k}} \left\langle \psi_{j\vec{k}} \right| \hat{h}_0 \left| \psi_{j\vec{k}} \right\rangle - o^0_{j\vec{k}} \left\langle \psi^0_{j\vec{k}} \right| \hat{h}_0 \left| \psi^0_{j\vec{k}} \right\rangle \right]$$



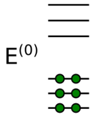
$\hat{h}_0$ is Kohn-Sham Hamiltonian for reference density:

$$\hat{h}_0 = \hat{t} + v_{\text{ext}} + v_{\text{H}}(n_0) + v_{\text{xc}}[n_0]$$

Energy terms: $E = E^{(0)} + E^{(1)} + E^{(2)}$ One electron

Reference

- Full DFT energy for n_0
- Force field

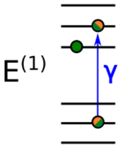


$E^{(1)}$ contains differences in one-electron energies

$$E^{(1)} = \sum_{j\vec{k}} \left[o_{j\vec{k}} \left\langle \psi_{j\vec{k}} \right| \hat{h}_0 \left| \psi_{j\vec{k}} \right\rangle - o^0_{j\vec{k}} \left\langle \psi^0_{j\vec{k}} \right| \hat{h}_0 \left| \psi^0_{j\vec{k}} \right\rangle \right]$$

$$= \sum_{\mathbf{ab}} D_{\mathbf{ab}} \gamma_{\mathbf{ab}} \quad (\text{Wannier basis, } \chi_{\mathbf{a}})$$

$$\delta n(\vec{r}) = \sum_{\mathbf{ab}} D_{\mathbf{ab}} \chi^*_{\mathbf{a}}(\vec{r}) \chi_{\mathbf{b}}(\vec{r})$$



$\hat{h}_0$ is Kohn-Sham Hamiltonian for reference density:

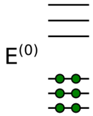
$$\hat{h}_0 = \hat{t} + v_{\text{ext}} + v_{\text{H}}(n_0) + v_{\text{xc}}[n_0]$$

$\gamma_{\mathbf{ab}} = \int d^3 r \chi^*_{\mathbf{a}}(\vec{r}) \hat{h}_0 \chi_{\mathbf{b}}(\vec{r})$ similar to hopping constant in TB

Energy terms: $E = E^{(0)} + E^{(1)} + E^{(2)}$ One electron

Reference

- Full DFT energy for n_0
- Force field

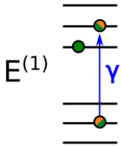


$E^{(1)}$ contains differences in one-electron energies

$$E^{(1)} = \sum_{j\vec{k}} \left[o_{j\vec{k}} \left\langle \psi_{j\vec{k}} \right| \hat{h}_0 \left| \psi_{j\vec{k}} \right\rangle - o^0_{j\vec{k}} \left\langle \psi^0_{j\vec{k}} \right| \hat{h}_0 \left| \psi^0_{j\vec{k}} \right\rangle \right]$$

$$= \sum_{\mathbf{ab}} D_{\mathbf{ab}} \gamma_{\mathbf{ab}} \quad \text{(Wannier basis, } \chi_{\mathbf{a}}\text{)}$$

$$\delta n(\vec{r}) = \sum_{\mathbf{ab}} D_{\mathbf{ab}} \chi^*_{\mathbf{a}}(\vec{r}) \chi_{\mathbf{b}}(\vec{r})$$



Only depends on difference density!

$\hat{h}_0$ is Kohn-Sham Hamiltonian for reference density:

$$\hat{h}_0 = \hat{t} + v_{\text{ext}} + v_{\text{H}}(n_0) + v_{\text{xc}}[n_0]$$

$\gamma_{\mathbf{ab}} = \int d^3 r \chi^*_{\mathbf{a}}(\vec{r}) \hat{h}_0 \chi_{\mathbf{b}}(\vec{r})$ similar to hopping constant in TB

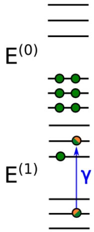
Energy terms: $E = E^{(0)} + E^{(1)} + E^{(2)}$ Two electron

Reference

- Full DFT energy for n_0
- Force field

One-electron

- Depends only on difference density
- Tight-binding like



$E^{(2)}$ are interactions between 2 electrons (E_3 three-electron, etc.):

$$E^{(2)} = \frac{1}{2} \int d^3r \int d^3r' g(\vec{r}, \vec{r}', s, s') \delta n(\vec{r}, s) \delta n(\vec{r}', s')$$

where g is a screened electron-electron interaction operator.

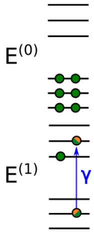
Energy terms: $E = E^{(0)} + E^{(1)} + E^{(2)}$ Two electron

Reference

- Full DFT energy for n_0
- Force field

One-electron

- Depends only on difference density
- Tight-binding like



$E^{(2)}$ are interactions between 2 electrons (E_3 three-electron, etc.):

$$E^{(2)} = \frac{1}{2}\int d^3r \int d^3r' g(\vec{r}, \vec{r}', s, s')\delta n(\vec{r}, s)\delta n(\vec{r}', s')$$

$$= \frac{1}{2}\sum_{\mathbf{ab}}\sum_{\mathbf{a'b'}}\left\{\left[D^{\uparrow}_{\mathbf{ab}} + D^{\downarrow}_{\mathbf{ab}}\right]\left[D^{\uparrow}_{\mathbf{a'b'}} + D^{\downarrow}_{\mathbf{a'b'}}\right]U_{\mathbf{aba'b'}} + \left[D^{\uparrow}_{\mathbf{ab}} - D^{\downarrow}_{\mathbf{ab}}\right]\left[D^{\uparrow}_{\mathbf{a'b'}} - D^{\downarrow}_{\mathbf{a'b'}}\right]I_{\mathbf{aba'b'}}\right\}$$

where g is a screened electron-electron interaction operator.

$E^{(2)}$ only depends the difference density

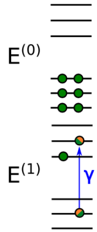
Energy terms: $E = E^{(0)} + E^{(1)} + E^{(2)}$ Two electron

Reference

- Full DFT energy for n_0
- Force field

One-electron

- Depends only on difference density
- Tight-binding like



$E^{(2)}$ are interactions between 2 electrons (E_3 three-electron, etc.):

$$
\begin{aligned}
E^{(2)} =& \frac{1}{2}\int d^3r \int d^3r' g(\vec{r},\vec{r}',s,s')\delta n(\vec{r},s)\delta n(\vec{r}',s') \\
=& \frac{1}{2}\sum_{\mathbf{ab}}\sum_{\mathbf{a'b'}}\Big\{\left[D^{\uparrow}_{\mathbf{ab}} + D^{\downarrow}_{\mathbf{ab}}\right]\left[D^{\uparrow}_{\mathbf{a'b'}} + D^{\downarrow}_{\mathbf{a'b'}}\right]U_{\mathbf{aba'b'}} \\
& + \left[D^{\uparrow}_{\mathbf{ab}} - D^{\downarrow}_{\mathbf{ab}}\right]\left[D^{\uparrow}_{\mathbf{a'b'}} - D^{\downarrow}_{\mathbf{a'b'}}\right]I_{\mathbf{aba'b'}}\Big\}
\end{aligned}
$$

where g is a screened electron-electron interaction operator.

$E^{(2)}$ only depends the difference density

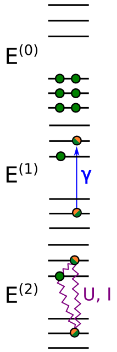
The term in I depends on the spin polarization

Energy terms: $E = E^{(0)} + E^{(1)} + E^{(2)}$ Calculation

Reference
- Full DFT energy for n_0
- Force field

One-electron
- Depends only on difference density
- Tight-binding like

Two-electron
- Depends only on difference density
- Screened mean-field interactions



- Accurate
- Fast
- Valid for all kind of systems (magnetic, metallic, ...)

Efficiency of solid-state methods

Hartree-Fock

$$\hat{h} = \hat{t} + v[\{\psi_i\}](\vec{r})$$

Non-local exchange

N^5 scaling

Efficiency of solid-state methods

Hartree-Fock

$$\hat{h} = \hat{t} + v[\{\psi_i\}](\vec{r})$$

Non-local exchange

N^5 scaling

DFT

$$\hat{h} = \hat{t} + v[n](\vec{r})$$

Local xc functionals

N^3 scaling

Efficiency of solid-state methods

Hartree-Fock	DFT	SP-DFT
$\hat{h} = \hat{t} + v[\{\psi_i\}](\vec{r})$	$\hat{h} = \hat{t} + v[n](\vec{r})$	$h_{\mathbf{ab}} = \gamma_{\mathbf{ab}} + \sum_{\mathbf{a'b'}} D_{\mathbf{a'b'}} U_{\mathbf{aba'b'}}$
Non-local exchange	Local xc functionals	Equations non-local
N^5 scaling	N^3 scaling	N^4 scaling??

SP-DFT is not intrinsically fast unless models are range-limited

Efficiency of solid-state methods

Hartree-Fock

$$\hat{h} = \hat{t} + v[\{\psi_i\}](\vec{r})$$

Non-local exchange

N^5 scaling

DFT

$$\hat{h} = \hat{t} + v[n](\vec{r})$$

Local xc functionals

N^3 scaling

SP-DFT

$$h_{\mathbf{ab}} = \gamma_{\mathbf{ab}} + \sum_{\mathbf{a'b'}} D_{\mathbf{a'b'}} U_{\mathbf{aba'b'}}$$

Equations non-local

N^4 scaling??

SP-DFT is not intrinsically fast unless models are range-limited

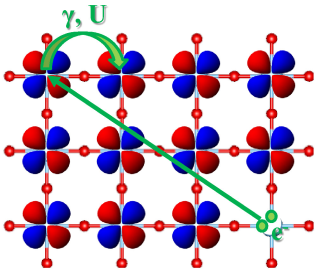
1. Accuracy is given by including many terms ($\gamma_{\mathbf{ab}}$, $U_{\mathbf{aba'b'}}$)
2. Speed is dictated by having range-limited interactions

Model building is a key step in the use of SP-DFT!

Electrostatics allows reducing number of parameters in simulation

Electrostatics

All interactions in the model are between localized objects:



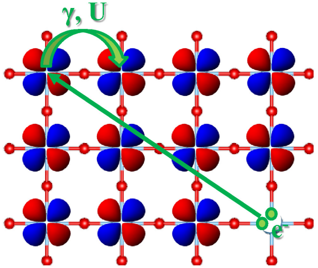
Atomic displacement $\rightarrow$ local dipole

γ and U contain electrostatic (long-range) contributions

Hartree/electron-nucleus

Electrostatics

All interactions in the model are between localized objects:



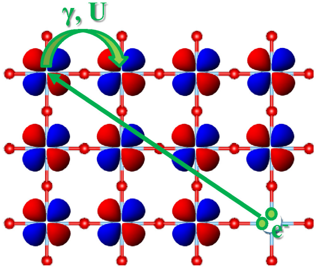
Atomic displacement $\rightarrow$ local dipole
γ and U contain electrostatic (long-range) contributions
Hartree/electron-nucleus

At long-range (far-field regime) shape of source density is unimportant
Multipolar expansion!

We approximate the full charge density by a field of point charges and dipoles localized at the reference geometry

Electrostatics

All interactions in the model are between localized objects:



Atomic displacement $\rightarrow$ local dipole
γ and U contain electrostatic (long-range) contributions
Hartree/electron-nucleus

At long-range (far-field regime) shape of source density is unimportant
Multipolar expansion!

We approximate the full charge density by a field of point charges and dipoles localized at the reference geometry

Model parameters are separated in long and short contributions.

The short-range part $\rightarrow$ quickly converging to zero
It also creates forces - long-range electron-lattice interaction

What can SP-DFT realistically do?

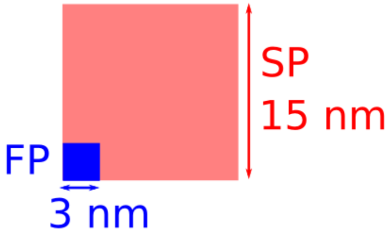
Good test $\rightarrow$ impurity concentration

Semiconductors $\rightarrow$ 10^{13}cm^{-3}-10^{18}cm^{-3}

What can SP-DFT realistically do?

Good test → impurity concentration

Semiconductors → 10^{13}cm^{-3}-10^{18}cm^{-3}



In FP size limit ∼700 atoms (3nm/cube side)

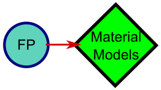
FP impurity density $\approx 3.7 \cdot 10^{19} cm^{-3}$

SP detailed model → 30000 atoms (12nm/cube side)
SP simple model → 60000 atoms (15nm/cube side)

SP impurity density $\approx 3.0 \cdot 10^{17} cm^{-3}$

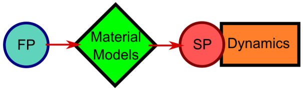
The SP-DFT goal

Our goal is getting closer to the ideal of computer experiments...



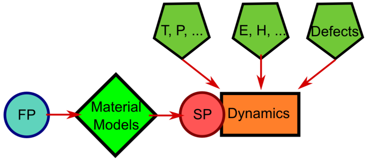
The SP-DFT goal

Our goal is getting closer to the ideal of computer experiments...



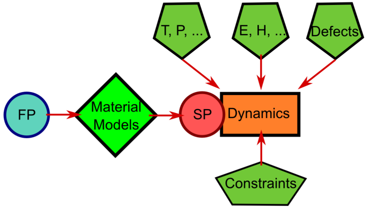
The SP-DFT goal

Our goal is getting closer to the ideal of computer experiments...



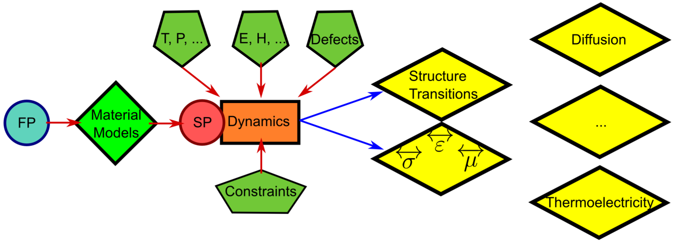
The SP-DFT goal

Our goal is getting closer to the ideal of computer experiments...



The SP-DFT goal

Our goal is getting closer to the ideal of computer experiments...



... with truly polyvalent models and the ability to understand and predict a wide variety of properties!

The implementation of SP-DFT: SCALE-UP

P. Garcia-Fernandez, J. Wojdeł, J. Iñiguez and J. Junquera
Phys. Rev. B, **93**, 195137 (2016)

- Fully integrated electron+lattice models

The implementation of SP-DFT: Scale-Up

P. Garcia-Fernandez, J. Wojdeł, J. Iñiguez and J. Junquera
Phys. Rev. B, **93**, 195137 (2016)

- Fully integrated electron+lattice models
- Single-points, Dynamics (NVT, Langevin), Montecarlo

The implementation of SP-DFT: Scale-Up

P. Garcia-Fernandez, J. Wojdeł, J. Iñiguez and J. Junquera
Phys. Rev. B, **93**, 195137 (2016)

- Fully integrated electron+lattice models
- Single-points, Dynamics (NVT, Langevin), Montecarlo
- SCF with convergence acelerators
- TDDFT: Density propagation, $\vec{E}(t)$ fields

The implementation of SP-DFT: Scale-Up

P. Garcia-Fernandez, J. Wojdeł, J. Iñiguez and J. Junquera
Phys. Rev. B, **93**, 195137 (2016)

- Fully integrated electron+lattice models
- Single-points, Dynamics (NVT, Langevin), Montecarlo
- SCF with convergence acelerators
- TDDFT: Density propagation, $\vec{E}(t)$ fields
- Fortran 90 code + python utilities and interface
- Parallelization: Hybrid scheme, MPI+OPENMP

The implementation of SP-DFT: Scale-Up

P. Garcia-Fernandez, J. Wojdeł, J. Iñiguez and J. Junquera
Phys. Rev. B, **93**, 195137 (2016)

- Fully integrated electron+lattice models
- Single-points, Dynamics (NVT, Langevin), Montecarlo
- SCF with convergence acelerators
- TDDFT: Density propagation, $\vec{E}(t)$ fields
- Fortran 90 code + python utilities and interface
- Parallelization: Hybrid scheme, MPI+OPENMP
- Model building suite

The implementation of SP-DFT: Scale-Up

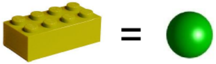
P. Garcia-Fernandez, J. Wojdeł, J. Iñiguez and J. Junquera
Phys. Rev. B, **93**, 195137 (2016)

- Fully integrated electron+lattice models
- Single-points, Dynamics (NVT, Langevin), Montecarlo
- SCF with convergence acelerators
- TDDFT: Density propagation, $\vec{E}(t)$ fields
- Fortran 90 code + python utilities and interface
- Parallelization: Hybrid scheme, MPI+OPENMP
- Model building suite
- Future: Spin-orbit, defects

Running Scale-Up

Running is analogous to ab initio code with a key difference

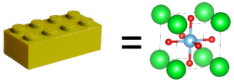
First-principles



Based on atoms

Pseudopotential/basis files

Second-principles



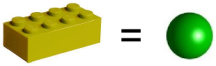
Based on materials

Material xml file

Running Scale-Up

Running is analogous to ab initio code with a key difference

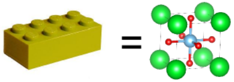
First-principles



Based on atoms

Pseudopotential/basis files

Second-principles



Based on materials

Material xml file

There are two main ways of running Scale-Up :

1. Calling the code from the terminal, requires input file

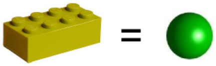
```
$ more input.fdf
System_name a_linear_chain
Parameter_file linear_chain.xml
%block Supercell
1 1 2
%endblock Supercell
```

```
$ scaleup.x < input.fdf > output
```

Running Scale-Up

Running is analogous to ab initio code with a key difference

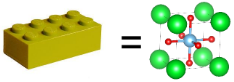
First-principles



Based on atoms

Pseudopotential/basis files

Second-principles



Based on materials

Material xml file

There are two main ways of running Scale-Up :

1. Calling the code from the terminal, requires input file

```
$ more input.fdf
System_name a_linear_chain
Parameter_file linear_chain.xml
%block Supercell
1 1 2
%endblock Supercell
```

```
$ scaleup.x < input.fdf > output
```

2. Using the python scaleup module in myscript.py

```
$ python myscript.py
```

Running SCALE-UP from the terminal

SCALE-UP uses one environment variable:

SCALEUPHOME $\rightarrow$ points to SCALE-UP root directory

```
$ ls $SCALEUPHOME
bin dev docs interface scripts src test
```

Running Scale-Up from the terminal

Scale-Up uses one environment variable:

SCALEUPHOME → points to Scale-Up root directory

```
$ ls $SCALEUPHOME
bin dev docs interface scripts src test
```

1. bin contains the executable

   ```
   $ $SCALEUPHOME/bin/scaleup.x < input.fdf > output
   ```

Running Scale-Up from the terminal

Scale-Up uses one environment variable:

SCALEUPHOME → points to Scale-Up root directory

```
$ ls $SCALEUPHOME
bin dev docs interface scripts src test
```

1. bin contains the executable

   ```
   $ $SCALEUPHOME/bin/scaleup.x < input.fdf > output
   ```

2. docs contains the manual

   ```
   $ evince $SCALEUPHOME/docs/manual.pdf &
   ```

Running SCALE-UP from the terminal

SCALE-UP uses one environment variable:
SCALEUPHOME $\rightarrow$ points to SCALE-UP root directory

```
$ ls $SCALEUPHOME
bin dev docs interface scripts src test
```

1. bin contains the executable

   ```
   $ $SCALEUPHOME/bin/scaleup.x < input.fdf > output
   ```

2. docs contains the manual

   ```
   $ evince $SCALEUPHOME/docs/manual.pdf &
   ```

3. scripts contains some utils

   ```
   $ python $SCALEUPHOME/scripts/scaleup_utils.py -dos
   ```

Running SCALE-UP from the terminal

SCALE-UP uses one environment variable:

SCALEUPHOME → points to SCALE-UP root directory

```
$ ls $SCALEUPHOME
bin dev docs interface scripts src test
```

1. bin contains the executable
   ```
   $ $SCALEUPHOME/bin/scaleup.x < input.fdf > output
   ```
2. docs contains the manual
   ```
   $ evince $SCALEUPHOME/docs/manual.pdf &
   ```
3. scripts contains some utils
   ```
   $ python $SCALEUPHOME/scripts/scaleup_utils.py -dos
   ```
4. src contains the source code
5. test contains examples and tutorials
6. interface contains the python scaleup.py module
7. dev contains some tools used during development

The input file: fdf format

SCALE-UP input is given in text file using SIESTA's fdf format...

- Keywords: A keyword string and a value (integer, real, word)

  Geometrymode single-point
  MaximumSCFiter 100

The input file: fdf format

SCALE-UP input is given in text file using SIESTA's fdf format...

- Keywords: A keyword string and a value (integer, real, word)

  ```
  Geometrymode single-point
  MaximumSCFiter 100
  ```

- Blocks: A chunck of information that is read together. It starts with %block and ends with %endblock.

  ```
  %block band_path
  2
  30    0.0 0.0 0.0    0.0 0.0 0.5
  30    0.0 0.0 0.5    0.0 0.5 0.5
  %endblock band_path
  ```

The input file: fdf format

Scale-Up input is given in text file using SIESTA's fdf format...

- Keywords: A keyword string and a value (integer, real, word)

  ```
  Geometrymode single-point
  MaximumSCFiter 100
  ```

- Blocks: A chunck of information that is read together. It starts with %block and ends with %endblock.

  ```
  %block band_path
  2
  30     0.0 0.0 0.0     0.0 0.0 0.5
  30     0.0 0.0 0.5     0.0 0.5 0.5
  %endblock band_path
  ```

- Physical: A keyword string with value (real) and the unit.

  ```
  Temperature 100.0 kelvin
  ```

Extra auxiliary files may be required to input initial geometries, etc.

Basic input and running

SCALE-UP can be run as a serial program:

```
$ $SCALEUPHOME/bin/scaleup_serial.x < input.fdf > output
```

or as a full parallel MPI Fortran program:

```
$ mpirun -n 4 $SCALEUPHOME/bin/scaleup.x < input.fdf > output
```

Your current compilation is serial only!

Basic input and running

SCALE-UP can be run as a serial program:

```
$ $SCALEUPHOME/bin/scaleup_serial.x < input.fdf > output
```

or as a full parallel MPI Fortran program:

```
$ mpirun -n 4 $SCALEUPHOME/bin/scaleup.x < input.fdf > output
```

Your current compilation is serial only!

The result of a simulation is:

- Output file human readable plain-text with information about the SCF steps, convergence, etc.
- Auxiliary files starting with the System_name that can be read by the scaleup_utils.py.

Basic input and running

SCALE-UP can be run as a serial program:

```
$ $SCALEUPHOME/bin/scaleup_serial.x < input.fdf > output
```

or as a full parallel MPI Fortran program:

```
$ mpirun -n 4 $SCALEUPHOME/bin/scaleup.x < input.fdf > output
```

Your current compilation is serial only!

The result of a simulation is:

- Output file human readable plain-text with information about the SCF steps, convergence, etc.
- Auxiliary files starting with the System_name that can be read by the scaleup_utils.py.

Let's have a look at the main keywords!

Minimum input

A minimum Scale-Up input contains:

1. system_name name Output files will start with name
2. parameter_file filename Material parameters in filename
3. % block Supercell The size of the system

Minimum input

A minimum SCALE-UP input contains:

1. system_name name Output files will start with name
2. parameter_file filename Material parameters in filename
3. % block Supercell The size of the system

The value to most parameters include sensible defaults

Minimum input

A minimum Scale-Up input contains:

1. system_name name Output files will start with name
2. parameter_file filename Material parameters in filename
3. % block Supercell The size of the system

The value to most parameters include sensible defaults

Important keywords include:

1. No_lattice Do not read parameters for $E^{(0)}$ from file
2. No_electron Do not read parameters for $E^{(1)}$ or $E^{(2)}$ from file

Minimum input

A minimum SCALE-UP input contains:

1. system_name name Output files will start with name
2. parameter_file filename Material parameters in filename
3. % block Supercell The size of the system

The value to most parameters include sensible defaults

Important keywords include:

1. No_lattice Do not read parameters for $E^{(0)}$ from file
2. No_electron Do not read parameters for $E^{(1)}$ or $E^{(2)}$ from file
3. mode
    - single_point (default)
    - monte_carlo
    - dynamics
    - optimization

Basic keywords

Some useful keywords are:

- geometry_restart filename read initial geometry
- read_orbocc filename read initial orbital occupation
- print_bands int print bands file
- print_forces int output the forces to file
- temperate real kelvin The temperature
- %block static_electric_field The electric field

Basic keywords

Some useful keywords are:

- geometry_restart filename read initial geometry
- read_orbocc filename read initial orbital occupation
- print_bands int print bands file
- print_forces int output the forces to file
- temperate real kelvin The temperature
- %block static_electric_field The electric field

SCF keywords:

- Magnetic Carry out spin-polarized calculations
- MaximumSCFiter int Maximum SCF iterations before aborting
- SCFthreshold real convergence criterion for SCF (def. 10^{-3})
- SCFmixing real amount of new density matrix mixed with old
- StartPulay int iterations when Pulay mixing kicks in (min 3)

Basic keywords

Some useful keywords are:

- geometry_restart filename read initial geometry
- read_orbocc filename read initial orbital occupation
- print_bands int print bands file
- print_forces int output the forces to file
- temperate real kelvin The temperature
- %block static_electric_field The electric field

SCF keywords:

- Magnetic Carry out spin-polarized calculations
- MaximumSCFiter int Maximum SCF iterations before aborting
- SCFthreshold real convergence criterion for SCF (def. 10^{-3})
- SCFmixing real amount of new density matrix mixed with old
- StartPulay int iterations when Pulay mixing kicks in (min 3)

Check all the keywords in the manual!

```
$ evince $SCALEUPHOME/docs/manual.pdf &
```

The visualization tools

The visualization tools are in $SCALEUPHOME/scripts:

scaleup_utils.py

```
$ python $SCALEUPHOME/scripts/scaleup_utils.py -help
```

The visualization tools

The visualization tools are in $SCALEUPHOME/scripts:

scaleup_utils.py

```
$ python $SCALEUPHOME/scripts/scaleup_utils.py -help
```

1. Geometry takes .REF and .restart files and outputs xcrysden
   - Allows representing absolute geometry
   - Allows representing reference geometry and distortion
2. DOS takes a .ener file and plots the DOS
3. bands takes a .bands file and plots the bands
4. current takes current/polarization file and plots conductivity
5. other we are expanding it to take:
   - .elec files to plot magnetization, electrostatic potentials, atomic dipoles in xcrysden format
   - .orbocc files to plot electron and hole densities in space

The python interface

SCALE-UP contains a python interface to the Fortran code

Check: $PYTHONPATH="$SCALEUPHOME/interface"

The python interface

SCALE-UP contains a python interface to the Fortran code

Check: $PYTHONPATH="$SCALEUPHOME/interface"

To use it simple load the py_scaleup module:

```
$ more my_python_scup_script.py
"""
A simple python script using the py_scaleup module
"""
import py_scaleup as sclup
# Initialize an scale-up simulation of SrTiO3 in a 2x2x1 supercell
# with just lattice degrees of freedom
 scaleupsim=sclup.scaleup('srtio3_full_lat.xml',supercell=[2,2,1],
lattice=True,electrons=False)
# Change an atom position and calculate energy
scaleupsim.displacements[9]=0.1
energy=scaleupsim.get_energy()
```

At present functionality is limited but will quickly be expanded

The python interface

SCALE-UP contains a python interface to the Fortran code

Check: $PYTHONPATH="$SCALEUPHOME/interface"

To use it simple load the py_scaleup module:

```
$ more my_python_scup_script.py
"""
A simple python script using the py_scaleup module
"""
import py_scaleup as sclup
# Initialize an scale-up simulation of SrTiO3 in a 2x2x1 supercell
# with just lattice degrees of freedom
 scaleupsim=sclup.scaleup('srtio3_full_lat.xml',supercell=[2,2,1],
lattice=True,electrons=False)
# Change an atom position and calculate energy
scaleupsim.displacements[9]=0.1
energy=scaleupsim.get_energy()
```

At present functionality is limited but will quickly be expanded

Check definition of the scaleup class!

```
$ vi $SCALEUPHOME/interface/py_scaleup.py
```