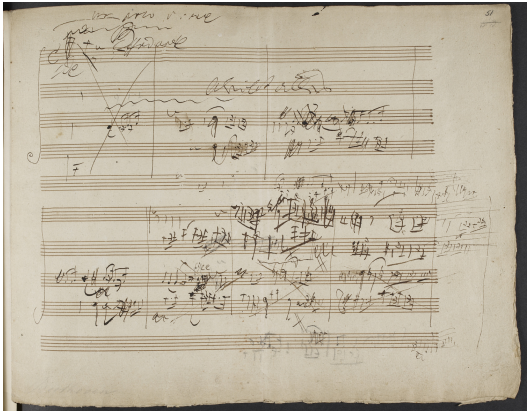
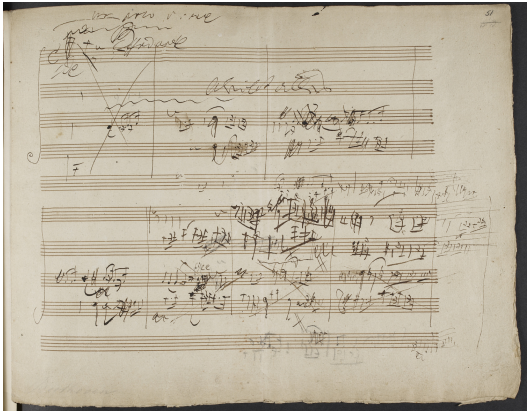
String Quartet No. 14 - Adagio ma non troppo e molto espressivo



click here for video

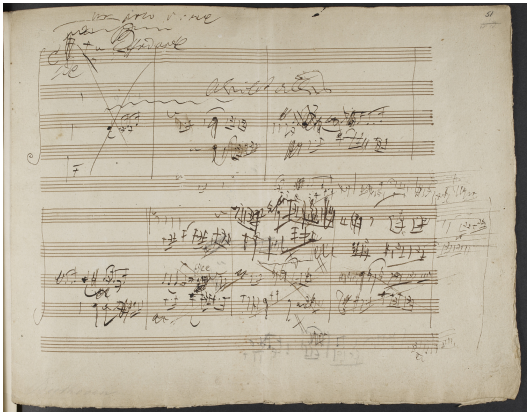
String Quartet No. 14 - Adagio ma non troppo e molto espressivo



click here for video

"indecipherable, uncorrected horrors" – Spohr

String Quartet No. 14 - Adagio ma non troppo e molto espressivo



click here for video

"indecipherable, uncorrected horrors" – Spohr
"After this, what is left for us to write?" – Schubert

Towards black-box Koopmans band structures

or: getting lost down a pseudopotential-generation rabbit hole

Edward Linscott | PSI | THEOS Group Meeting | 11 April 2024 |

Swiss National Science Foundation
MARVEL

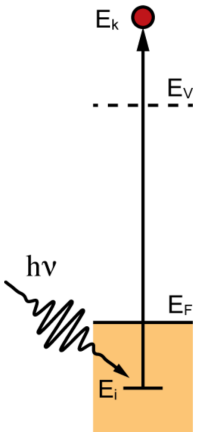
Outline

- a quick recap of Koopmans functionals
- automated Wannierisation and in the context of Koopmans functionals
- generating better PAOs for automated Wannierisation
- how to make this framework systematic and accessible

A quick recap of Koopmans functionals

A quick recap of Koopmans functionals

How can we calculate the energies of charged excitations? Why does DFT fail?



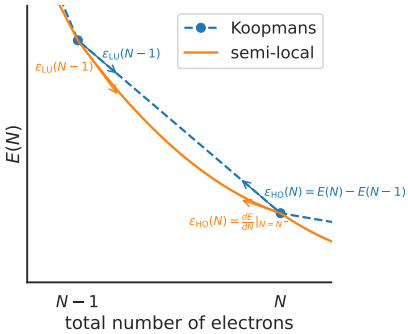
R. Costantini et al. “Exciton Dynamics in Molecular Heterojunctions”. PhD thesis. Universita degli Studi di Trieste, 2020

A quick recap of Koopmans functionals

How can we calculate the energies of charged excitations? Why does DFT fail?

For the exact Green's function, we have poles that correspond to total energy differences

$$\varepsilon_i = \begin{cases} E(N) - E_i(N-1) & i \in \text{occ} \\ E_i(N+1) - E(N) & i \in \text{emp} \end{cases}$$



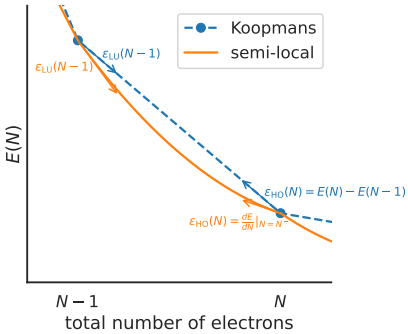
A quick recap of Koopmans functionals

How can we calculate the energies of charged excitations? Why does DFT fail?

For the exact Green's function, we have poles that correspond to total energy differences

$$\varepsilon_i = \begin{cases} E(N) - E_i(N-1) & i \in \text{occ} \\ E_i(N+1) - E(N) & i \in \text{emp} \end{cases}$$

For DFT, this condition is *not* satisfied in general



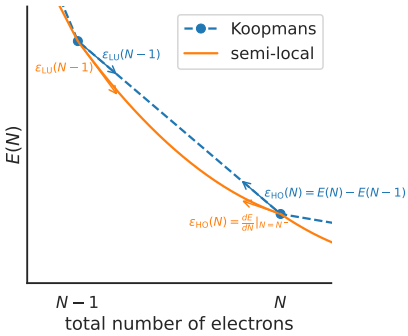
A quick recap of Koopmans functionals

Core idea: for every orbital i their energy

$$\varepsilon_i^{\text{Koopmans}} = \langle \varphi_i | H | \varphi_i \rangle = \partial E_{\text{Koopmans}} / \partial f_i$$

ought to be...

- independent of its own occupation f_i
- equal to the corresponding total energy difference $E_i(N-1) - E(N)$



R. Costantini et al. “Exciton Dynamics in Molecular Heterojunctions”. PhD thesis. Universita degli Studi di Trieste, 2020

A quick recap of Koopmans functionals

$$E_{\mathrm{KI}}[\rho, \{\rho_i\}, \{\alpha_i\}] = E_{\mathrm{DFT}}[\rho] + \sum_i \Bigg(\underbrace{-\left(E_{\mathrm{DFT}} - E_{\mathrm{DFT}}|_{f_i=0}\right)}_{\text{removes erroneous curvature}} + \underbrace{f_i \left(E_{\mathrm{DFT}}|_{f_i=1} - E_{\mathrm{DFT}}|_{f_i=0}\right)}_{\text{restores linear behaviour}} \Bigg)$$

General features:

A quick recap of Koopmans functionals

$$E_{\mathrm{KI}}[\rho, \{\rho_i\}, \{\alpha_i\}] = E_{\mathrm{DFT}}[\rho] + \sum_i \Bigg(\underbrace{- \left(E_{\mathrm{DFT}} - E_{\mathrm{DFT}}|_{f_i=0}\right)}_{\text{removes erroneous curvature}} + \underbrace{f_i \left(E_{\mathrm{DFT}}|_{f_i=1} - E_{\mathrm{DFT}}|_{f_i=0}\right)}_{\text{restores linear behaviour}} \Bigg)$$

General features:

- a correction to DFT that ensures eigenvalues match total energy differences

A quick recap of Koopmans functionals

$$E_{\mathrm{KI}}[\rho, \{\rho_i\}, \{\alpha_i\}] = E_{\mathrm{DFT}}[\rho] + \sum_i \alpha_i \Bigg(\underbrace{E_{\mathrm{Hxc}}[\rho - \rho_i] - E_{\mathrm{Hxc}}[\rho]}_{\text{removes erroneous curvature}} + \underbrace{f_i \left(E_{\mathrm{Hxc}}[\rho - \rho_i + n_i] - E_{\mathrm{Hxc}}[\rho - \rho_i] \right)}_{\text{restores linear behaviour}} \Bigg)$$

General features:

- a correction to DFT that ensures eigenvalues match total energy differences
- to evaluate, requires the introduction of screening parameters α_i (replacing $E_{\mathrm{DFT}}|_{f_i=f}$ with $E_{\mathrm{DFT}}[\rho - \rho_i + f n_i]$)

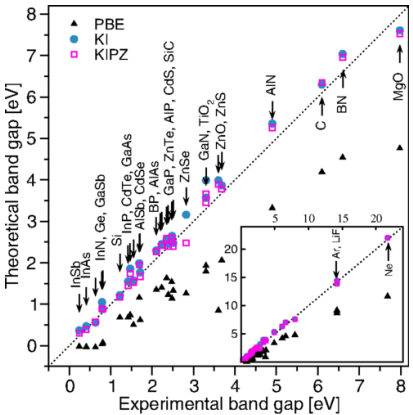
A quick recap of Koopmans functionals

$$E_{\mathrm{KI}}[\rho, \{\rho_i\}, \{\alpha_i\}] = E_{\mathrm{DFT}}[\rho] + \sum_i \alpha_i \Bigg(\underbrace{E_{\mathrm{Hxc}}[\rho - \rho_i] - E_{\mathrm{Hxc}}[\rho]}_{\text{removes erroneous curvature}} + \underbrace{f_i \left(E_{\mathrm{Hxc}}[\rho - \rho_i + n_i] - E_{\mathrm{Hxc}}[\rho - \rho_i] \right)}_{\text{restores linear behaviour}} \Bigg)$$

General features:

- a correction to DFT that ensures eigenvalues match total energy differences
- to evaluate, requires the introduction of screening parameters α_i (replacing $E_{\mathrm{DFT}}|_{f_i=f}$ with $E_{\mathrm{DFT}}[\rho - \rho_i + f n_i]$)
- is orbital-density-dependent $\rightarrow$ localised orbitals

A quick recap of Koopmans functionals



Mean absolute error (eV) across prototypical semiconductors and insulators

	PBE	G_0W_0	KI	KIPZ	QSG$\tilde{W}$
E_{gap}	2.54	0.56	0.27	0.22	0.18
IP	1.09	0.39	0.19	0.21	0.49

A quick recap of Koopmans functionals

	PBE	G_0W_0[1]	scG$\tilde{W}$[2]	KI@[PBE,MLWFs]	KIPZ@PBE	exp[3]
E_g	0.49	1.06	1.14	1.16	1.15	1.17
$\Gamma_{1v} \to \Gamma_{25'v}$	11.97	12.04		11.97	12.09	12.5 $\pm$0.6
$X_{1v} \to \Gamma_{25'v}$	7.82			7.82		7.75
$X_{4v} \to \Gamma_{25'v}$	2.85	2.99		2.85	2.86	2.90
$L_{2'v} \to \Gamma_{25'v}$	9.63	9.79		9.63	9.74	9.3 $\pm$0.4
$L_{1v} \to \Gamma_{25'v}$	6.98	7.18		6.98	7.04	6.8 $\pm$0.2
$L_{3'v} \to \Gamma_{25'v}$	1.19	1.27		1.19		1.2 $\pm$0.2
$\Gamma_{25'v} \to \Gamma_{15c}$	2.48	3.29		3.17	3.20	3.35$\pm$0.01
$\Gamma_{25'v} \to \Gamma_{2'c}$	3.28	4.02		3.95	3.95	4.15$\pm$0.05
$\Gamma_{25'v} \to X_{1c}$	0.62	1.38		1.28	1.31	1.13
$\Gamma_{25'v} \to L_{1c}$	1.45	2.21		2.12	2.13	2.04$\pm$0.06
$\Gamma_{25'v} \to L_{3c}$	3.24	4.18		3.91	3.94	3.9 $\pm$0.1
MSE	0.35	0.02		0.01	0.03	
MAE	0.44	0.21		0.14	0.17	

[1] M. Shishkin et al. *Phys. Rev. Lett.* 99.24 (2007), 246403 for E_g and M. S. Hybertsen et al. *Phys. Rev. B* 34.8 (1986), 5390 for the transitions;

[2] M. Shishkin et al. *Phys. Rev. B* 75.23 (2007), 235102.

[3] O. Madelung. *Semiconductors*. 3rd ed. Berlin: Springer-Verlag, 2004.

A quick recap of Koopmans functionals

$$E_{\mathrm{KI}}[\rho, \{\rho_i\}, \{\alpha_i\}] = E_{\mathrm{DFT}}[\rho] + \sum_i \alpha_i \Bigg(\underbrace{E_{\mathrm{Hxc}}[\rho - \rho_i] - E_{\mathrm{Hxc}}[\rho]}_{\text{removes erroneous curvature}} + \underbrace{f_i \left(E_{\mathrm{Hxc}}[\rho - \rho_i + n_i] - E_{\mathrm{Hxc}}[\rho - \rho_i] \right)}_{\text{restores linear behaviour}} \Bigg)$$

A quick recap of Koopmans functionals

$$E_{\mathrm{KI}}[\rho, \{\rho_i\}, \{\alpha_i\}] = E_{\mathrm{DFT}}[\rho] + \sum_i \alpha_i \Bigg(\underbrace{E_{\mathrm{Hxc}}[\rho - \rho_i] - E_{\mathrm{Hxc}}[\rho]}_{\text{removes erroneous curvature}} + \underbrace{f_i \left(E_{\mathrm{Hxc}}[\rho - \rho_i + n_i] - E_{\mathrm{Hxc}}[\rho - \rho_i] \right)}_{\text{restores linear behaviour}} \Bigg)$$

In order to evaluate this ODD functional, one must...

A quick recap of Koopmans functionals

$$E_{\mathrm{KI}}[\rho, \{\rho_i\}, \{\alpha_i\}] = E_{\mathrm{DFT}}[\rho] + \sum_i \alpha_i \Bigg(\underbrace{E_{\mathrm{Hxc}}[\rho - \rho_i] - E_{\mathrm{Hxc}}[\rho]}_{\text{removes erroneous curvature}} + \underbrace{f_i \left(E_{\mathrm{Hxc}}[\rho - \rho_i + n_i] - E_{\mathrm{Hxc}}[\rho - \rho_i] \right)}_{\text{restores linear behaviour}} \Bigg)$$

In order to evaluate this ODD functional, one must...

1. perform a Wannierisation

A quick recap of Koopmans functionals

$$E_{\mathrm{KI}}[\rho, \{\rho_i\}, \{\alpha_i\}] = E_{\mathrm{DFT}}[\rho] + \sum_i \alpha_i \Bigg(\underbrace{E_{\mathrm{Hxc}}[\rho - \rho_i] - E_{\mathrm{Hxc}}[\rho]}_{\text{removes erroneous curvature}} + \underbrace{f_i \left(E_{\mathrm{Hxc}}[\rho - \rho_i + n_i] - E_{\mathrm{Hxc}}[\rho - \rho_i] \right)}_{\text{restores linear behaviour}} \Bigg)$$

In order to evaluate this ODD functional, one must...

1. perform a Wannierisation
2. calculate the screening parameters $\{\alpha_i\}$

A quick recap of Koopmans functionals

$$E_{\mathrm{KI}}[\rho, \{\rho_i\}, \{\alpha_i\}] = E_{\mathrm{DFT}}[\rho] + \sum_i \alpha_i \Bigg(\underbrace{E_{\mathrm{Hxc}}[\rho - \rho_i] - E_{\mathrm{Hxc}}[\rho]}_{\text{removes erroneous curvature}} + \underbrace{f_i \left(E_{\mathrm{Hxc}}[\rho - \rho_i + n_i] - E_{\mathrm{Hxc}}[\rho - \rho_i] \right)}_{\text{restores linear behaviour}} \Bigg)$$

In order to evaluate this ODD functional, one must...

1. perform a Wannierisation
2. calculate the screening parameters $\{\alpha_i\}$
3. minimize the functional (sometimes)

A quick recap of Koopmans functionals

$$E_{\mathrm{KI}}[\rho, \{\rho_i\}, \{\alpha_i\}] = E_{\mathrm{DFT}}[\rho] + \sum_i \alpha_i \Bigg(\underbrace{E_{\mathrm{Hxc}}[\rho - \rho_i] - E_{\mathrm{Hxc}}[\rho]}_{\text{removes erroneous curvature}} + \underbrace{f_i \left(E_{\mathrm{Hxc}}[\rho - \rho_i + n_i] - E_{\mathrm{Hxc}}[\rho - \rho_i] \right)}_{\text{restores linear behaviour}} \Bigg)$$

In order to evaluate this ODD functional, one must...

1. perform a Wannierisation
2. calculate the screening parameters $\{\alpha_i\}$
3. minimize the functional (sometimes)
4. diagonalize the Hamiltonian

A quick recap of Koopmans functionals

$$E_{\mathrm{KI}}[\rho, \{\rho_i\}, \{\alpha_i\}] = E_{\mathrm{DFT}}[\rho] + \sum_i \alpha_i \Big(\underbrace{E_{\mathrm{Hxc}}[\rho - \rho_i] - E_{\mathrm{Hxc}}[\rho]}_{\text{removes erroneous curvature}} + \underbrace{f_i \left(E_{\mathrm{Hxc}}[\rho - \rho_i + n_i] - E_{\mathrm{Hxc}}[\rho - \rho_i] \right)}_{\text{restores linear behaviour}} \Big)$$

In order to evaluate this ODD functional, one must...

1. perform a Wannierisation
2. calculate the screening parameters $\{\alpha_i\}$
3. minimize the functional (sometimes)
4. diagonalize the Hamiltonian

All implemented in Koopmans

A quick recap of Koopmans functionals

What still stands in our way? Take the example of silicon:

A quick recap of Koopmans functionals

What still stands in our way? Take the example of silicon:

```
{
  "workflow": {
    "task": "singlepoint",
    "functional": "ki",
    "base_functional": "lda",
    "method": "dfpt",
    "pseudo_library": "pseudo_dojo_standard"},
  "atoms": {
    "cell_parameters": {"periodic": true, "ibrav": 2, "celldms": {"1": 10.2622}},
    "atomic_positions": {
       "units": "crystal",
       "positions": [["Si", 0.00, 0.00, 0.00], ["Si", 0.25, 0.25, 0.25]]}},
  "kpoints": {"grid": [8, 8, 8]},
  "calculator_parameters": {
    "ecutwfc": 60.0,
    "pw": {"nbnd": 20},
    "w90": {
      "projections": [[{"fsite": [0.25, 0.25, 0.25], "ang_mtm": "sp3"}],
                      [{"fsite": [0.25, 0.25, 0.25], "ang_mtm": "sp3"}]],
      "dis_froz_max": 10.6,
      "dis_win_max": 16.9}}}
```

Automated Wannierisation

Automated Wannierisation

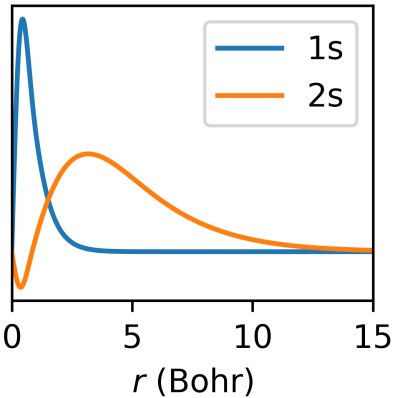
One step still very manual: Wannierisation
Can we automate it?

L. A. Agapito et al. *Phys. Rev. B* 88.16 (2013), 165127
J. Qiao et al. *npj Comput Mater* 9 (2023), 208; J. Qiao et al. *npj Comput Mater* 9 (2023), 206

Automated Wannierisation

One step still very manual: Wannierisation
Can we automate it?

- use PAOs found in pseudopotential files as initial guesses for Wannier functions

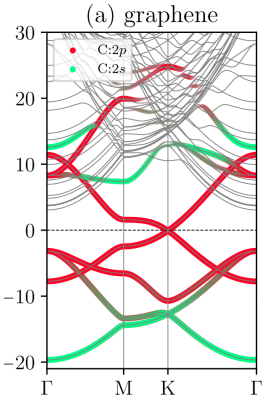


L. A. Agapito et al. *Phys. Rev. B* 88.16 (2013), 165127
J. Qiao et al. *npj Comput Mater* 9 (2023), 208; J. Qiao et al. *npj Comput Mater* 9 (2023), 206

Automated Wannierisation

One step still very manual: Wannierisation
Can we automate it?

- use PAOs found in pseudopotential files as initial guesses for Wannier functions
- projectability-based disentanglement instead of energy-based disentanglement

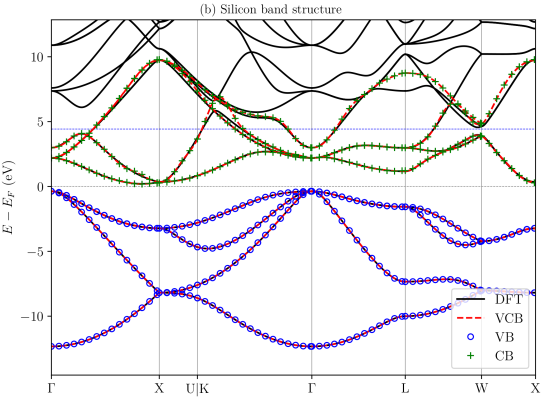


L. A. Agapito et al. *Phys. Rev. B* 88.16 (2013), 165127
J. Qiao et al. *npj Comput Mater* 9 (2023), 208; J. Qiao et al. *npj Comput Mater* 9 (2023), 206

Automated Wannierisation

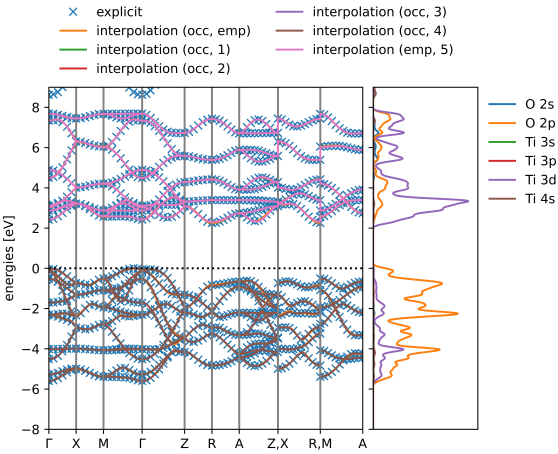
One step still very manual: Wannierisation
Can we automate it?

- use PAOs found in pseudopotential files as initial guesses for Wannier functions
- projectability-based disentanglement instead of energy-based disentanglement
- use a parallel transport algorithm to separate the occupied and empty manifolds

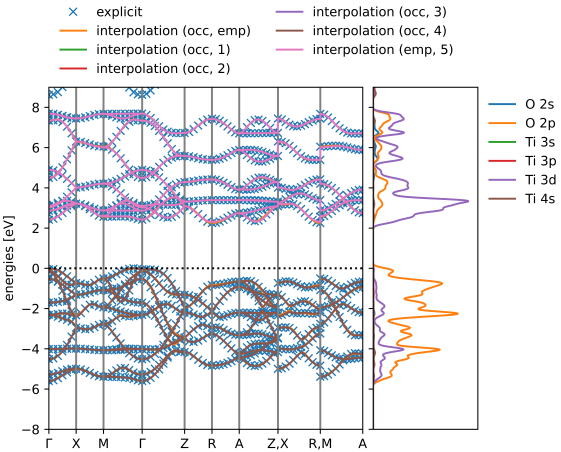


L. A. Agapito et al. *Phys. Rev. B* 88.16 (2013), 165127
J. Qiao et al. *npj Comput Mater* 9 (2023), 208; J. Qiao et al. *npj Comput Mater* 9 (2023), 206

Automated Wannierisation



Automated Wannierisation




```
{
  "workflow": {
    "task": "wannierize",
    "init_orbitals": "mlwfs",
    "pseudo_library": "pseudo_dojo_standard_v0.4.1"},
  "atoms": {
    "cell_parameters": {
      "ibrav": 6,
      "celldms": {"1": 8.675923, "3": 0.645248},
      "periodic": true},
    "atomic_positions": {
      "positions": [
        ["Ti", 0.5, 0.5, 0.5]
        ["Ti", 0.0, 0.0, 0.0],
        ["O", 0.1814, 0.8186, 0.5],
        ["O", 0.8186, 0.1814, 0.5],
        ["O", 0.3186, 0.3186, 0.0],
        ["O", 0.6814, 0.6814, 0.0]
      ],
      "units": "crystal"}},
  "kpoints": {"grid": [3, 3, 4]},
  "calculator_parameters": {
      "w90": {"auto_projections": true}}}
```

Automated Wannierisation

A drawback: the number of Wannier functions is determined by the number of PAOs in the pseudopotentials

Automated Wannierisation

A drawback: the number of Wannier functions is determined by the number of PAOs in the pseudopotentials... which can be a problem e.g. LiF

Automated Wannierisation

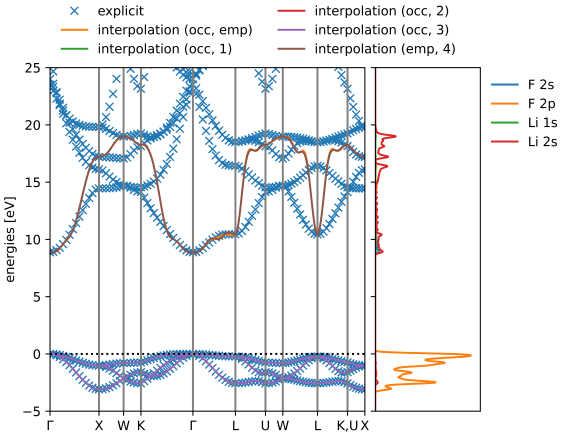
A drawback: the number of Wannier functions is determined by the number of PAOs in the pseudopotentials... which can be a problem e.g. LiF

element	configuration	PAOs
Li	$1s^2\ 2s^1$	1s, 2s
F	$[1s^2]\ 2s^2\ 2p^5$	2s, $2p_x$, $2p_y$, $2p_z$

Automated Wannierisation

A drawback: the number of Wannier functions is determined by the number of PAOs in the pseudopotentials... which can be a problem e.g. LiF

element	configuration	PAOs
Li	$1s^2\ 2s^1$	1s, 2s
F	$[1s^2]\ 2s^2\ 2p^5$	2s, $2p_x$, $2p_y$, $2p_z$

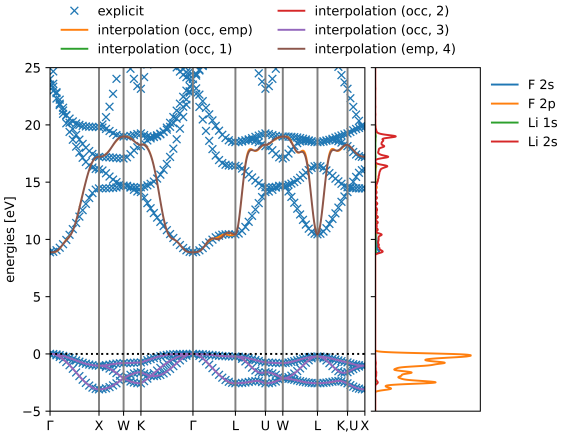


Automated Wannierisation

A drawback: the number of Wannier functions is determined by the number of PAOs in the pseudopotentials... which can be a problem e.g. LiF

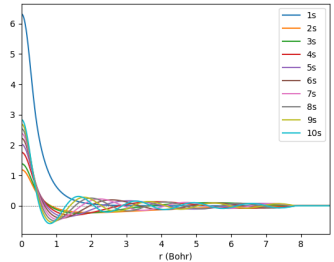
element	configuration	PAOs
Li	$1s^2\ 2s^1$	1s, 2s
F	$[1s^2]\ 2s^2\ 2p^5$	2s, $2p_x$, $2p_y$, $2p_z$

If we want a better representation of the conduction bands, we're gonna need ~~a bigger boat~~ more PAOs...



Automated Wannierization

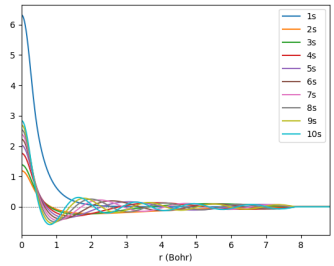
old



use the PAOs provided by OpenMX

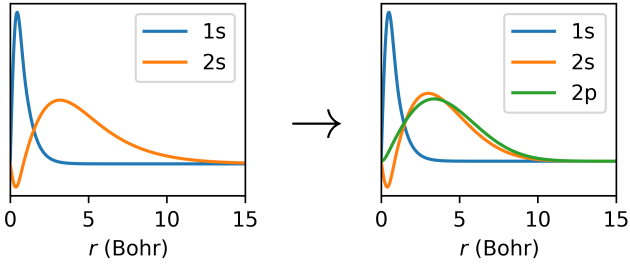
Automated Wannierization

old



use the PAOs provided by OpenMX

new



for the pseudopotential in question, resolve the radial Schrödinger equation for the next subshell

Pseudopotential generation

Solving the KS equations for a radial scalar-relativistic potential

Solving the KS equations for a radial scalar-relativistic potential

Solved on a logarithmic grid

$$r_i = r_0 e^{i\Delta x}$$

Solving the KS equations for a radial scalar-relativistic potential
Solved on a logarithmic grid

$$r_i = r_0 e^{i\Delta x}$$

Start with a trial potential $V[n](r)$ and eigenvalue ε. Then the radial wavefunction R_{nl} satisfies

$$\frac{1}{r^2}\frac{d^2R_{nl}(x)}{dx^2} = \frac{1}{r^2}\frac{dR_{nl}(x)}{dx} + \left(\frac{l(l+1)}{r^2} + M(r)(V(r)-\varepsilon)\right)R_{nl}(r) - \frac{\alpha^2}{4M(r)}\frac{dV(r)}{dr}\left(\frac{1}{r}\frac{dR_{nl}(x)}{dx} - \frac{R_{nl}(r)}{r}\right).$$

where $M(r) = 1 - \frac{\alpha^2}{4}(V(r)-\varepsilon)$

Solving scalar-relativisitic radial KS

Extrapolate outwards from the known small-r behaviour

$$R_{nl}(r) \sim r^{\gamma}; \qquad \gamma = \frac{l\sqrt{l^2 - \alpha^2 Z^2} + (l+1)\sqrt{(l+1)^2 - \alpha^2 Z^2}}{2l+1}$$

Solving scalar-relativisitic radial KS

Extrapolate outwards from the known small-r behaviour

$$R_{nl}(r) \sim r^{\gamma}; \qquad \gamma = \frac{l\sqrt{l^2 - \alpha^2 Z^2} + (l+1)\sqrt{(l+1)^2 - \alpha^2 Z^2}}{2l+1}$$

Extrapolate inwards from the known large-r behaviour

$$R_{nl}(r) \sim e^{-k(r)r}, \quad k(r) = \sqrt{\frac{l(l+1)}{r^2} + (V(r) - \epsilon)}$$

Solving scalar-relativisitic radial KS

Extrapolate outwards from the known small-r behaviour

$$R_{nl}(r) \sim r^{\gamma}; \qquad \gamma = \frac{l\sqrt{l^2 - \alpha^2 Z^2} + (l+1)\sqrt{(l+1)^2 - \alpha^2 Z^2}}{2l+1}$$

Extrapolate inwards from the known large-r behaviour

$$R_{nl}(r) \sim e^{-k(r)r}, \quad k(r) = \sqrt{\frac{l(l+1)}{r^2} + (V(r) - \epsilon)}$$

Solving scalar-relativisitic radial KS

Extrapolate outwards from the known small-r behaviour

$$R_{nl}(r) \sim r^{\gamma}; \qquad \gamma = \frac{l\sqrt{l^2 - \alpha^2 Z^2} + (l+1)\sqrt{(l+1)^2 - \alpha^2 Z^2}}{2l+1}$$

Extrapolate inwards from the known large-r behaviour

$$R_{nl}(r) \sim e^{-k(r)r}, \quad k(r) = \sqrt{\frac{l(l+1)}{r^2} + (V(r) - \epsilon)}$$

- if the wavefunction has too many/few nodes, ε is too high/low

Solving scalar-relativisitic radial KS

Extrapolate outwards from the known small-r behaviour

$$R_{nl}(r) \sim r^{\gamma}; \qquad \gamma = \frac{l\sqrt{l^2 - \alpha^2 Z^2} + (l+1)\sqrt{(l+1)^2 - \alpha^2 Z^2}}{2l+1}$$

Extrapolate inwards from the known large-r behaviour

$$R_{nl}(r) \sim e^{-k(r)r}, \quad k(r) = \sqrt{\frac{l(l+1)}{r^2} + (V(r) - \epsilon)}$$

- if the wavefunction has too many/few nodes, ε is too high/low
- if the wavefunction has a discontinuity where the outward and inward solutions intercept, we can estimate how much ε needs to change via perturbation theory

Solving scalar-relativisitic radial KS

Extrapolate outwards from the known small-r behaviour

$$R_{nl}(r) \sim r^{\gamma}; \qquad \gamma = \frac{l\sqrt{l^2 - \alpha^2 Z^2} + (l+1)\sqrt{(l+1)^2 - \alpha^2 Z^2}}{2l+1}$$

Extrapolate inwards from the known large-r behaviour

$$R_{nl}(r) \sim e^{-k(r)r}, \quad k(r) = \sqrt{\frac{l(l+1)}{r^2} + (V(r) - \epsilon)}$$

- if the wavefunction has too many/few nodes, ε is too high/low
- if the wavefunction has a discontinuity where the outward and inward solutions intercept, we can estimate how much ε needs to change via perturbation theory
- if the wavefunction does not have a discontinuity, we have found a solution! Now cycle for self-consistency over $V[n[(r)$

Pseudopotential generation

Use this procedure to generate extra PAOs for the Wannierisation

Pseudopotential generation

Use this procedure to generate extra PAOs for the Wannierisation

Problem: extra bound states may not exist

Pseudopotential generation

Use this procedure to generate extra PAOs for the Wannierisation

Problem: extra bound states may not exist

Solution: add a confining potential

Pseudopotential generation

Use this procedure to generate extra PAOs for the Wannierisation

Problem: extra bound states may not exist

Solution: add a confining potential

Problem: how to choose the confining potential parameters?

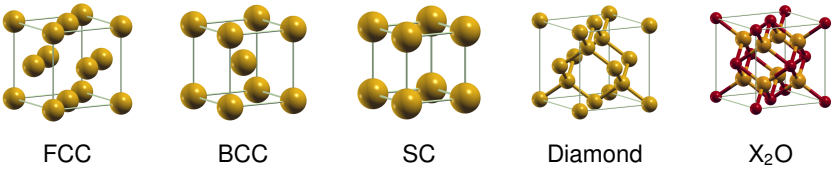
Pseudopotential generation

Use this procedure to generate extra PAOs for the Wannierisation

Problem: extra bound states may not exist

Solution: add a confining potential

Problem: how to choose the confining potential parameters?
Solution: optimise these criteria across typical structures:



Pseudopotential generation

The projectability is

$$p_{m\mathbf{k}} = \sum_n |\langle \varphi_n \mid \psi_{m\mathbf{k}} \rangle|^2$$

Pseudopotential generation

The projectability is

$$p_{m\mathbf{k}} = \sum_n |\langle \varphi_n \mid \psi_{m\mathbf{k}} \rangle|^2$$

We want a set of projectors such that...

- they should have a large overlap with some Kohn-Sham states and a small overlap with all the others
- they should lie within the span of the Kohn-Sham states

Pseudopotential generation

The projectability is

$$p_{m\mathbf{k}} = \sum_n |\langle \varphi_n \mid \psi_{m\mathbf{k}} \rangle|^2$$

We want a set of projectors such that...

- they should have a large overlap with some Kohn-Sham states and a small overlap with all the others
- they should lie within the span of the Kohn-Sham states

We can meet these criteria if we maximise the functional

$$F[\{\varphi_n\}] = \frac{1}{N_{\mathbf{k}} N_w} \sum_{\mathbf{k}} \sum_{m \in \mathcal{S}_{\mathbf{k}}} p_{m\mathbf{k}}$$

where $\mathcal{S}_{\mathbf{k}}$ corresponds to the N_w-largest values of $p_{m\mathbf{k}}$.

Pseudopotential generation

The projectability is

$$p_{m\mathbf{k}} = \sum_n |\langle \varphi_n \mid \psi_{m\mathbf{k}} \rangle|^2$$

We want a set of projectors such that...

- they should have a large overlap with some Kohn-Sham states and a small overlap with all the others
- they should lie within the span of the Kohn-Sham states

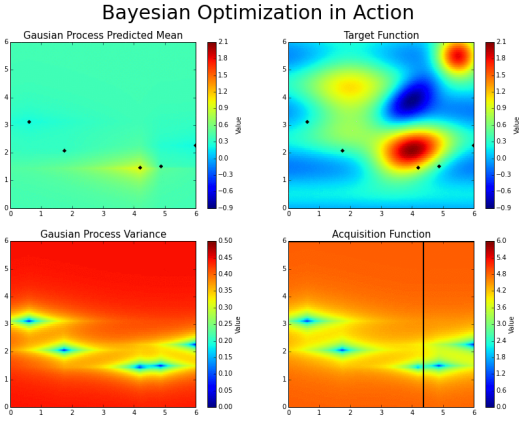
We can meet these criteria if we maximise the functional

$$F[\{\varphi_n\}] = \frac{1}{N_\mathbf{k} N_w} \sum_\mathbf{k} \sum_{m \in \mathcal{S}_\mathbf{k}} p_{m\mathbf{k}}$$

where $\mathcal{S}_\mathbf{k}$ corresponds to the N_w-largest values of $p_{m\mathbf{k}}$.

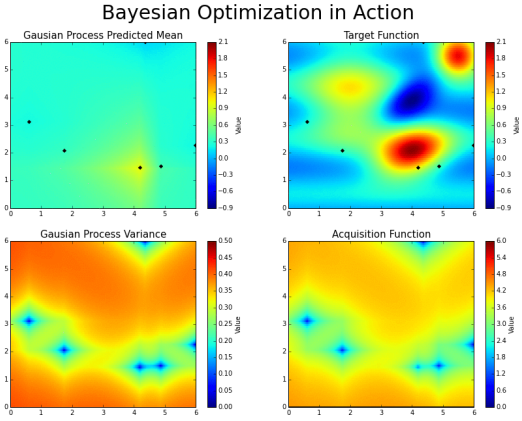
Maximise via *Bayesian optimisation*

Pseudopotential generation



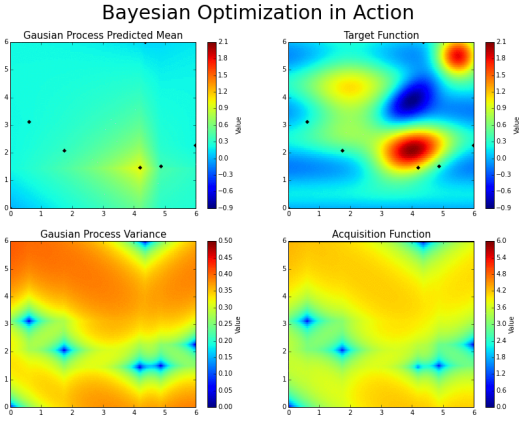
https://github.com/bayesian-optimization/BayesianOptimization

Pseudopotential generation

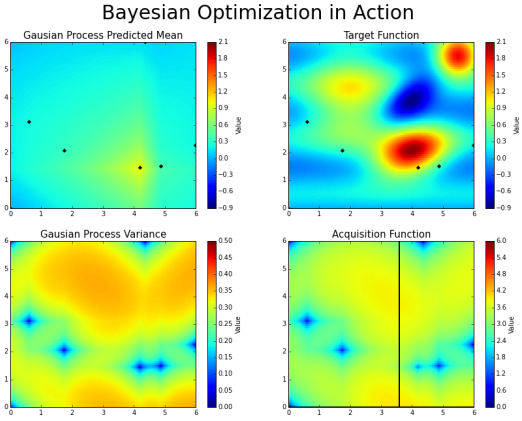


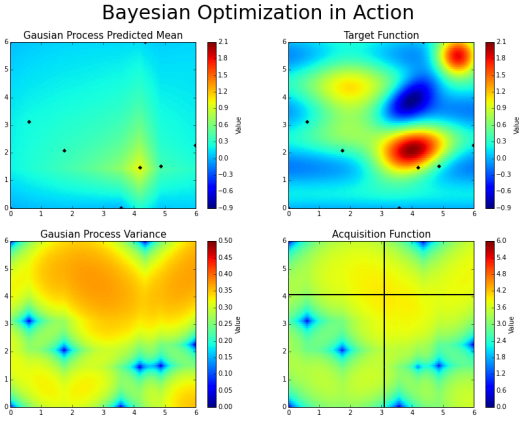
https://github.com/bayesian-optimization/BayesianOptimization

Pseudopotential generation

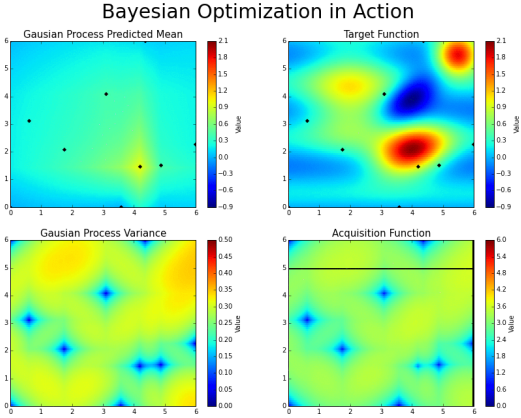


https://github.com/bayesian-optimization/BayesianOptimization


Bayesian Optimization in Action
Gausian Process Predicted Mean
Target Function
Gausian Process Variance
Acquisition Function
Value

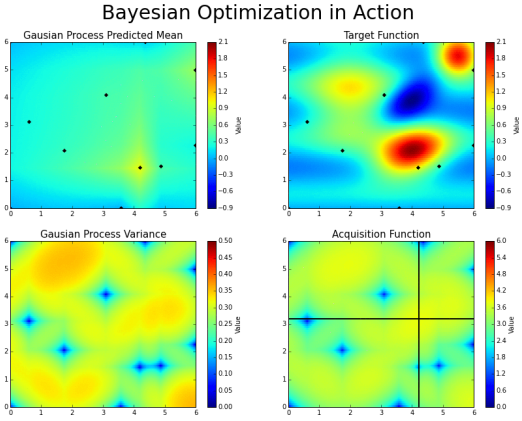

Bayesian Optimization in Action
Gausian Process Predicted Mean
Target Function
Gausian Process Variance
Acquisition Function
Value

Pseudopotential generation



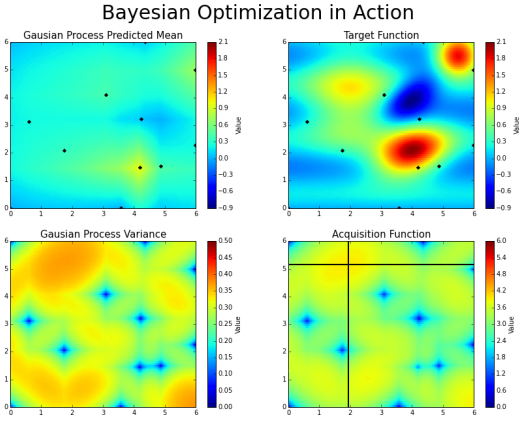
https://github.com/bayesian-optimization/BayesianOptimization

Pseudopotential generation

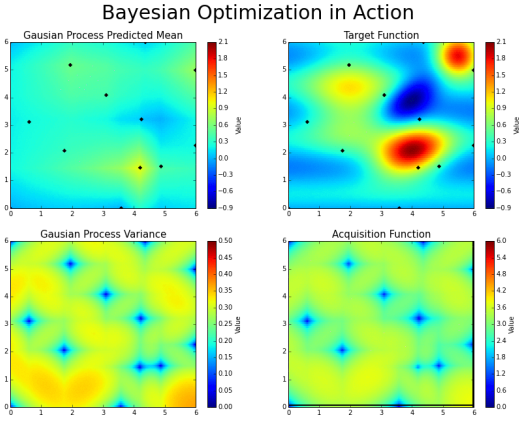


https://github.com/bayesian-optimization/BayesianOptimization

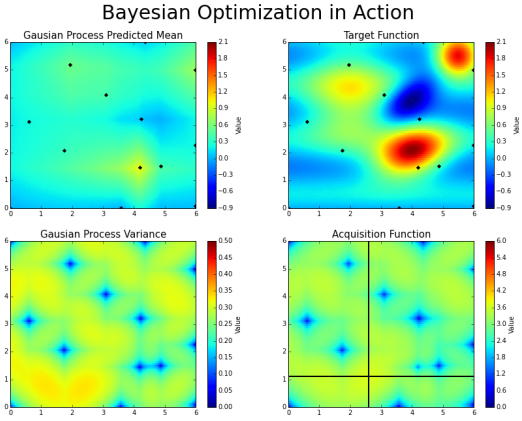
Pseudopotential generation

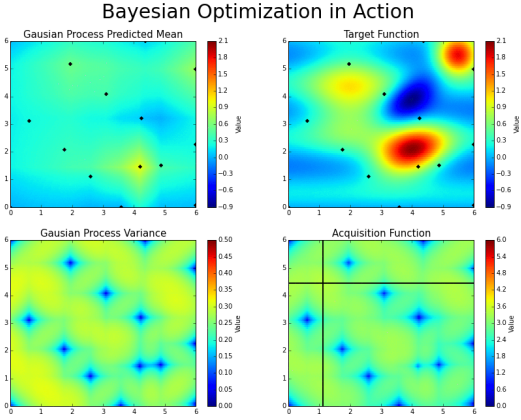


https://github.com/bayesian-optimization/BayesianOptimization



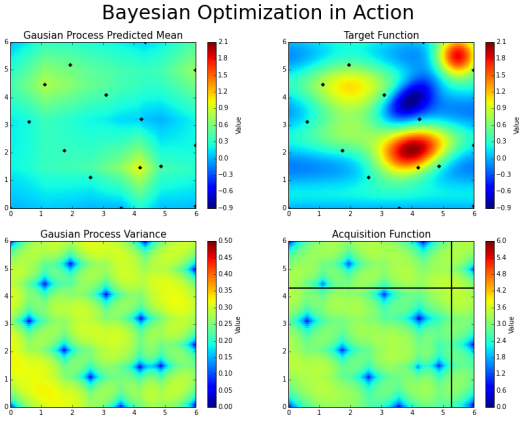
https://github.com/bayesian-optimization/BayesianOptimization


Bayesian Optimization in Action
Gausian Process Predicted Mean
Target Function
Gausian Process Variance
Acquisition Function
Value



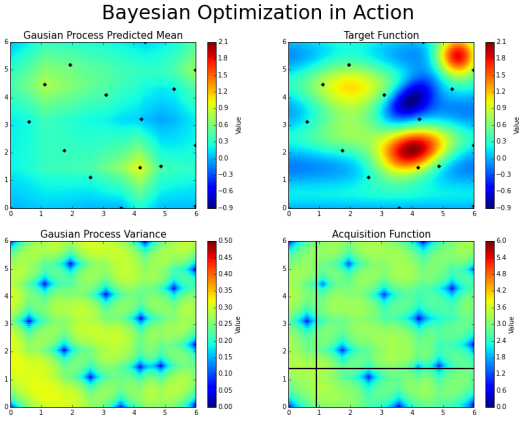
https://github.com/bayesian-optimization/BayesianOptimization

Pseudopotential generation



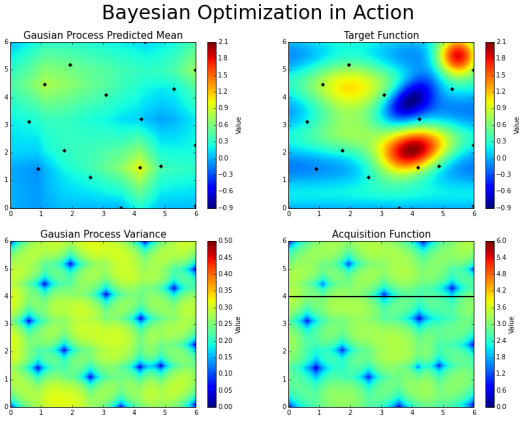
https://github.com/bayesian-optimization/BayesianOptimization

Pseudopotential generation



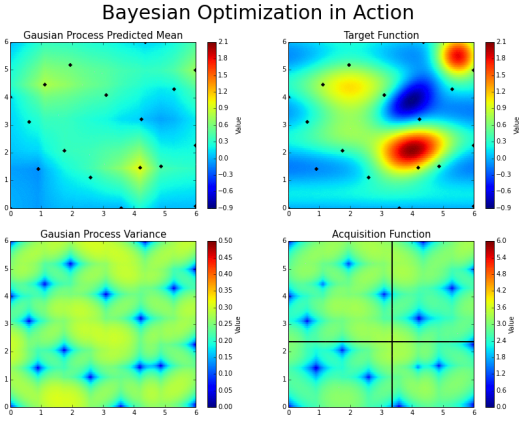
https://github.com/bayesian-optimization/BayesianOptimization

Pseudopotential generation



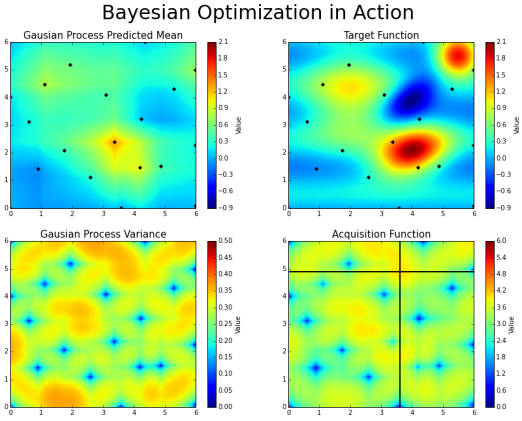
https://github.com/bayesian-optimization/BayesianOptimization

Pseudopotential generation



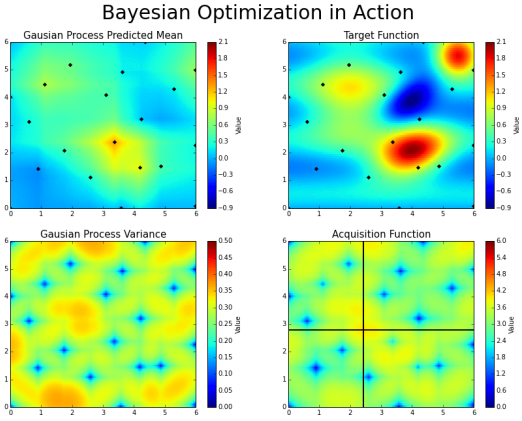
https://github.com/bayesian-optimization/BayesianOptimization

Pseudopotential generation



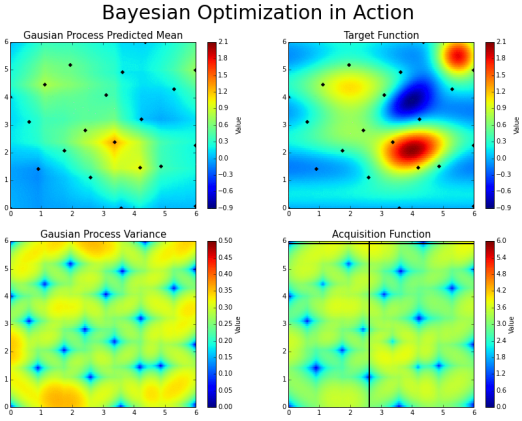
https://github.com/bayesian-optimization/BayesianOptimization

Pseudopotential generation



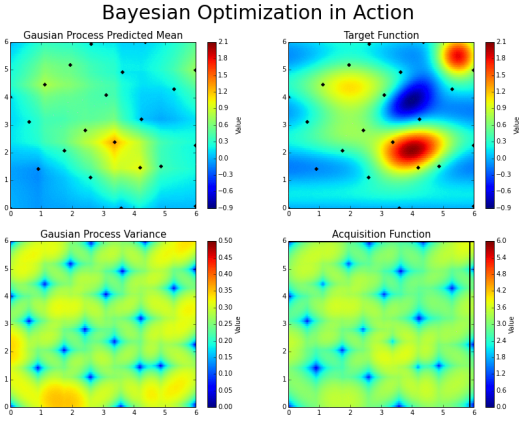
https://github.com/bayesian-optimization/BayesianOptimization

Pseudopotential generation



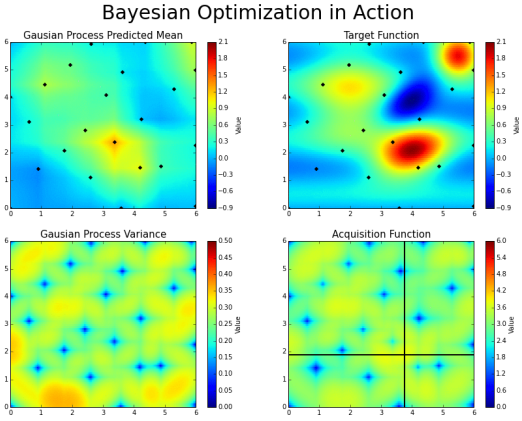
https://github.com/bayesian-optimization/BayesianOptimization

Pseudopotential generation

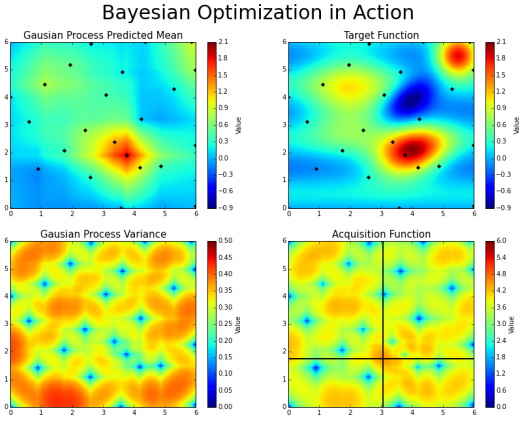


https://github.com/bayesian-optimization/BayesianOptimization

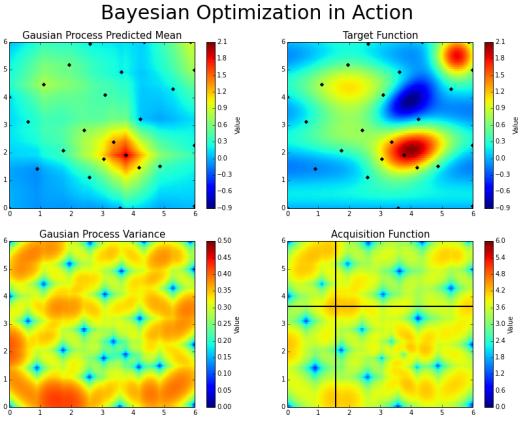
Pseudopotential generation



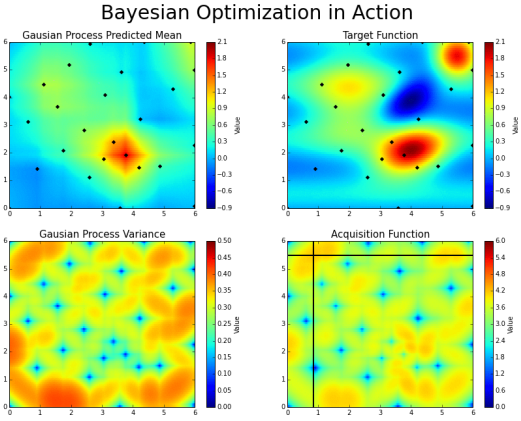
https://github.com/bayesian-optimization/BayesianOptimization



https://github.com/bayesian-optimization/BayesianOptimization

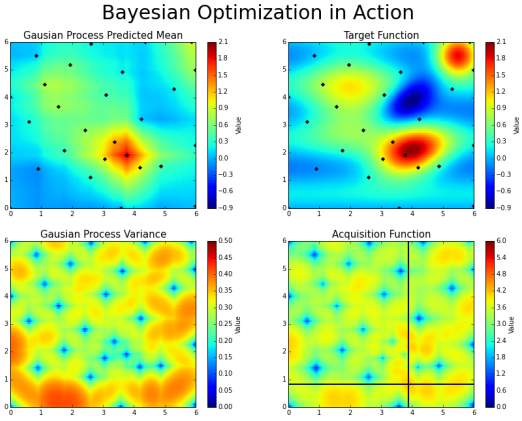

Bayesian Optimization in Action
Gausian Process Predicted Mean
Target Function
Gausian Process Variance
Acquisition Function
Value

Pseudopotential generation



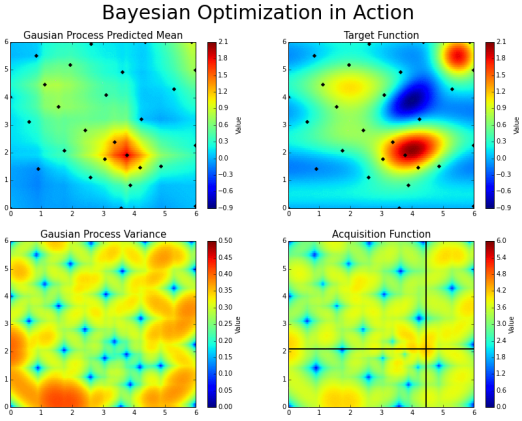
https://github.com/bayesian-optimization/BayesianOptimization

Pseudopotential generation

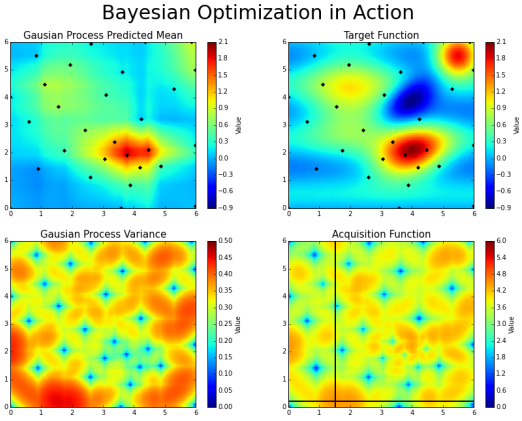


https://github.com/bayesian-optimization/BayesianOptimization

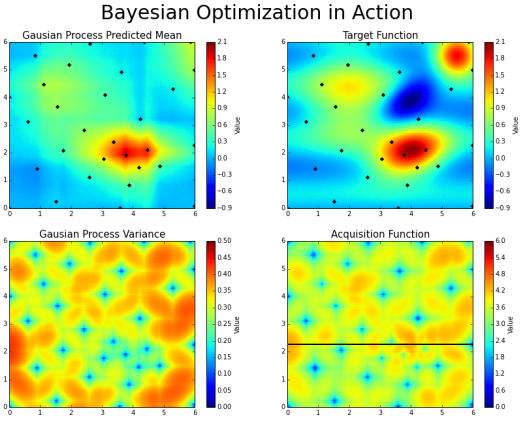
Pseudopotential generation



https://github.com/bayesian-optimization/BayesianOptimization

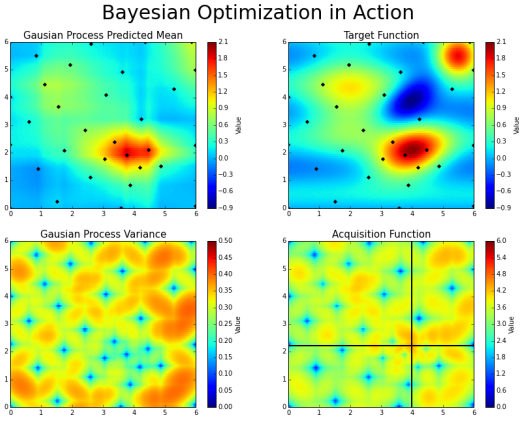

Bayesian Optimization in Action
Gausian Process Predicted Mean
Target Function
Gausian Process Variance
Acquisition Function
Value

Pseudopotential generation



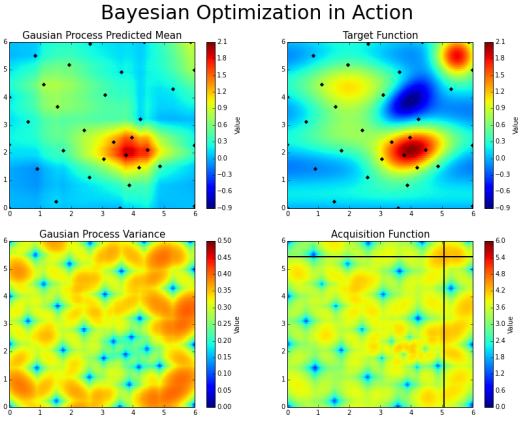
https://github.com/bayesian-optimization/BayesianOptimization

Pseudopotential generation



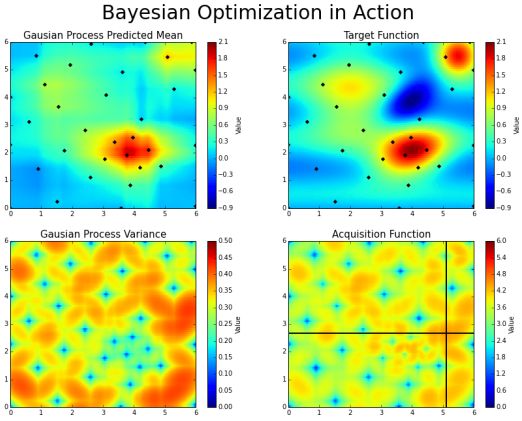
https://github.com/bayesian-optimization/BayesianOptimization

Pseudopotential generation



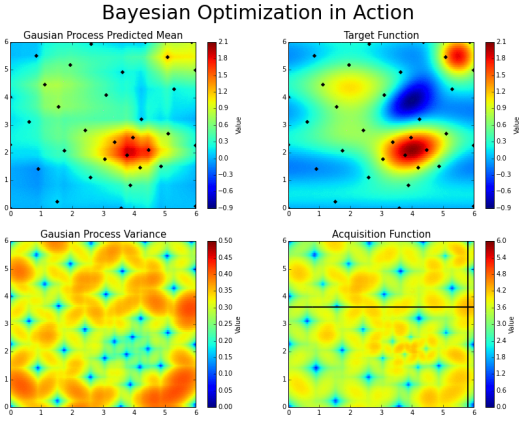
https://github.com/bayesian-optimization/BayesianOptimization

Pseudopotential generation

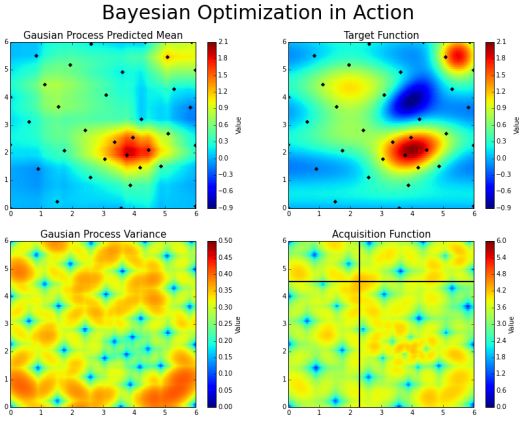


https://github.com/bayesian-optimization/BayesianOptimization

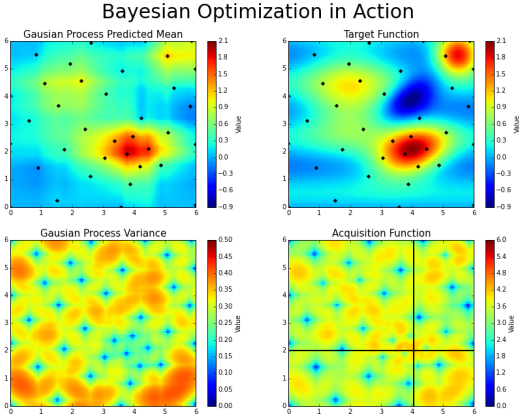
Pseudopotential generation



https://github.com/bayesian-optimization/BayesianOptimization

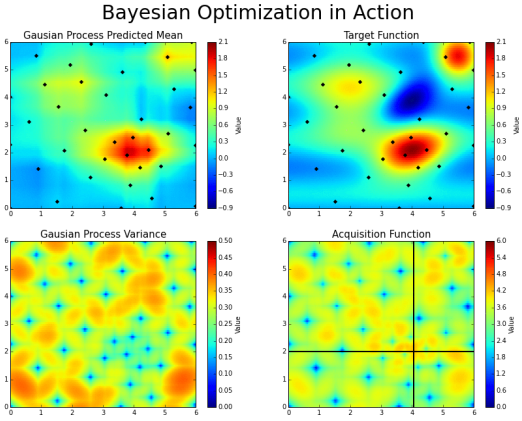

Bayesian Optimization in Action
Gausian Process Predicted Mean
Target Function
Gausian Process Variance
Acquisition Function
Value

Pseudopotential generation

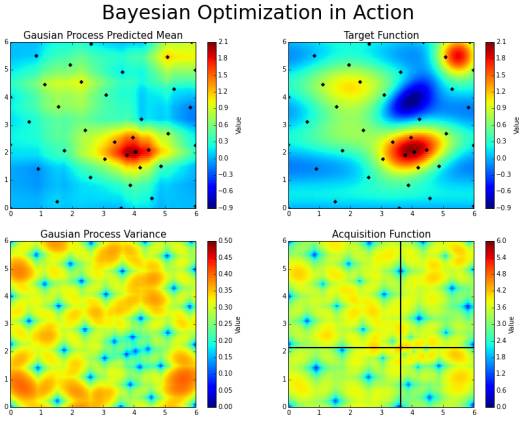


https://github.com/bayesian-optimization/BayesianOptimization

Pseudopotential generation

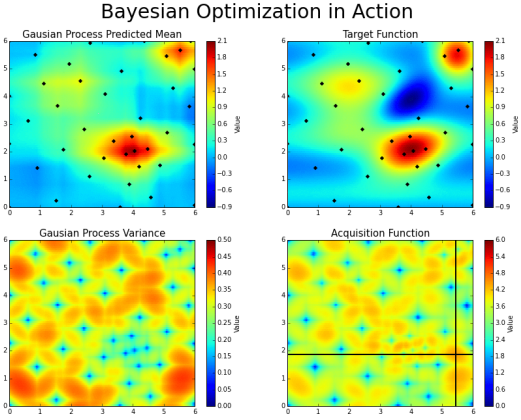


https://github.com/bayesian-optimization/BayesianOptimization



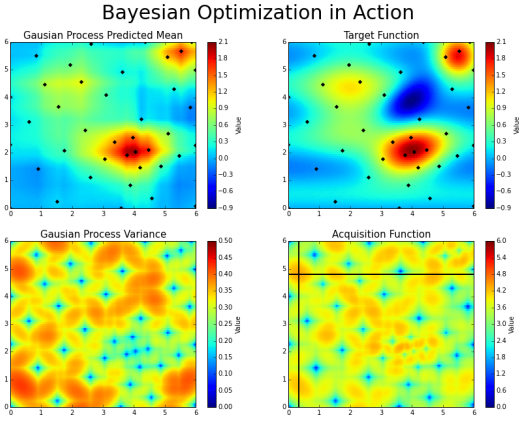
https://github.com/bayesian-optimization/BayesianOptimization

Pseudopotential generation



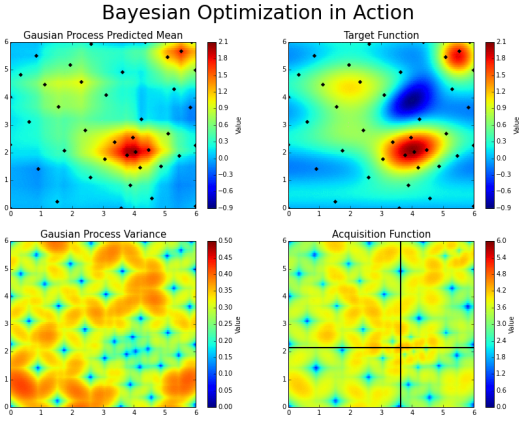
https://github.com/bayesian-optimization/BayesianOptimization

Pseudopotential generation



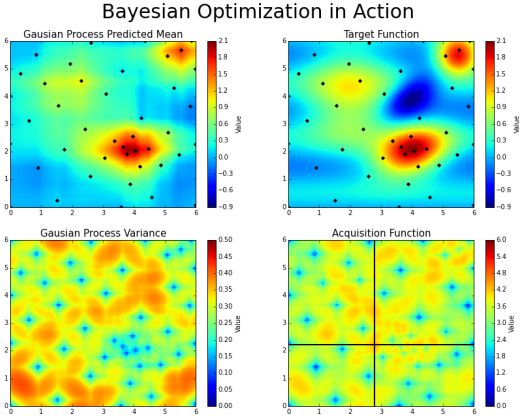
https://github.com/bayesian-optimization/BayesianOptimization

Pseudopotential generation



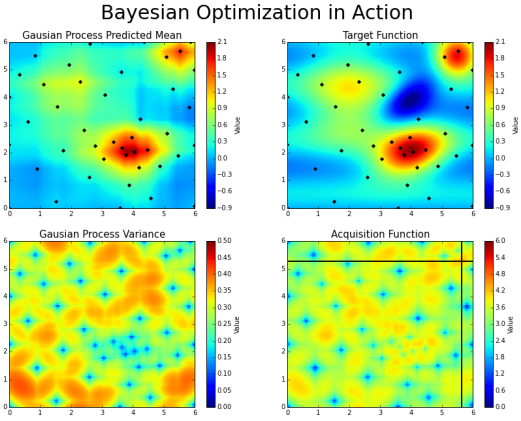
https://github.com/bayesian-optimization/BayesianOptimization

Pseudopotential generation

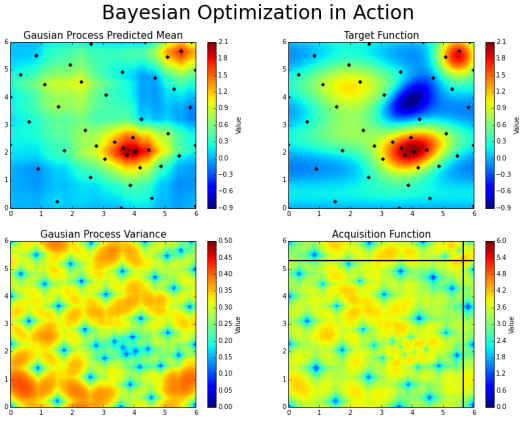


https://github.com/bayesian-optimization/BayesianOptimization

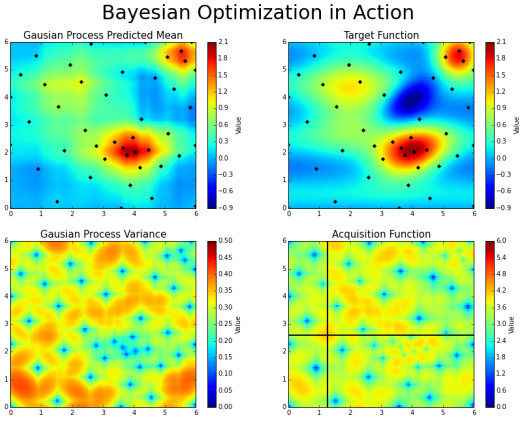
Pseudopotential generation



Pseudopotential generation

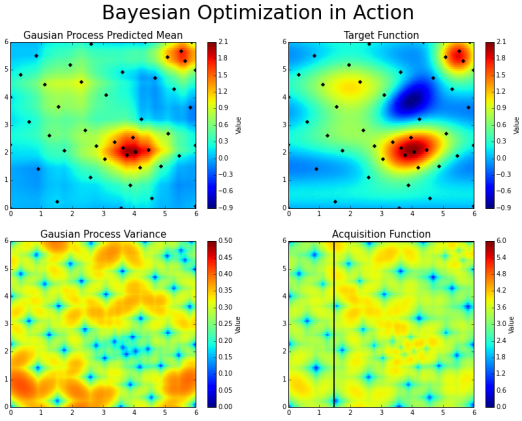


Pseudopotential generation

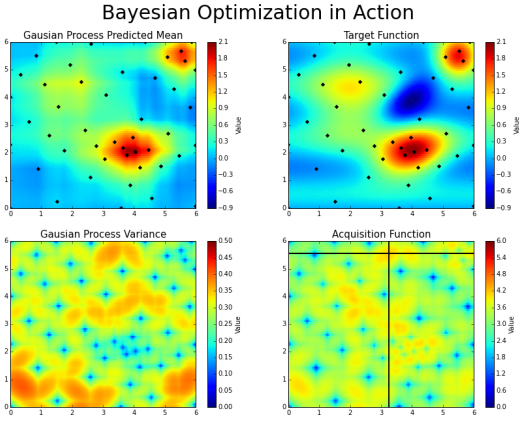


https://github.com/bayesian-optimization/BayesianOptimization

Pseudopotential generation

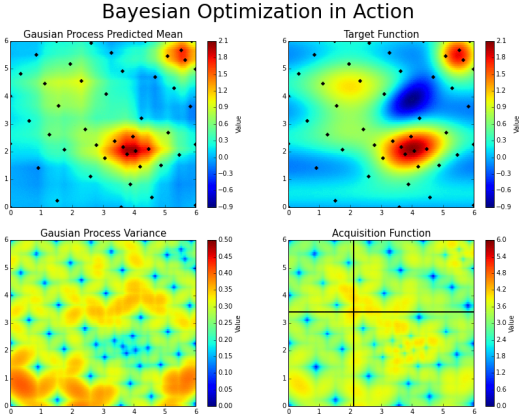


Pseudopotential generation

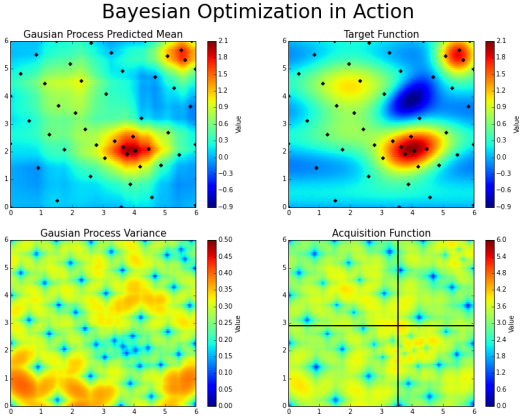


https://github.com/bayesian-optimization/BayesianOptimization

Pseudopotential generation

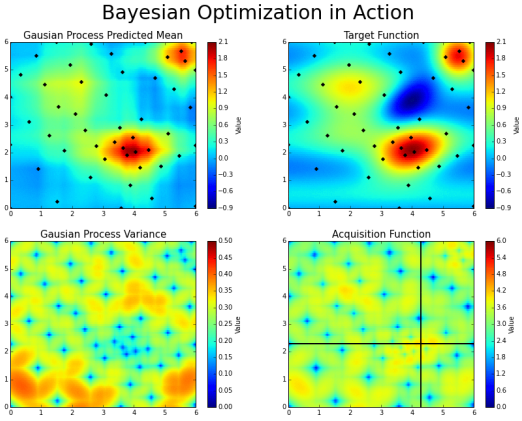


Pseudopotential generation



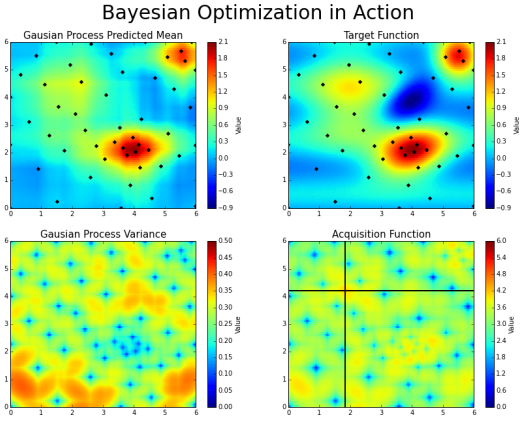
https://github.com/bayesian-optimization/BayesianOptimization

Pseudopotential generation



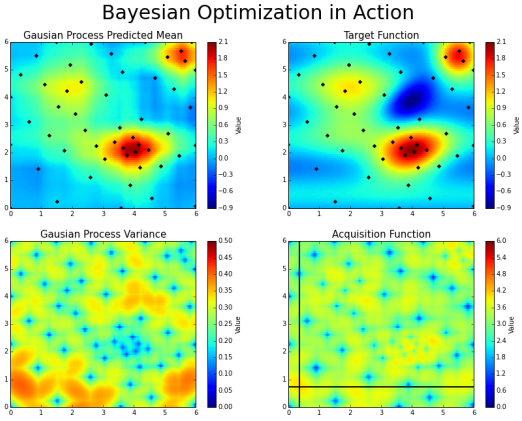
https://github.com/bayesian-optimization/BayesianOptimization

Pseudopotential generation

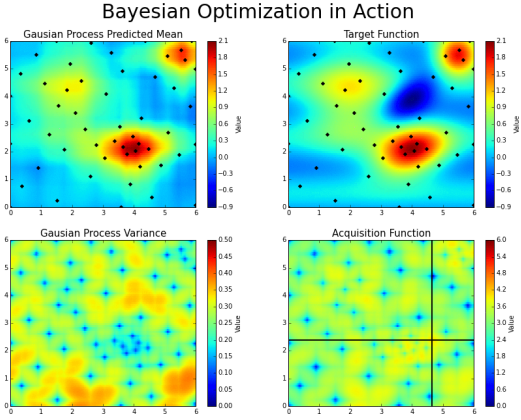


https://github.com/bayesian-optimization/BayesianOptimization

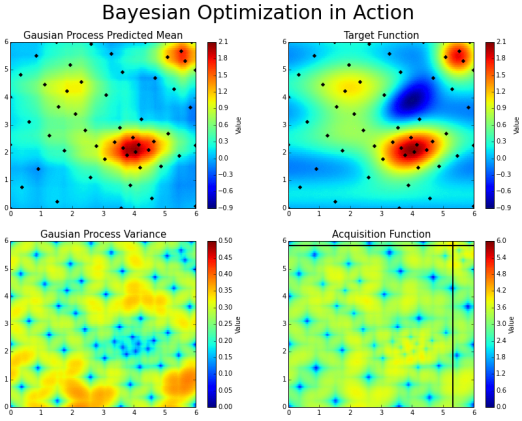
Pseudopotential generation



https://github.com/bayesian-optimization/BayesianOptimization

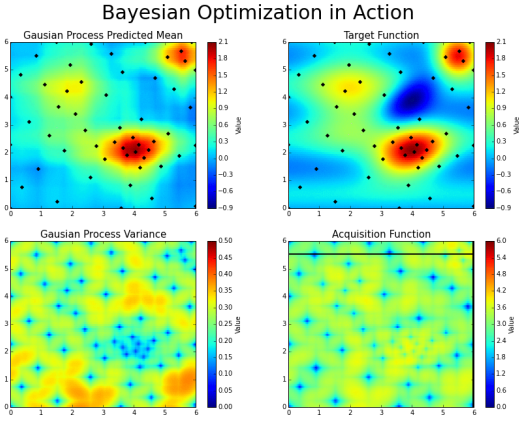

Bayesian Optimization in Action
Gausian Process Predicted Mean
Target Function
Gausian Process Variance
Acquisition Function
Value

Pseudopotential generation



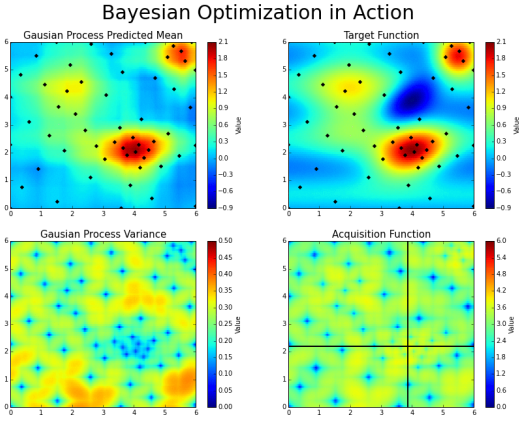
https://github.com/bayesian-optimization/BayesianOptimization

Pseudopotential generation



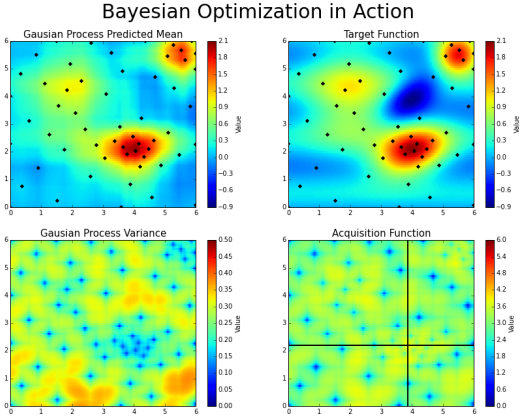
https://github.com/bayesian-optimization/BayesianOptimization

Pseudopotential generation



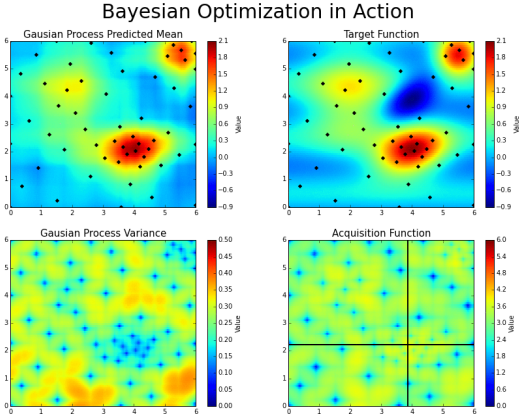
https://github.com/bayesian-optimization/BayesianOptimization

Pseudopotential generation



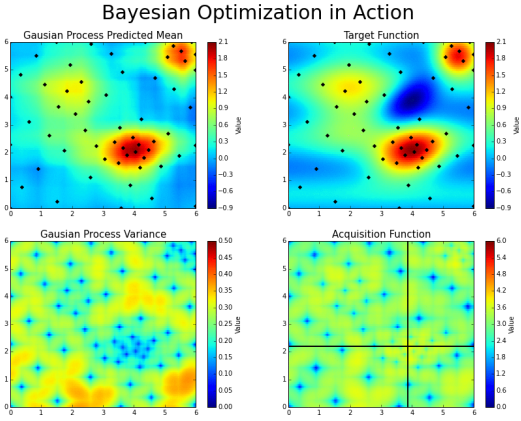
https://github.com/bayesian-optimization/BayesianOptimization

Pseudopotential generation



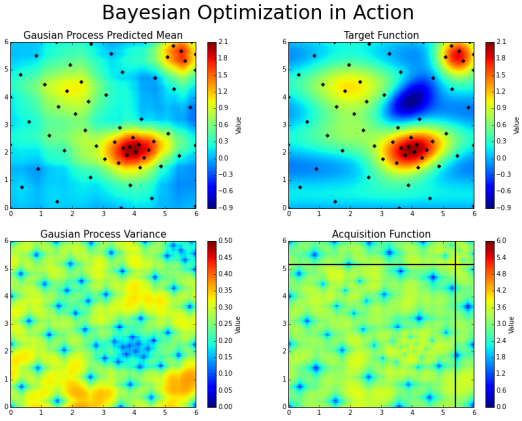
https://github.com/bayesian-optimization/BayesianOptimization

Pseudopotential generation



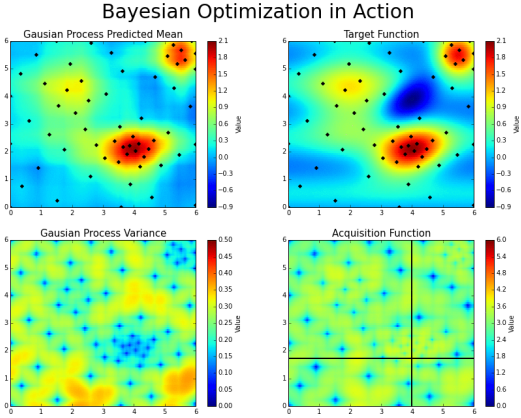
https://github.com/bayesian-optimization/BayesianOptimization

Pseudopotential generation



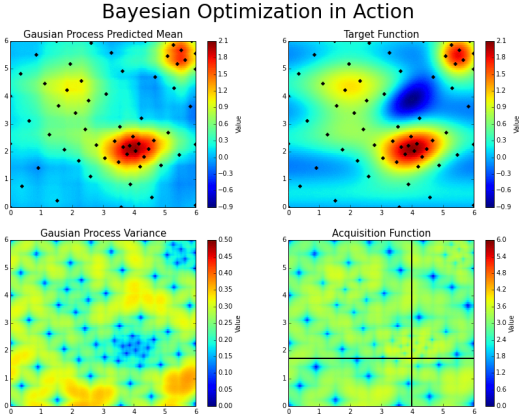
https://github.com/bayesian-optimization/BayesianOptimization

Pseudopotential generation



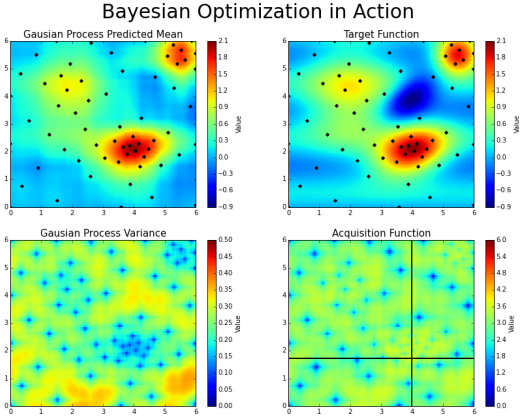
https://github.com/bayesian-optimization/BayesianOptimization

Pseudopotential generation



https://github.com/bayesian-optimization/BayesianOptimization

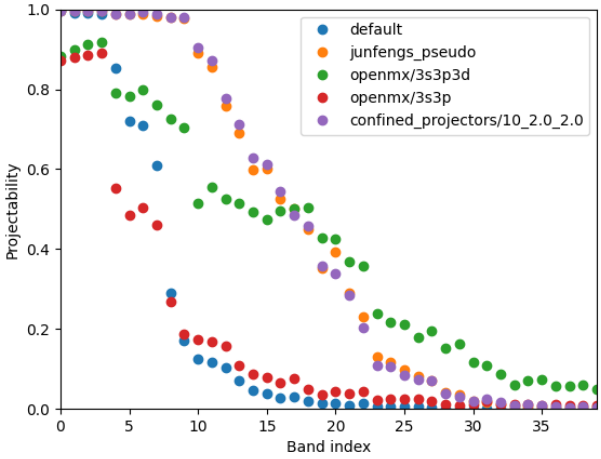
Pseudopotential generation



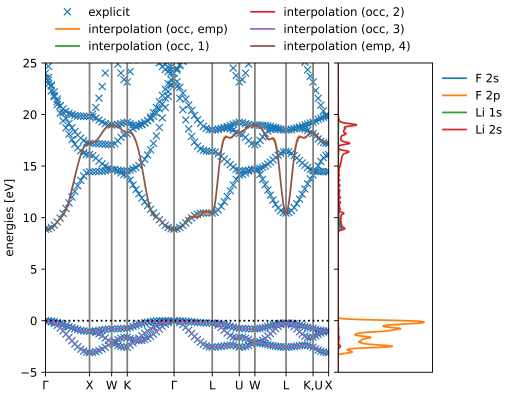
https://github.com/bayesian-optimization/BayesianOptimization

Pseudopotential generation

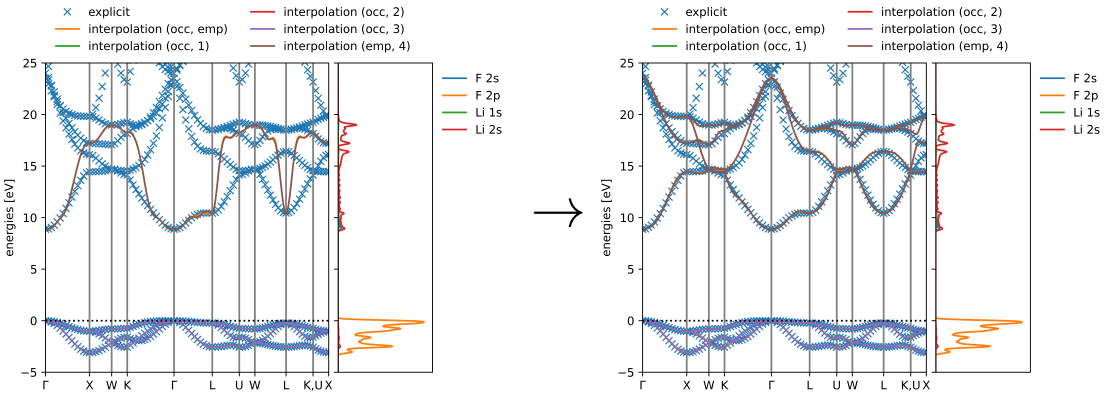
It works!



Pseudopotential generation

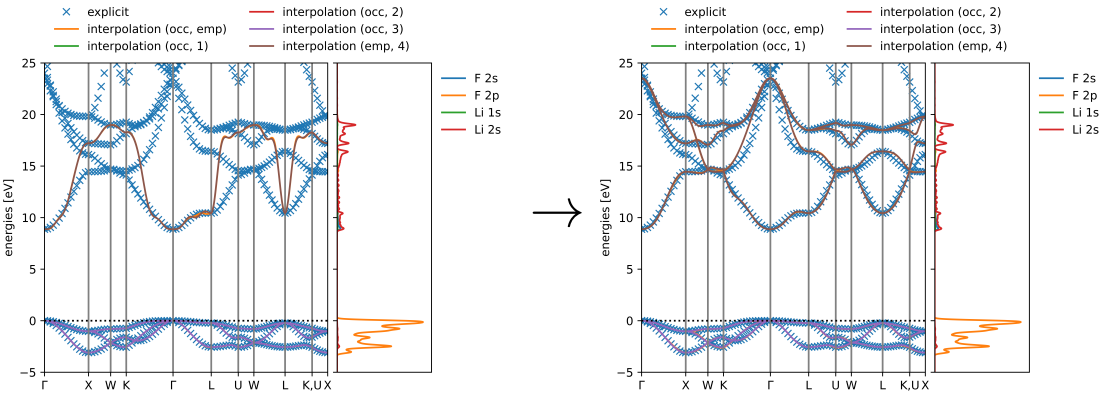


Pseudopotential generation



N. Colonna et al. *J. Chem. Theory Comput.* 18.9 (2022), 5435

Pseudopotential generation



	LDA	HSE	GW_0	scG$\tilde{W}$	KI	exp
E_{gap} (eV)	8.87	11.61	13.96	14.5	15.28	15.35

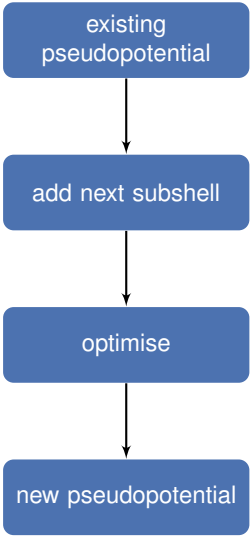
Systematic and accessible infrastructure?

Systematic and accessible infrastructure?

Currently implemented within a modified version of `oncvpsp.x`
In `oncvpsp.x...`

- the bracketing of ε is not robust (especially bad with the external potential)
- the code is not modular
- the code repository is not actively maintained

Systematic and accessible infrastructure?



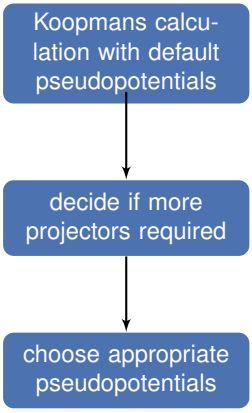
must be generated via `oncvpsp.x`

performed with a modified version of `oncvpsp.x`; would be nice to replace with a general solver

use Bayesian optimisation to select a confining potential that maximises projectability across the SSSP dataset

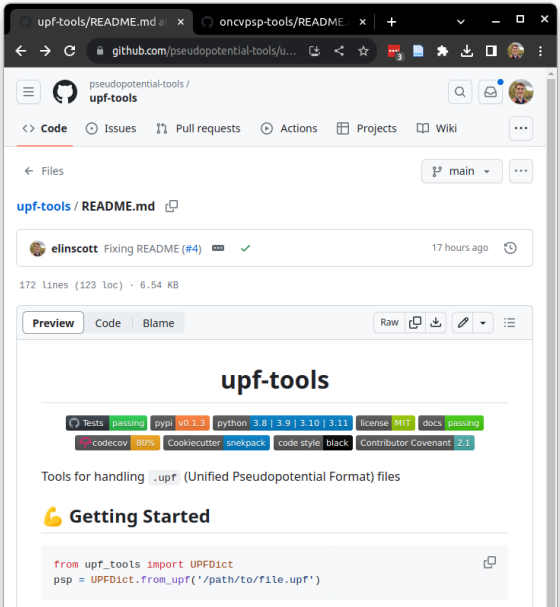
how to distribute?

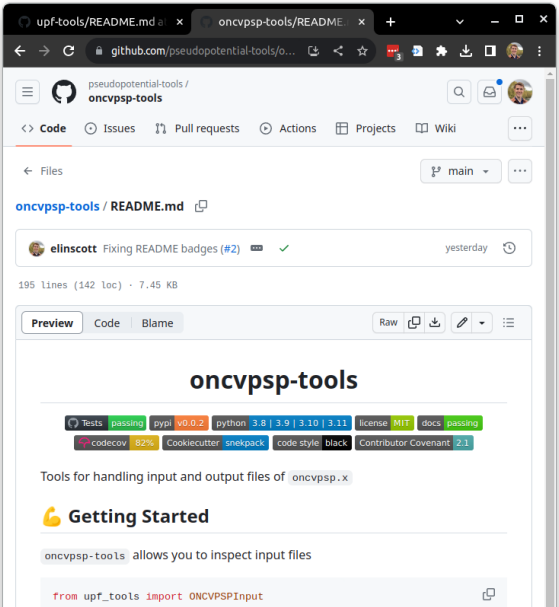
Systematic and accessible infrastructure?



band interpolation criteria? (But there are many reasons why bands don't match...)

we don't need/want extra subshells for all elements


upf-tools/README.md
github.com/pseudopotential-tools/u...
pseudopotential-tools /
upf-tools
Code
Issues
Pull requests
Actions
Projects
Wiki
Files
main
upf-tools / README.md
elinscott Fixing README (#4)
17 hours ago
172 lines (123 loc) · 6.54 KB
Preview
Code
Blame
Raw
upf-tools
Tests passing
pypi v0.1.3
python 3.8 | 3.9 | 3.10 | 3.11
license MIT
docs passing
codecov 80%
Cookiecutter snekpack
code style black
Contributor Covenant 2.1
Tools for handling .upf (Unified Pseudopotential Format) files
Getting Started
from upf_tools import UPFDict
psp = UPFDict.from_upf('/path/to/file.upf')


upf-tools/README.md
oncvpsp-tools/README
github.com/pseudopotential-tools/o...
pseudopotential-tools /
oncvpsp-tools
Code
Issues
Pull requests
Actions
Projects
Wiki
Files
main
oncvpsp-tools / README.md
elinscott Fixing README badges (#2)
yesterday
195 lines (142 loc) · 7.45 KB
Preview
Code
Blame
Raw
oncvpsp-tools
Tests passing
pypi v0.0.2
python 3.8 | 3.9 | 3.10 | 3.11
license MIT
docs passing
codecov 82%
Cookiecutter snekpack
code style black
Contributor Covenant 2.1
Tools for handling input and output files of oncvpsp.x
Getting Started
oncvpsp-tools allows you to inspect input files
from upf_tools import ONCVPSPInput

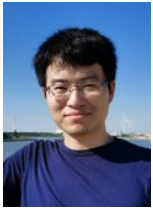
Publish your code!

- It's easy: most of the infrastructure was from the `cookiecutter` I used (see my July 2022 GM)
- It avoids duplication: `upf-tools` was discovered by Marnik and is now used in `AiiDA`

Acknowledgements


Nicola Marzari


Nicola Colonna


Junfeng Qiao

Swiss National Science Foundation

MARVEL
NATIONAL CENTRE OF COMPETENCE IN RESEARCH

slides available at github/elinscott-talks and on the THEOS wiki