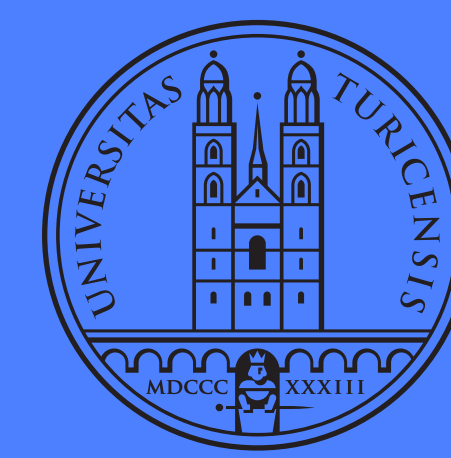




Wannier19: an advanced tool for interpolation of Berry curvature and related quantities.

Stepan S. Tsirkin

Department of Physics, University of Zürich, Switzerland



Universität
Zürich^{UZH}

ANOMALOUS HALL AND NERNST EFFECTS

The AHC tensor is given by:

$$\sigma_{\alpha\beta}^{\text{AHE}}(\mathbf{0}) = -\frac{e^2}{h} \int \frac{d\mathbf{k}}{(2\pi)^3} \sum_n f_{n\mathbf{k}} \Omega_{n,\alpha\beta}(\mathbf{k}) \quad (1)$$

in terms of Berry curvature:

$$\Omega_{n,\alpha\beta}(\mathbf{k}) = -2\text{Im} \langle \nabla_{\mathbf{k}\alpha} \mathbf{u}_{n\mathbf{k}} | \nabla_{\mathbf{k}\beta} \mathbf{u}_{n\mathbf{k}} \rangle. \quad (2)$$

where $|\mathbf{u}_{n\mathbf{k}}\rangle = e^{-i\mathbf{k}\cdot\mathbf{r}}|\psi_{n\mathbf{k}}\rangle$ is the cell-periodic Bloch function.

to evaluate it one needs:

- ▶ a smooth gauge
- ▶ dense grid of \mathbf{k} -points

That is where Wannier interpolation helps a lot.

Anomalous Nernst Effect is even harder

$$\alpha_{xy}^{\text{ANE}} = -\frac{1}{e} \int d\varepsilon \frac{\partial f}{\partial \varepsilon} \sigma_{xy}^{\text{AHE}}(\varepsilon) \frac{\varepsilon - \mu}{T} \quad (3)$$

because at low T it is essentially the derivative of σ^{AHE}

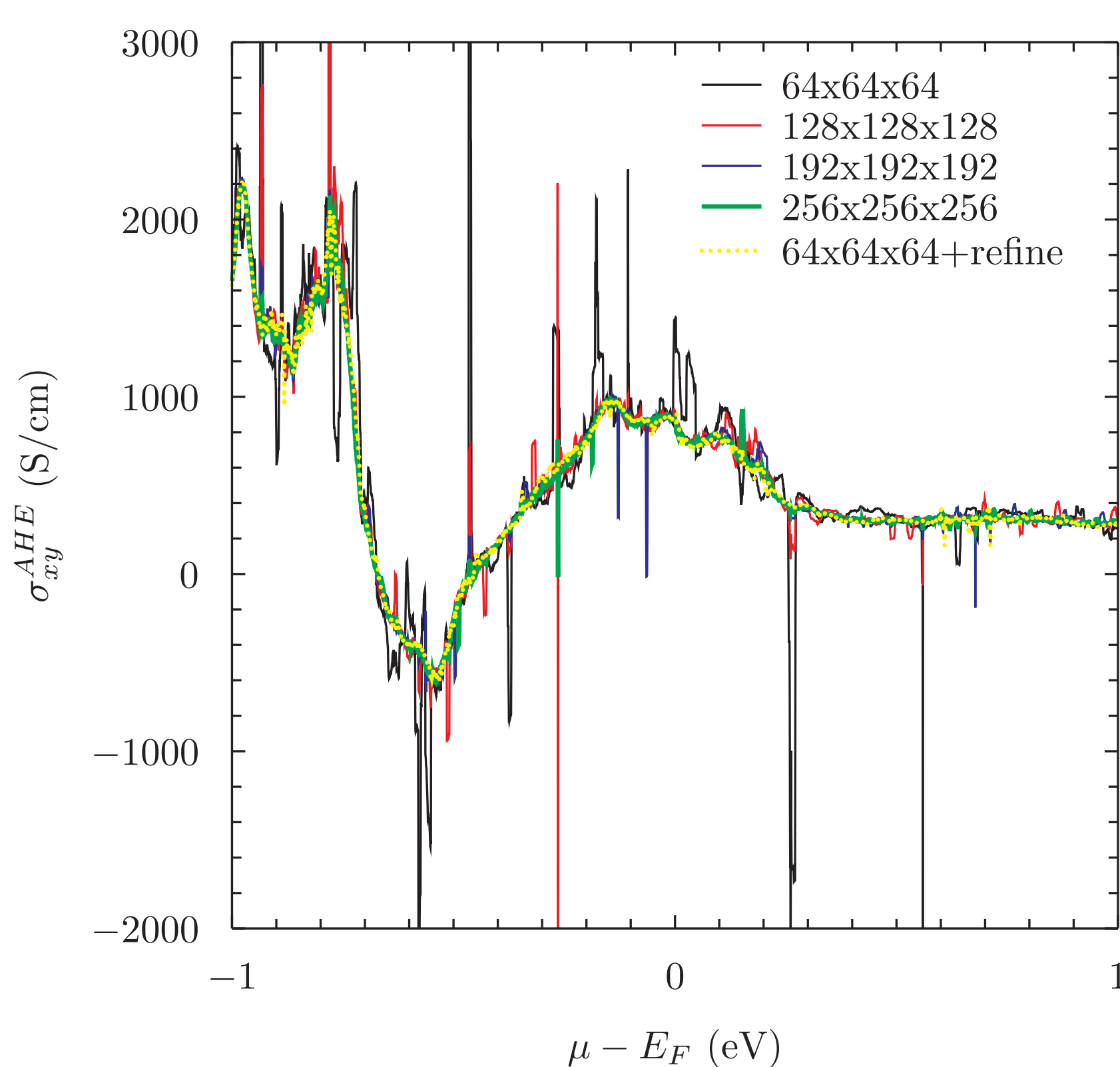


Figure 1: Example: AHC of bcc Fe as a function of chemical potential μ , for different interpolation grids, with and without recursive refinement

OVERVIEW

Wannier19 is an improved python implementation of postw90.x part of **Wannier90**. ("19" – because it was started in 2019.)

Wannier19 accepts the Wannier functions generated by Wannier90, or any tight-binding model. Then a BZ integral of any combination of velocities, Berry curvatures and other \mathbf{k} -space quantities can be evaluated.

Wannier90 is **good**, because it

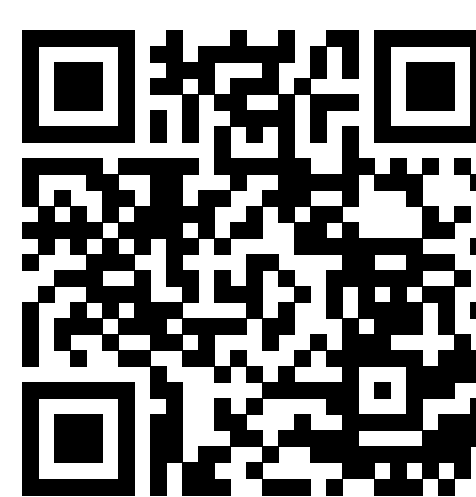
- ▶ is a well-established code
- ▶ known for ages
- ▶ has a broad community of developers
- ▶ has many implemented features
- ▶ (many more)

Wannier19 is **better**, because it

- ▶ is faster
- ▶ Employs a mixture of fast and slow Fourier transforms
- ▶ uses recursive adaptive refinement for accurate evaluation around special points in BZ
- ▶ accounts for the symmetries, to integrate only the irreducible part of BZ and make the result symmetric.
- ▶ is written in object-oriented Python, which allows easier further development.
- ▶ has some more efficient algorithms, in particular
 - ▶ for multiple Fermi levels
 - ▶ minimal-distance replica selection method

HOSTED ON GITHUB

<https://github.com/stepan-tsirkin/wannier19>



INSTALL WITH PIP

pip3 install wannier19

FIRST REAL-LIFE USE: AHE AND ANE IN MAGNETIC WEYL SEMIMETAL PRALGE

"Magnetism and anomalous transport in the Weyl semimetal PrAlGe: possible route to axial gauge fields". D. Destraz, L. Das, S.S. Tsirkin, Y. Xu, *et al.* Accepted in **npj Quantum Materials**.

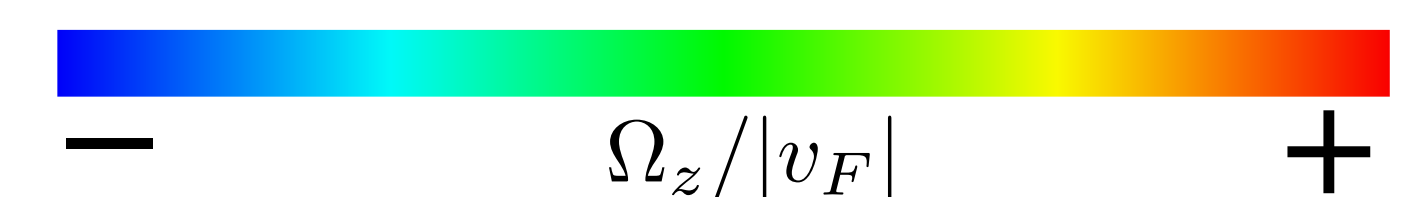
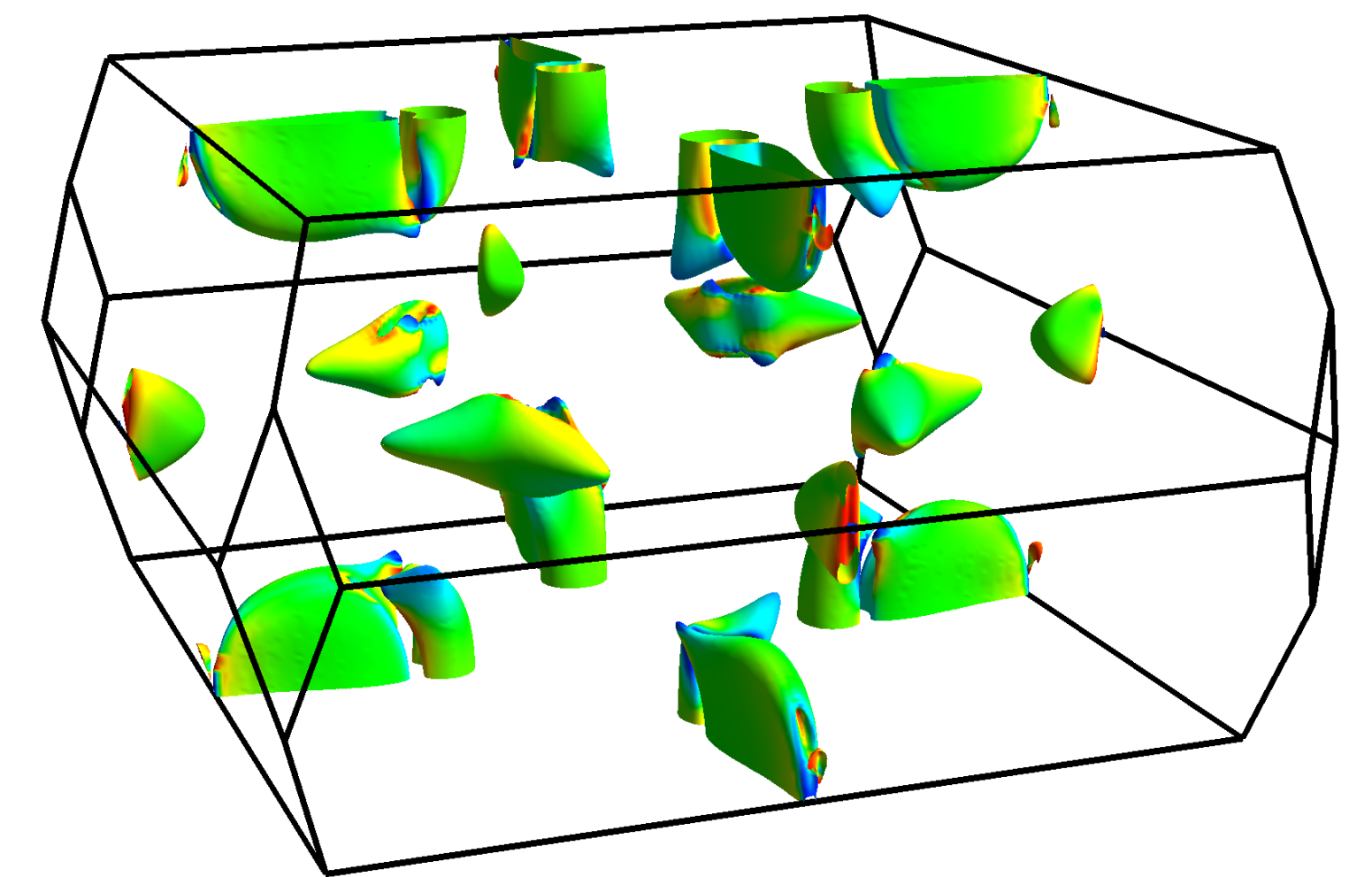


Figure 2: Fermi surface of ReAlGe coloured by Berry curvature divided by Fermi velocity. Plot produced by FermiSurfer (<http://fermisurfer.osdn.jp/>)

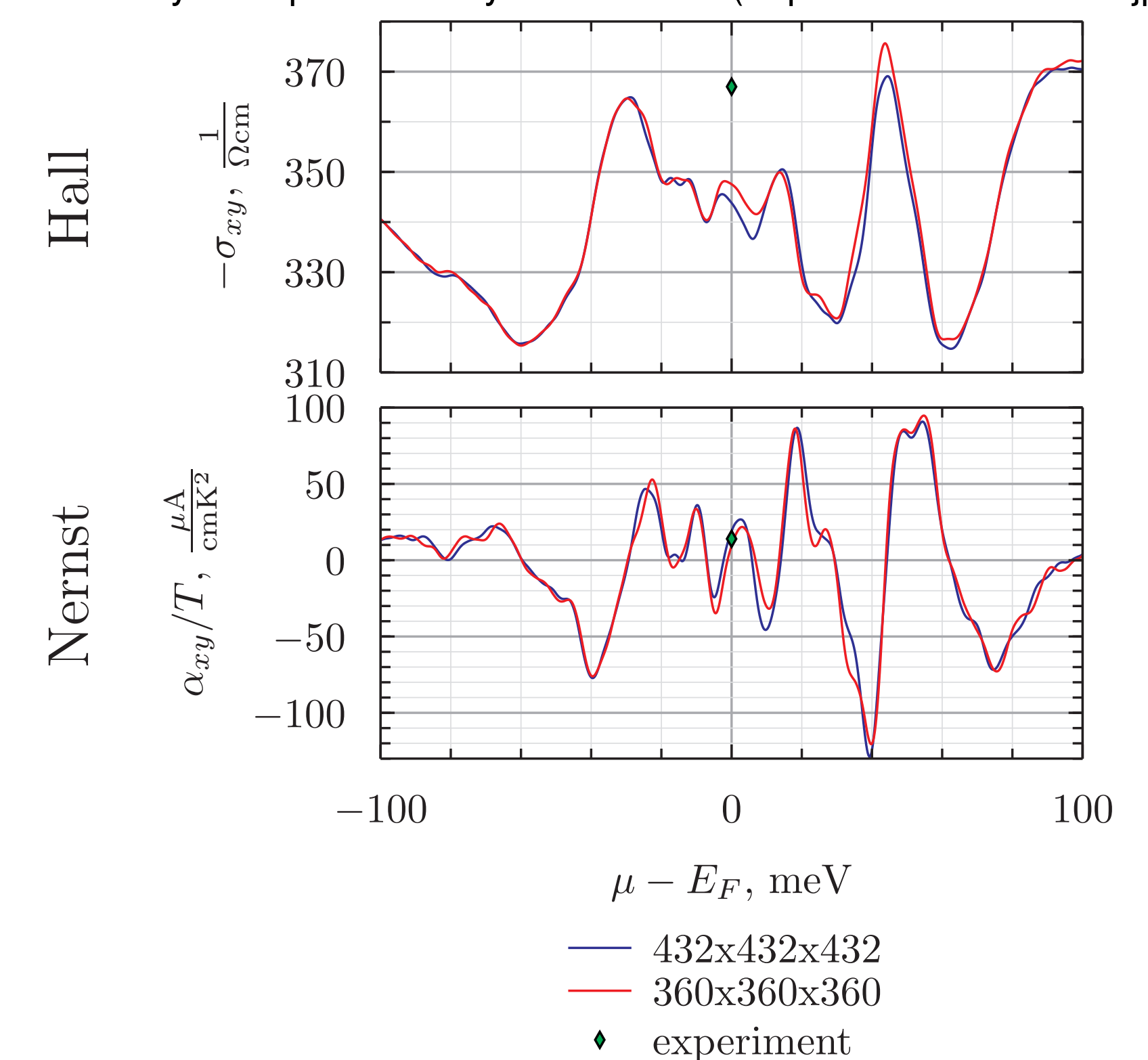
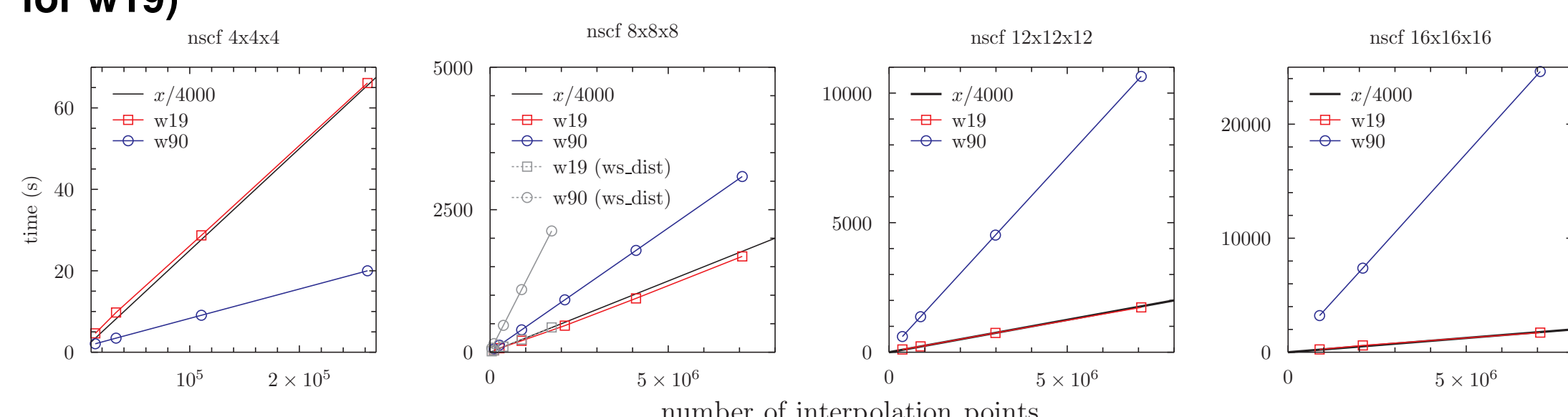


Figure 3: AHE and ANE calculated in PrAlGe in comparison with experiment

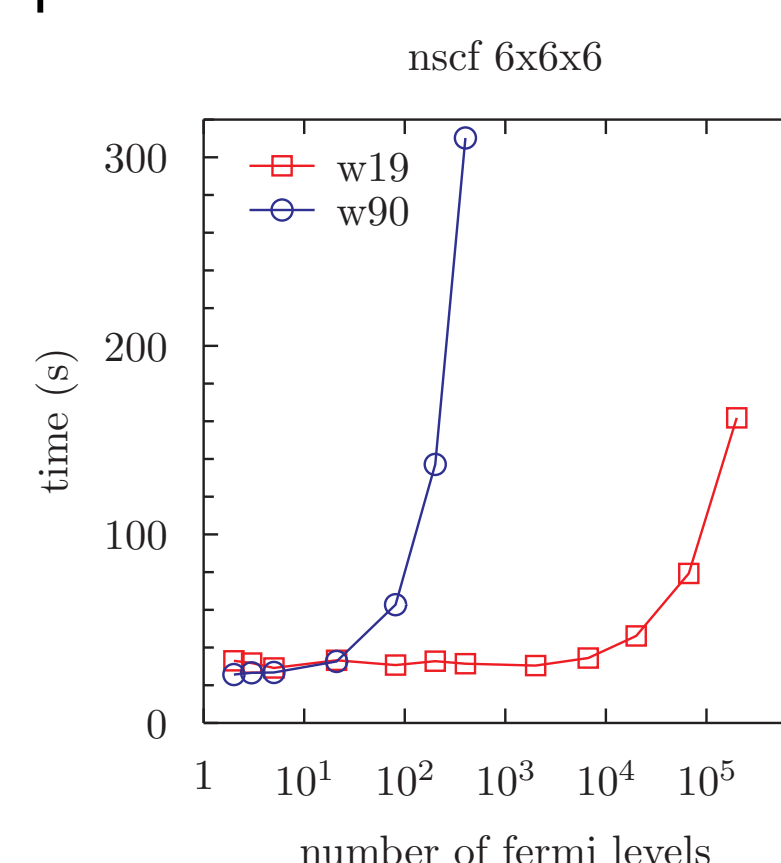
COMPUTATIONAL TIME

Calculations were performed with wannier19 and wannier90 on identical 40-core nodes of a cluster at UZH. Different grids of ab-initio (nscf) and interpolation grids were used. (**account of symmetries and adaptive refinement are switched off for w19**)



One can see that while for w19 the computational time does not depend on the ab-initio grid and the use of parameter "use ws distance". While for w90 they increase time a lot.

Also scanning multiple fermi levels has almost no cost in w19 up to 10^4 points:



MIXED FOURIER TRANSFORM

n — size of *ab initio* grid

$N = m \times n$ — size of interpolation grid (m -integer)

we want to Fourier Transform a quantity A :

$A(\mathbf{R}_1), \dots, A(\mathbf{R}_n) \rightarrow A(\mathbf{k}_0), \dots, A(\mathbf{k}_{N-1})$, where $\mathbf{k}_i = 2\pi\mathbf{i}/N$

For each $j = 0, \dots, m-1$ we "slow FT" $A(\mathbf{R}_i) \rightarrow \tilde{A}^j(\mathbf{R}_i) = A(\mathbf{R}_i) \times e^{2\pi i \mathbf{j} \cdot \mathbf{R}_i}$ and then do a FFT of $\tilde{A}^j(\mathbf{R}_i)$, thus getting \mathbf{k} -points $\mathbf{k}_{i,j} = 2\pi(\mathbf{j} + i\mathbf{m})/N$

time scaling:

slow FT $\propto N \times n$

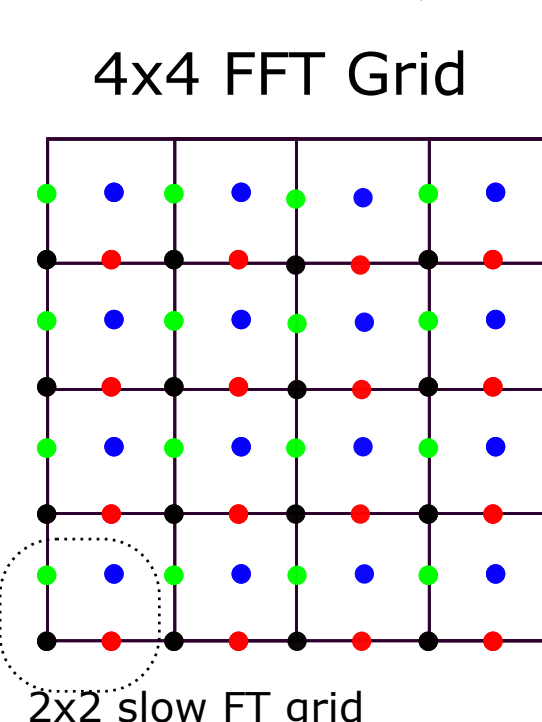
fast FT $\propto N \log N$

mixed FT $\propto N \log n$

Thus, FFT is better than SFT for large *ab initio* grids, while MFT is better in any case.

Advantages of mixed FT over FFT:

- ▶ easier parallelization
- ▶ allows to remove symmetry-equivalent "slow" \mathbf{k} -point
- ▶ no need to store the whole interpolation grid (less memory needed)
- ▶ allows adaptive recursive refinement of most "important" points.



RECURSIVE ADAPTIVE REFINEMENT

A few \mathbf{k} -points are chosen that give the largest contribution to the integral, or to the jittering of the curve. — typically the points near (avoided) degeneracies. They are replaced by grids of $2 \times 2 \times 2$ points surrounding it. The procedure is repeated recursively (new points can be refined again) to get a smoother curve.

TUTORIAL.PY

```
1 #!/usr/bin/env python3
2 import wannier19 as w19
3 import numpy as np
4
5 name="Fe"
6 NK=(96,96,96) # a grid 96x96x96
7 Efermi=np.linspace(12.,13.,1001) # range of Fermi levels
8 system=w19.System(tb_file='Fe_tb.dat',getAA=True)
9
10 # symmetries of the system to be accounted for :
11 SYM=w19.symmetry
12 generators=[SYM.Inversion,SYM.C4z,SYM.TimeReversal*SYM.C2x]
13
14 # tabulate quantities to be plotted by FermiSurfer
15 w19.tabulate(system,
16             NK=NK,
17             quantities=["V","berry"],
18             symmetry_gen=generators,
19             fout_name=name,
20             numproc=4,
21             Ef0=0,
22             restart=False )
23
24 # integrated quantities
25 w19.integrate(system,
26             NK=NK,
27             Efermi=Efermi,
28             smearEf=10,
29             quantities=["ahc","dos"],
30             numproc=4,
31             adpt_num_iter=10,
32             fout_name=name,
33             symmetry_gen=generators,
34             restart=False )
35
```

ACKNOWLEDGEMENTS

I thank Ivo Souza and Miguel Angel Jiménez Herrera for fruitful discussion