npj Computational Materials

www.nature.com/npjcompumats

ARTICLE OPEN

Check for updates

High performance Wannier interpolation of Berry curvature and related quantities with WannierBerri code

Stepan S. Tsirkin[1]✉

Wannier interpolation is a powerful tool for performing Brillouin zone integrals over dense grids of **k** points, which are essential to evaluate such quantities as the intrinsic anomalous Hall conductivity or Boltzmann transport coefficients. However, more complex physical problems and materials create harder numerical challenges, and computations with the existing codes become very expensive, which often prevents reaching the desired accuracy. In this article, I present a series of methods that boost the speed of Wannier interpolation by several orders of magnitude. They include a combination of fast and slow Fourier transforms, explicit use of symmetries, and recursive adaptive grid refinement among others. The proposed methodology has been implemented in the python code WannierBerri, which also aims to serve as a convenient platform for the future development of interpolation schemes for other phenomena.

npj Computational Materials (2021)7:33 ; https://doi.org/10.1038/s41524-021-00498-5

INTRODUCTION

Wannier functions[1,2] (WFs) are a powerful tool for evaluating various electronic properties of solids, ranging from electric polarization[3–5] and orbital magnetization[6–8] to topological properties[9–13]. Moreover, WFs provide a way to describe a group of energy bands in a crystal by a compact Hamiltonian, thus allowing a fast evaluation of the band structure at any point of the Brillouin zone (BZ) without an extra call to the ab initio code[14]. This procedure called Wannier interpolation is similar in spirit to the tight-binding method[15], but with the significant advantage that it offers a systematic way of precise description of any number of bands without truncation of the hopping integrals. Moreover, not only the band energies but also the matrix elements of various observables and operators, which are expressed in terms of wavefunctions and their derivatives over momentum space are precisely interpolated, while the tight-binding approach contains an unavoidable error due to the limited basis set. Wannier interpolation is particularly useful in the search for Weyl points in the band structure[11,16], and in the evaluation of momentum-space integrals of rapidly varying functions. Such integrals appear, for example, in calculations of the anomalous Hall conductivity[17], orbital magnetization[18], Boltzmann transport coefficients[19], and optical properties[20,21]. By now, Wannier interpolation schemes have been developed for many other properties that demand dense BZ samplings, such as electron-phonon coupling[22,23], gyrotropic effects[24], and spin Hall conductivity[25,26].

Due to their gauge dependence, WFs are strongly non-unique and may be constructed in multiple ways. The most popular technique is the maximal localization procedure[2,27], implemented in the well-established code Wannier90[28,29], whose development is now driven by a broad community[30]. The construction of a good set of WFs requires some careful input from the user. This includes specifying a set of trial orbitals, which serve as an initial guess for the target WFs, and, if the bands of interest do not form an isolated group, choosing the "disentanglement" energy windows[14]. Recently, there has been significant progress in the automated construction of WFs with minimal user intervention. Different techniques have been proposed, such as the optimized projection functions method[31], the selected columns of the density matrix method[32–34], and an automated method to choose trial orbitals and energy windows[35,36]. In addition, a database of Wannier Hamiltonians is currently being constructed[37]. These advances constitute significant steps towards employing Wannier interpolation for high-throughput automated calculations of electronic properties of solids.

Most of the Wannier interpolation schemes mentioned above have been implemented within the popular codes—Wannier90 (namely its post-processing module `postw90.x`) and WannierTools[11]. These codes, being well-established and widely adopted by the community, have quite broad functionalities. However, more complex materials and physical effects pose harder numerical challenges, and calculation with those codes can become quite heavy. In particular, for a system with a large number of WFs and a complicated Fermi surface, it may be hard to achieve convergence with respect to momentum-space grid in a reasonable time. When it comes to high-throughput calculations, performance becomes even more important.

In this article, I present a series of methodological improvements that allow to improve dramatically the performance of the Wannier interpolation method, without compromising its accuracy. The proposed methodology is implemented in the python code `WannierBerri` (`WB`), which is freely available to install and open for contributors (see "Code availability"). The present article covers only the core methodology of the code which procure its efficiency. The more broad scope of the code can be found on its web page and will be detailed in future publications. Interesting to note that `WannierBerri` may be equally used for Wannier interpolation and tight-binding calculations, and also offers a convenient platform for the development of more functionality. The name of the code is derived from Wannier functions and the Basque word "berri" which means "new" and enters local toponyms (e.g., Lekunberri, Ekainberri).

The high efficiency of the `WB` code is achieved by the combination of several methodological improvements. First, note that in a typical calculation for a 3D bulk material using `postw90.x`, the bottleneck is the Fourier transform, which is

[1]Department of Physics, University of Zurich, Winterthurerstrasse 190, Zurich CH-8057, Switzerland. ✉email: stepan.tsirkin@uzh.ch

implemented as a standard discrete Fourier transform. Therefore, it looks appealing to use fast Fourier transforms (FFTs)[38,39] which are widely used in numerical calculations[40,41]. However, it is problematic to do so over a very dense grid of points, which may include up to 10^8 points. This issue is overcome in this work by a mixed scheme employing both fast and "slow" Fourier transforms. Next, the symmetries of the system may be used to reduce the evaluation only to the irreducible points and obtain the contributions from other points by applying symmetry operations. This also helps to render the result more symmetric (tensor components that should be equal or vanish will be exactly equal, or exactly vanish), even if the symmetries are slightly broken due to numerical inaccuracies in wannierization. Then I introduce an adaptive refinement algorithm that identifies points that give the largest contribution to the integral and makes the grid denser in the vicinity of such points. This helps to have a more accurate description near special points, where the integrand is rapidly changing—for example, near Weyl nodes or nodal lines. This is similar in spirit to the adaptive refinement scheme used in[17,42], but is more automatic and requires less input from the user. Finally, I introduce methods that drastically reduce the computational cost of the minimal-distance replica selection method (MDRS)[30], as well as the cost of scanning over multiple Fermi levels.

The article is organized as follows. First, I describe the set of proposed improvements to the existing Wannier interpolation methodology. Next, the usage of the code is demonstrated for the "textbook" example of the anomalous Hall conductivity of bcc iron, and the performance of the `WB` code is benchmarked against postw90 on the basis of that example. Finally, a broader functionality implemented in the `WB` code is described. "Methods" describes more computational details, and the routines to obtain the matrix elements that are not implemented in the interfaces of most ab initio codes to Wannier90, but essential to calculate the properties related to spin and orbital moment of Bloch electrons.

RESULTS

General equations for Wannier interpolation

We start with a brief overview of the Wannier interpolation method, mainly with a goal to introduce notation necessary for further discussion. For more details, please refer to review ref. [2] and original articles cited therein. The problem of Wannier interpolation is stated in the following way. First, we evaluate the energies $E_{n\mathbf{q}}$ and wavefunctions $\psi_{n\mathbf{q}}(\mathbf{r}) \equiv e^{i\mathbf{q}\cdot\mathbf{r}} u_{n\mathbf{q}}(\mathbf{r})$ from first principles on a rather coarse grid of $N_\mathbf{q} = N_\mathbf{q}^1 \times N_\mathbf{q}^2 \times N_\mathbf{q}^3$ wavevectors $\mathbf{q}$ within the reciprocal unit cell. Next, we want to find the energies and wavefunctions at points on a denser grid of wavevectors $\mathbf{k}$. Further, we will consistently use $\mathbf{q}$ and $\mathbf{k}$ to denote the ab initio and interpolation grids, respectively.

For a group of entangled bands, one can define a set of J WFs defined as

$$|\mathbf{R}n\rangle = \frac{1}{N_\mathbf{q}} \sum_\mathbf{q} e^{-i\mathbf{q}\cdot\mathbf{R}} \sum_{m=1}^{\mathcal{J}_\mathbf{q}} |\psi_{m\mathbf{q}}\rangle V_{mn}(\mathbf{q}), \tag{1}$$

where $\mathcal{J}_\mathbf{q} \geq J$ and $\mathbf{R}$ are real-space lattice vectors. The matrices $V_{mn'}(\mathbf{q})$ contain all the information on the construction of WFs and may be generated by the Wannier90 code. They are constrained by $\sum_{m=1}^{\mathcal{J}_\mathbf{q}} V_{mn}^*(\mathbf{q}) V_{mn'}(\mathbf{q}) = \delta_{nn'}$ and are chosen in such a way that the WFs are localized, which yields that the Bloch wavefunctions in the Wannier gauge

$$\left|\psi_{n\mathbf{k}}^\mathrm{W}\right\rangle \equiv e^{i\mathbf{k}\cdot\mathbf{r}} \left|u_{n\mathbf{k}}^\mathrm{W}\right\rangle \equiv \sum_\mathbf{R} e^{i\mathbf{k}\cdot\mathbf{R}} |\mathbf{R}n\rangle \tag{2}$$

vary slowly with the $\mathbf{k}$ vector, unlike the true wavefunctions. Now let us see how WFs may be used to interpolate the band energies. First, one evaluates the matrix elements of the Hamiltonian

$$\begin{aligned} H_{mn}(\mathbf{R}) &\equiv \tfrac{1}{N_\mathbf{q}} \sum_\mathbf{q} e^{-i\mathbf{q}\cdot\mathbf{R}} \langle \psi_{m\mathbf{q}}^\mathrm{W} | \hat{H} | \psi_{n\mathbf{q}}^\mathrm{W} \rangle \\ &= \tfrac{1}{N_\mathbf{q}} \sum_\mathbf{q} e^{-i\mathbf{q}\cdot\mathbf{R}} \sum_l V_{lm}^*(\mathbf{q}) E_{l\mathbf{q}} V_{ln}(\mathbf{q}). \end{aligned} \tag{3}$$

Next, to obtain energies at an arbitrary point $\mathbf{k}$ one needs to construct the Wannier Hamiltonian

$$H_{mn}^\mathrm{W}(\mathbf{k}) = \sum_\mathbf{R} H_{mn}(\mathbf{R}) e^{i\mathbf{k}\cdot\mathbf{R}}, \tag{4}$$

which further may be diagonalized as

$$\sum_{mn} U_{ml}^*(\mathbf{k}) H_{mn}^\mathrm{W}(\mathbf{k}) U_{nl'}(\mathbf{k}) = E_l(\mathbf{k}) \delta_{ll'}, \tag{5}$$

where $U_{nl'}(\mathbf{k})$ are unitary matrices with columns corresponding to the eigenvectors of the Hamiltonian (4). In a similar way, for any operator $\hat{X}$, for which the matrix elements are evaluated on the ab initio grid, one may obtain the real-space matrix elements

$$X_{mn}(\mathbf{R}) \equiv \frac{1}{N_\mathbf{q}} \sum_\mathbf{q} e^{-i\mathbf{q}\cdot\mathbf{R}} X_{mn}^\mathrm{W}(\mathbf{q}), \tag{6}$$

where in a simple case (e.g., $\hat{X} = \boldsymbol{\sigma}$)

$$X_{mn}^\mathrm{W}(\mathbf{q}) = \sum_{ll'} V_{lm}^*(\mathbf{q}) \langle \psi_{m\mathbf{q}} | \hat{X} | \psi_{n\mathbf{q}} \rangle V_{l'n}(\mathbf{q}), \tag{7}$$

or if $\hat{X}$ involves momentum-space derivatives, (e.g., the position operator $\hat{r}_a \equiv i\partial/\partial k_a$) may also involve matrix elements between neighboring $\mathbf{q}$ points (see refs. [17,18] for details). Then the matrix elements may be interpolated to any $\mathbf{k}$ point in the Wannier gauge by

$$X_{mn}^\mathrm{W}(\mathbf{k}) = \sum_\mathbf{R} X_{mn}(\mathbf{R}) e^{i\mathbf{k}\cdot\mathbf{R}}, \tag{8}$$

and further rotated to the Hamiltonian gauge

$$\overline{X}_{mn}^\mathrm{H}(\mathbf{k}) = \left(U^\dagger \cdot X^\mathrm{W} \cdot U\right)_{mn}. \tag{9}$$

Note that Eqs. (3), (4), and (5) are particular cases of (6), (8), and (9). Equation (6) can be performed by means of FFT, and its result is periodic in $\mathbf{R}$ with a supercell formed by vectors $\mathbf{A}_i = \mathbf{a}_i N_\mathbf{q}^i$, where $\mathbf{a}_i$ $(i = 1, 2, 3)$ are the primitive unit cell vectors. Among the equivalent $\mathbf{R}$ vectors, we choose those belonging to the corresponding Wigner–Seitz (WS) supercell. If an $\mathbf{R}$ vector belongs to the WS supercell boundary, we include all equivalent vectors on the boundary with the corresponding elements $X(\mathbf{R})$ divided by the degeneracy of the $\mathbf{R}$ vector. Further, the MDRS method (see "Minimal-distance replica selection method (MDRS)") may also slightly modify the set of $\mathbf{R}$ vectors.

As an example, the total Berry curvature of the occupied manifold is interpolated[17] via

$$\begin{aligned} \Omega_\gamma(\mathbf{k}) &= \mathrm{Re} \sum_n^\mathrm{occ} \overline{\Omega}_{nn,\gamma}^\mathrm{H} - 2\epsilon_{\alpha\beta\gamma} \mathrm{Re} \sum_n^\mathrm{occ} \sum_l^\mathrm{unocc} D_{nl,\alpha} \overline{A}_{ln,\beta}^\mathrm{H} \\ &\quad + \epsilon_{\alpha\beta\gamma} \mathrm{Im} \sum_n^\mathrm{occ} \sum_l^\mathrm{unocc} D_{nl,\alpha} D_{ln,\beta}, \end{aligned} \tag{10}$$

where the ingredients of the equation are obtained using Eqs. (8), (9) starting from $D_{nl,\alpha} \equiv \frac{\overline{H}_{nl,\alpha}^\mathrm{H}}{E_l - E_n}$, $H_\alpha^\mathrm{W} \equiv \partial_\alpha H^\mathrm{W}$, $A_{mn,\alpha}(\mathbf{R}) \equiv \langle \mathbf{0}m | \hat{r}_\alpha | \mathbf{R}n \rangle$, $\overline{\Omega}_\gamma^\mathrm{W} \equiv \epsilon_{\alpha\beta\gamma} \partial_\alpha A_\beta^\mathrm{W}$, $\partial_\alpha \equiv \partial/\partial k_\alpha$. The anomalous Hall conductivity is evaluated as an integral

$$\sigma_{\alpha\beta}^\mathrm{AHE} = -\frac{e^2}{\hbar} \epsilon_{\alpha\beta\gamma} \int \frac{d\mathbf{k}}{(2\pi)^3} \Omega_\gamma(\mathbf{k}). \tag{11}$$

Note, that while the direct Fourier transform (6) is performed only once for the calculation, and is not repeated for the multiple $\mathbf{k}$ points upon interpolation, the inverse Fourier transform (8) is repeated for every interpolation $\mathbf{k}$ point. In fact, it presents the

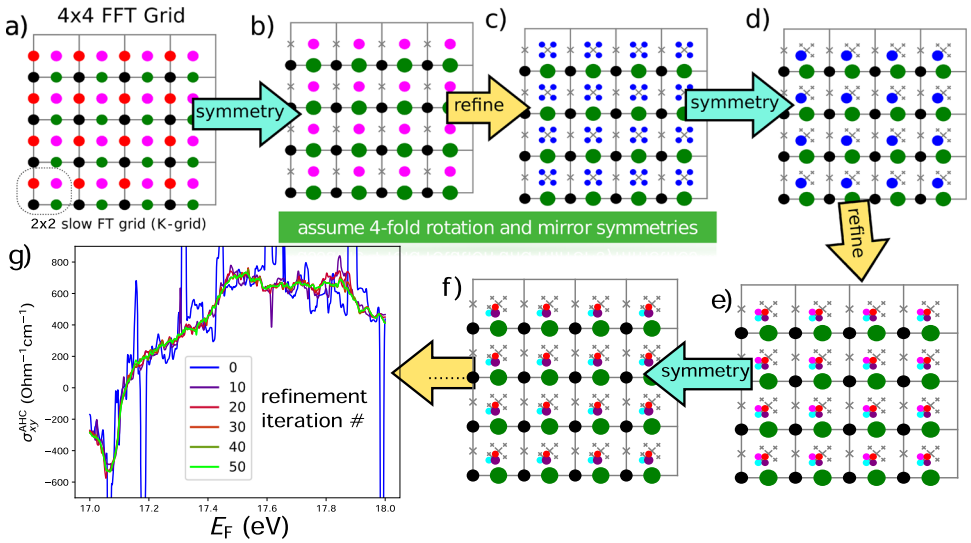

Fig. 1 Illustration of the procedure of mixed Fourier transform, adaptive refinement, and use of symmetries. a–f show the procedure step by step using a 2D picture for visualization purposes, while the code actually works in 3D. The area of colored circles corresponds to the weight of the **K** point, gray crosses denote the points with zero weight. See the text for a detailed description. **g** AHC of bcc Fe, evaluated from a grid of $50\times50\times50$ **k** points and 50 recursive adaptive refinement iterations.

most time-consuming part of the calculation of AHE in relatively small 3D bulk systems as implemented in the Wannier90 code.

Mixed Fourier transform

In this section, we will see how the evaluation of Eq. (8) may be accelerated. It is easy to see that the computation time of a straightforward discrete Fourier transform scales with the number of **R** vectors and **k** points as $t\propto N_{\mathbf{R}}N_{\mathbf{k}}$, and we are typically interested in a case $N_{\mathbf{k}}\gg N_{\mathbf{R}}$ ($N_{\mathbf{R}}\approx N_{\mathbf{q}}$).

When the Fourier transform is done on a regular grid of **k** points, it is usually appealing to use the FFT. For that one needs to place the **R** vectors on a regular grid of size $N_{\mathbf{k}}$, fill the missing spots with zeros and perform the standard FFT, for which the time will scale as $t\propto N_{\mathbf{k}}\log N_{\mathbf{k}}$. However, there are some difficulties with such FFT. Mainly, because to perform FFT on a large grid implies storing the data for all **k** points in memory at the same time, which becomes a severe computational limitation. Also, FFT does not allow to reduce computation to only the symmetry-irreducible **k** points and is more difficult to do in parallel. However, there is a way to combine the advantages of both the FFT and the usual discrete Fourier transform, leading to the concept of mixed Fourier transform.

We want to evaluate Eq. (8) for a set of **k** points.

$$\mathbf{k}_{n_1,n_2,n_3}=\frac{n_1}{N_{\mathbf{k}}^1}\mathbf{b}_1+\frac{n_2}{N_{\mathbf{k}}^2}\mathbf{b}_2+\frac{n_3}{N_{\mathbf{k}}^3}\mathbf{b}_3, \tag{12}$$

where $0\le n_i<N_{\mathbf{k}}^i$ – integers ($i=1,2,3$), $N_{\mathbf{k}}^i$—size of interpolation grid, $\mathbf{b}_i$—reciprocal lattice vectors. Now suppose we can factorize $N_{\mathbf{k}}^i=N_{\mathrm{FFT}}^iN_{\mathbf{K}}^i$. This is always possible unless $N_{\mathbf{k}}^i$ is a prime number. But for really dense grids, we can adjust $N_{\mathbf{k}}^i$ a bit, to be factorizable in any way we want. Then the set of points Eq. (12) is equivalent to a set of points $\mathbf{k}=\mathbf{K}+\boldsymbol{\kappa}$, where

$$\mathbf{K}_{l_1,l_2,l_3}=\frac{l_1}{N_{\mathbf{k}}^1}\mathbf{b}_1+\frac{l_2}{N_{\mathbf{k}}^2}\mathbf{b}_2+\frac{l_3}{N_{\mathbf{k}}^3}\mathbf{b}_3, \tag{13a}$$

$$\boldsymbol{\kappa}_{m_1,m_2,m_3}=\frac{m_1}{N_{\mathrm{FFT}}^1}\mathbf{b}_1+\frac{m_2}{N_{\mathrm{FFT}}^2}\mathbf{b}_2+\frac{m_3}{N_{\mathrm{FFT}}^3}\mathbf{b}_3, \tag{13b}$$

where $0\le l_i<N_{\mathbf{K}}^i$, $N_{\mathbf{K}}=\prod_iN_{\mathbf{K}}^i$, $0\le m_i<N_{\mathrm{FFT}}^i$. This separation is illustrated in Fig. 1a, which shows a 2×2 grid of **K** points, each corresponding to 4×4 FFT grid (dots of a certain color). Now for each **K** point, we can define

$$X_{mn}(\mathbf{K},\mathbf{R})\equiv X_{mn}(\mathbf{R})e^{i\mathbf{K}\cdot\mathbf{R}} \tag{14}$$

and then Eq. (8) reads as

$$X_{mn}^{\mathrm{W}}(\mathbf{k}=\mathbf{K}+\boldsymbol{\kappa})=\sum_{\mathbf{R}}X_{mn}(\mathbf{K},\mathbf{R})e^{i\boldsymbol{\kappa}\cdot\mathbf{R}} \tag{15}$$

The principle idea of mixed Fourier transform consists in performing the Fourier transform Eq. (15) as FFT, while Eq. (14) is performed directly. To perform the FFT, we put all the **R** vectors on a grid $N_{\mathrm{FFT}}^1\times N_{\mathrm{FFT}}^2\times N_{\mathrm{FFT}}^3$, and a vector $\mathbf{R}=\sum_{i=1}^3n_i\mathbf{a}_i$ is placed on a slot with coordinates $\tilde{n}_i=n_i \bmod N_{\mathrm{FFT}}^i$ (n_i are both positive and negative integers, while $0\le\tilde{n}_i<N_{\mathrm{FFT}}^i$). If the FFT grid is not big enough, contributions from different **R** vectors may appear on the same slot. In this case, they should be added up, and it does not affect the final results. However, for optimal performance, such a situation should be avoided at least for the majority of **R** vectors. Typically, it is a good choice to take the size of the FFT grid approximately equal to the ab initio grid.

The advantages of this approach are the following. First, the computational time scales as $t_1\propto N_{\mathbf{K}}N_{\mathbf{R}}$ for Eq. (14) and $t_2\propto N_{\mathbf{K}}N_{\mathrm{FFT}}\log N_{\mathrm{FFT}}$ for Eq. (15). Because it is required that $N_{\mathrm{FFT}}\ge N_{\mathbf{R}}$ (to fit all **R** vectors in the FFT box), we have $t_1\le t_2\propto N_{\mathbf{k}}\log N_{\mathrm{FFT}}$ (in practice it occurs that $t_1\ll t_2$), which scales better than both the Fast and "slow" Fourier transforms. Next, we can perform Eqs. (14) and (15) independently for different **K** points. This saves us memory and also offers a simple parallelization scheme. Also, we can further restrict evaluation only to symmetry-irreducible **K** points (see "Symmetries") and also perform adaptive refinement over **K** points (see "Recursive adaptive refinement").

Moreover, the evaluation time of a mixed Fourier transform only logarithmically depends on the size of the ab initio grid (recall that $N_{\mathrm{FFT}}\sim N_{\mathbf{R}}\sim N_{\mathbf{q}}$), while for the slow Fourier transform, the dependence is linear. However, in practice, we will see (see "Computation time") that the Fourier transform in the present implementation consumes only a small portion of computational time, and therefore the overall computational time is practically independent of the size of the ab initio grid.

Symmetries

When we integrate some quantity over the BZ, at every **K** point (after summing over $\boldsymbol{\kappa}$ points), we obtain the result as a rank-*m*

tensor $X_{i_1,\ldots,i_m}(\mathbf{K})$, for example, the berry curvature vector Ω_γ or the conductivity tensor σ_{xy}. Then the BZ integral is expressed as a sum

$$\mathcal{X} = \sum_{\mathbf{K}}^{\text{all}} X(\mathbf{K}) w_{\mathbf{K}} \tag{16}$$

and we initially set $\{\mathbf{K}\}$ as a regular grid (13a) and $w_{\mathbf{K}} = 1/N_{\mathbf{K}}$. Suppose G is the magnetic point group of the system. (Because $X(\mathbf{K})$ is invariant under translations, here we are interested in the point group, rather than space group.) We define the set of symmetry-irreducible $\mathbf{K}$ points *irr* as a set of points that $\forall \mathbf{K}, \mathbf{K}' \in \text{irr}, \forall\, g \in G$ holds $g\mathbf{K} \neq \mathbf{K}'$, unless $g = E$ (identity). Then we can rewrite the sum Eq. (16) as

$$\mathcal{X} = \sum_{\mathbf{K}}^{\text{all}} g_{\mathbf{K}} X(g_{\mathbf{K}}^{-1}\mathbf{K}) w_{\mathbf{K}} \tag{17}$$

where we choose $g_{\mathbf{K}}$ such that $g_{\mathbf{K}}^{-1}\mathbf{K} \in \text{irr}$ (this choice maybe not unique), and obviously $g_{\mathbf{K}} = E$ for $\mathbf{K} \in \text{irr}$. Thus, only the irreducible $\mathbf{K}$ points need to be evaluated. Next, to make sure that the result respects the symmetries, despite possible numerical inaccuracies, we symmetrize the result as:

$$\widetilde{\mathcal{X}} = \frac{1}{|G|} \sum_{f}^{G} f\mathcal{X}. \tag{18}$$

Note, that $\widetilde{\mathcal{X}} = \mathcal{X}$ if the model respects the symmetry precisely (e.g., when symmetry-adapted WFs[43] are used). Combining Eqs. (17) and (18) and using $\sum_f^G f \cdot g_{\mathbf{K}} = \sum_f^G f$, we get

$$\widetilde{\mathcal{X}} = \frac{1}{|G|} \sum_{f}^{G} f \left[\sum_{\mathbf{K}}^{\text{irr}} X(\mathbf{K}) \left(\sum_{\mathbf{K}'}^{G\cdot\mathbf{K}} w_{\mathbf{K}'} \right) \right], \tag{19}$$

where $G \cdot \mathbf{K}$ denotes the orbit of $\mathbf{K}$ under the action of group G. The latter equation reflects the implementation in the WB code. Starting from a regular grid of $\mathbf{K}$ points, we search for pairs of symmetry-equivalent points. Whenever such a pair is found, one of the points is excluded and it's weight is transferred to the other point. Compare Fig. 1a, b: the red points are removed and their weight is moved to green points. Thus we end with a set of irreducible $\mathbf{K}$ point with weights $\widetilde{w}_{\mathbf{K}} = \sum_{\mathbf{K}'}^{G\cdot\mathbf{K}} w_{\mathbf{K}'}$. Next, we evaluate $X(\mathbf{K})$ (employing the corresponding interpolation scheme) only at symmetry-irreducible $\mathbf{K}$ points. Note, that although some $\mathbf{k}$ points corresponding to the same $\mathbf{K}$ point (same color in Fig. 1) are equivalent, we have to evaluate them all to be able to use the FFT. Finally, after summation, we symmetrize the result. The described procedure achieves two goals: (i) reduce the computational costs, and (ii) make the result precisely symmetric, even if the WFs are not perfectly symmetric. In the present example, we managed to obtain highly symmetric WFs (although without the employment of symmetry-adapted WFs method), and therefore the symmetrization procedure does not change the result (within relative accuracy ~10^{-5}). However, for complex materials, such quality of WFs is not always easy to achieve.

Recursive adaptive refinement

It is well-known that in calculations of quantities involving Berry curvature (33) or orbital moment (34) of Bloch states (e.g., AHC, Berry curvature dipole (29)[44] or gyrotropic magnetoelectric tensor (30)[24,45]), one performs integration over $\mathbf{k}$-space of a function that rapidly changes with $\mathbf{k}$. As a result, small areas of $\mathbf{k}$-space give a major contribution to the integral. Such areas often appear in the vicinity of Weyl points, nodal lines, as well as avoided crossings. To accelerate convergence with respect to the number of $\mathbf{k}$ points, we utilize adaptive mesh refinement similar to refs. [17,42]. The authors of refs. [17,42] assumed a pre-defined threshold, and the $\mathbf{k}$ points yielding Berry curvature above the threshold was refined. This is inconvenient because one needs a good intuition to guess an optimal value for this threshold because it depends both on the quantity one wants to calculate, and the material considered.

In WB, it is implemented in a way that does not require an initial guess from the user. This procedure, in combination with the symmetrization described above, is illustrated in Fig. 1 in two dimensions (2D), while the actual work in 3D is described below. After excluding symmetry-equivalent $\mathbf{K}$ points (Fig. 1b), the results are evaluated for every $\mathbf{K}$ point and stored. We assume that initially each $\mathbf{K}$ point has weight $\widetilde{w}_{\mathbf{K}}$ and corresponds to a volume defined by vectors $\mathbf{c}^i_{\mathbf{K}} = \mathbf{b}_i/N^i_{\mathbf{k}}$ centered at $\mathbf{K}$. Then we pick a few "most important $\mathbf{K}$ points". The criteria of importance may be different—either the Maximal value for any E_F, or maximal value summed over all E_F, or yielding most variation over the E_F (if the evaluated quantity is a function of Fermi level E_F). Suppose we selected the magenta point, then those points are refined–replaced with eight points around it with coordinates

$$\mathbf{K}' = \mathbf{K} \pm \frac{\mathbf{c}^1_{\mathbf{K}}}{4} \pm \frac{\mathbf{c}^2_{\mathbf{K}}}{4} \pm \frac{\mathbf{c}^3_{\mathbf{K}}}{4}, \tag{20}$$

where all combinations of $\pm$ signs are used. In Fig. 1c, four new blue $\mathbf{K}$ points in the 2D case. The weight and volume of the initial point is distributed over the new points, thus $w_{\mathbf{K}'} = \widetilde{w}_{\mathbf{K}}/8$ and $\mathbf{c}^i_{\mathbf{K}'} = \mathbf{c}^i_{\mathbf{K}}/2$. Then the symmetrization is applied again (the four blue points are connected by fourfold rotation) to exclude the equivalent points, and the weight of the equivalent points is collected on the remaining point, while the vectors $\mathbf{c}^i_{\mathbf{K}'}$ are not changed. After the new $\mathbf{K}$ points are evaluated, we go to the next iteration of refinement. On each iteration, any point may be refined, including both those from the initial regular grid, and those created during previous refinement iterations. The procedure stops after the pre-defined number of iterations was performed. Figure 1g shows how undesired artificial peaks of the AHC curve are removed iteration by iteration, yielding a smooth curve (see "Example: AHC of bcc iron" for details).

Minimal-distance replica selection method (MDRS)

The MDRS method[30] allows to obtain a more accurate Wannier interpolation, in particular when moderate $\mathbf{q}$ grids are used in the ab initio calculations. With the MDRS method, the Fourier transform (8) is modified in the following way:

$$X^{\text{W}}_{mn}(\mathbf{k}) = \sum_{\mathbf{R}} \frac{1}{\mathcal{N}_{mn\mathbf{R}}} X_{mn}(\mathbf{R}) \sum_{j=1}^{\mathcal{N}_{mn\mathbf{R}}} e^{i\mathbf{k}\cdot\left(\mathbf{R}+\mathbf{T}^{(j)}_{mn\mathbf{R}}\right)}, \tag{21}$$

where $\mathbf{T}^{(j)}_{mn\mathbf{R}}$ are $\mathcal{N}_{mn\mathbf{R}}$ lattice vectors that minimize the distance $|\mathbf{r}_m - (\mathbf{r}_n + \mathbf{R} + \mathbf{T})|$ for a given set $m, n, \mathbf{R}$. However, the evaluation of Eq. (21) is quite slower than Eq. (8), because every $\mathbf{k}, m, n, \mathbf{R}$ an extra loop over j is needed. Therefore, calculations employing MDRS in postw90.x (which is enabled by default) takes more time. Instead, it is convenient to re-define the real-space matrix elements as

$$\widetilde{X}_{mn}(\mathbf{R}) = \sum_{\mathbf{R}'} \frac{1}{\mathcal{N}_{mn\mathbf{R}'}} X_{mn}(\mathbf{R}') \sum_{j=1}^{\mathcal{N}_{mn\mathbf{R}'}} \delta_{\mathbf{R},\mathbf{R}'+\mathbf{T}^{(j)}_{mn\mathbf{R}'}} \tag{22}$$

only once for the calculation, and then the transformation to $\mathbf{k}$-space is performed via

$$X^{\text{W}}_{mn}(\mathbf{k}) = \sum_{\mathbf{R}} e^{i\mathbf{k}\mathbf{R}} \widetilde{X}_{mn}(\mathbf{R}). \tag{23}$$

Note, that the set of $\mathbf{R}$ vectors in Eq. (22) is increased compared to the initial set of vectors in Eq. (6) in order to fit all nonzero elements $\widetilde{X}_{mn}(\mathbf{R})$. Equation (23) having essentially the same form as Eq. (8) can be evaluated via mixed Fourier transform, as described in section "Mixed Fourier transform".

Thus the MDRS method is implemented in WB via Eqs. (22)–(23), and has practically no extra computational cost, while giving notable accuracy improvement.

Scanning multiple Fermi levels

It is often needed to study anomalous Hall conductivity (AHC) not only for the pristine Fermi-level E_F but considering it as a free parameter ϵ. On the one hand, it gives an estimate of the accuracy of the calculation, e.g., sharp spikes may indicate that the result is not converged. On the other hand, ϵ-dependence gives access to the question of the influence of doping and temperature, and also allows calculation of anomalous Nernst effect (27). As implemented in `postw90.x`, evaluation of multiple Fermi levels has a large computational cost. However, there is a way to perform the computation of AHC for multiple Fermi levels almost without extra costs. To show this, let's rewrite Eqs. (10), (11) as $\sigma_{\alpha\beta}(\epsilon) = -\epsilon_{\alpha\beta\gamma}\frac{e^2}{\hbar}\Omega_\gamma(\epsilon)$, where $\Omega_\gamma(\epsilon) = \sum_{\mathbf{K}} w_{\mathbf{K}}\Omega_\gamma(\mathbf{K},\epsilon)$ and

$$\Omega(\mathbf{K},\epsilon) = \sum_{\boldsymbol{\kappa}}\left(\sum_{n}^{O(\mathbf{k},\epsilon)} P_n(\mathbf{k}) + \sum_{l}^{U(\mathbf{k},\epsilon)}\sum_{n}^{O(\mathbf{k},\epsilon)} Q_{ln}(\mathbf{k})\right), \tag{24}$$

where $\mathbf{k} = \mathbf{K} + \boldsymbol{\kappa}$, the definitions of P_n and Q_{ln} straightly follow from Eq. (10), and we omit the cartesian index γ further in this subsection. Now suppose we want to evaluate $\Omega(\epsilon_i)$ for a series of Fermi levels ϵ_i. For different $\mathbf{k}$-points and Fermi levels ϵ, the sets of occupied $O(\mathbf{k},\epsilon)$ and unoccupied states $U(\mathbf{k},\epsilon)$ change, and repeating these summations many times may be computationally heavy. Instead, we note that when going from one Fermi-level ϵ_i to another ϵ_{i+1} only a few states at a few $\boldsymbol{\kappa}$ points change from unoccupied to occupied. Let's denote the set of such $\boldsymbol{\kappa}$ points as $\delta\kappa_i$ then, the change of the total Berry curvature is

$$\begin{aligned}\delta\Omega_i \;&\equiv\; \Omega(\epsilon_{i+1}) - \Omega(\epsilon_i) = \\ &= \sum_{\mathbf{k}}^{\delta\kappa_i}\left(\sum_{n}^{O(\mathbf{k},\epsilon_{i+1})} P_n(\mathbf{k}) + \sum_{l}^{U(\mathbf{k},\epsilon_{i+1})}\sum_{n}^{O(\mathbf{k},\epsilon_{i+1})} Q_{ln}(\mathbf{k}) - \sum_{n}^{O(\mathbf{k},\epsilon_i)} P_n(\mathbf{k}) - \sum_{l}^{U(\mathbf{k},\epsilon_i)}\sum_{n}^{O(\mathbf{k},\epsilon_i)} Q_{ln}(\mathbf{k})\right) \\ &= \sum_{\mathbf{k}}^{\delta\kappa_i}\left(\sum_{n}^{\delta O_i(\mathbf{k})} P_n + \sum_{l}^{U(\mathbf{k},\epsilon_{i+1})}\sum_{n}^{\delta O_i(\mathbf{k})} Q_{ln}(\mathbf{k}) - \sum_{l}^{\delta O_i(\mathbf{k})}\sum_{n}^{O(\mathbf{k},\epsilon_i)} Q_{ln}(\mathbf{k})\right),\end{aligned} \tag{25}$$

where $\delta O_i(\mathbf{k}) \equiv O(\mathbf{k},\epsilon_{i+1}) - O(\mathbf{k},\epsilon_i)$. Note that if the step $\epsilon_{i+1} - \epsilon_i$ is small, then $\delta\kappa_i$ and $\delta O_i(\mathbf{k})$ include only a few elements, if not empty. Hence, the evaluation of Eq. (25) will be very fast. Thus, the full summation Eq. (24) is needed only for the first Fermi level.

In a similar way, this approach may be applied to other Fermi-sea properties such as orbital magnetization, which may be written as

$$\begin{aligned}M_\gamma(\mathbf{k}) \;&=\; \sum_{n}^{\mathrm{occ}} \mathrm{Re}\left[\overline{C}^{\mathrm{H}}_{nn,\gamma} + E_n\overline{\Omega}^{\mathrm{H}}_{nn,\gamma}\right] - \\ &\quad -2\epsilon_{\alpha\beta\gamma}\sum_{l}^{\mathrm{unocc}}\sum_{n}^{\mathrm{occ}} \mathrm{Re}\left[D_{nl,a}(\overline{B}^{\mathrm{H}}_{ln,\beta} + \overline{A}^{\mathrm{H}}_{ln,\beta}E_n)\right] \\ &\quad +\epsilon_{\alpha\beta\gamma}\mathrm{Im}\sum_{l}^{\mathrm{unocc}}\sum_{n}^{\mathrm{occ}} D_{nl,a}(E_l + E_n)D_{ln,\beta}\end{aligned} \tag{26}$$

where $C_{mn,\gamma}(\mathbf{R}) \equiv \epsilon_{\alpha\beta\gamma}\langle \mathbf{0}m|r_\alpha \cdot \hat{H}\cdot(r_\beta - R_\beta)|\mathbf{R}n\rangle$, $B_{mn,\beta}(\mathbf{R}) \equiv \langle \mathbf{0}m|\hat{H}\cdot(r_\beta - R_\beta)|\mathbf{R}n\rangle$ and the other ingredients were explained under Eq. (10). Equation (26) is written following the approach of ref. [18], but the result has a different form, which can be straightforwardly processed by analogy with Eqs. (24) and (25), where the first line of Eq. (26) expresses $P_n(\mathbf{k})$, while the second and third lines correspond to $Q_{ln}(\mathbf{k})$. Note that evaluation of $C_{mn,\gamma}(\mathbf{R})$ requires additional matrix elements evaluated on the ab initio grid, see "Evaluation of additional matrix elements" for details.

Example: AHC of bcc iron

In this section, the usage of the `WannierBerri` code is demonstrated on a simple example—anomalous Hall conductivity of bcc iron. After the Wannier functions are constructed with Wannier90 (see "Computational details"), the calculation is performed by the following short python script. First, we import the needed packages:

```
import wannierberri as WB
import numpy as np
```

Then, we read the information about the system and WFs:

```
system=WB.System_w90(Fe,berry=True)
```

from files Fe.chk, Fe.eig, Fe.mmn (the first is written by Wannier90, the other two by the interface of the ab initio code, e.g., pw2wannier90.x), or we can read all information from a file `Fe_tb.dat`, which is also written by Wannier90, or maybe composed by the user from any tight-binding model:

```
system=WB.System_tb('Fe_tb.dat',berry=True)
```

Next, we define the symmetries that we wish to take into account. In the ab initio calculation, we have specified the magnetization along the z axis, hence the symmetries that are preserved are inversion $\mathcal{I}$, fourfold rotation around the z axis C_{4z}, and a combination of time-reversal $\mathcal{T}$ and twofold rotation around the x axis C_{2x}. Here, we need only the generators of the symmetry group.

```
system.set_symmetries([
Inversion,C4z,TimeReversal*C2x])
```

The other symmetries will be automatically obtained by taking products of these generators, for example, the mirror is $M_z = (C_{4z})^2\cdot\mathcal{I}$.

Next, we need to set the grids of $\mathbf{k}$, $\mathbf{K}$, and $\boldsymbol{\kappa}$ points. Most conveniently, it can be done by setting the "length" parameter (in Å):

```
grid=WB.Grid(system,length=100)
```

This will guarantee the grid to be consistent with the symmetries, and the spacing of $\mathbf{k}$ points will be $\Delta k \approx \frac{2\pi}{\mathrm{length}}$. In this particular case, the reciprocal lattice vectors have length $|\mathbf{b}_i| = 3.1\,\text{Å}^{-1}$, hence the suggested grid size is $\frac{\mathrm{length}\cdot|\mathbf{b}_i|}{2\pi} \approx 49$. However, the grid size is adjusted to $50\times50\times50$ points in order to factorize it as $10\times10\times10\boldsymbol{\kappa}$ grid and $5\times5\times5\mathbf{K}$ grid.

Next, we want to integrate the Berry curvature to get the AHC. This is done by the `WB.integrate` method.

```
WB.integrate(system, grid,
Efermi=np.linspace(17.,18.,1001),
smearEf=10, # 10K
quantities=["ahc","cumdos"],
numproc=16,
adpt_num_iter=50,
fout_name="Fe")
```

and in addition to AHC, we evaluate the cumulative density of states (cDOS) Eq. (32). We consider Fermi level as a free parameter, scanning over a set of Fermi levels from 17 to 18 eV with a step of 1 meV, and small smearing over the Fermi level corresponding to temperature 10 K (~1 meV) is used. It is known, that in the BZ integration, some $\mathbf{k}$ points may give a large contribution to the integral. This is especially strong for Berry curvature, which blows up near band degeneracies and avoided crossings, that fall close to the Fermi level. This is reflected as huge spikes in the E_F-resolved curves—see blue curve in Fig. 1g. To make the calculation more precise around such points, an adaptive recursive refinement algorithm is used, and we set the number of iterations to 50.

From the cDOS, we can find the precise position of the Fermi level $E_F = 17.618$ eV—the energy at which the cumulative DOS reaches eight electrons per unit cell. This is more accurate than the result evaluated from a coarse ab initio grid. Next, it is instructive to plot the AHC after each iteration. In Fig. 1g, one can see that after 50 iterations the chaotic peaks are removed, and we

can get a reasonably smooth curve, although we have started from a rather coarse grid of only $50 \times 50 \times 50$ **k** points. In practice, it is still recommended to start with denser grids, when possible.

To demonstrate why the adaptive refinement is so useful for calculations of AHC, we can also visualize the Berry curvature. With the following lines

```
WB.tabulate(system, grid,
quantities=["berry"],
frmsf_name="Fe",
numproc=16,
ibands=np.arange(4,9),
Ef0=12.610)
```

we produce files Fe_berry-?.frmsf, containing the energies and Berry curvature of bands 4-8 (band counting starts from zero) tabulated over the 3D Brillouin zone. The format of the files allows being directly passed to the FermiSurfer visualization tool[46] which can produce a plot like Fig. 2, which clearly shows that Berry curvature is large in the regions where two bands come close to each other.

This short example demonstrates that the calculations with WB may be run with a few lines of Python script and the broader functionality will be detailed further.

Computation time

Now let us compare the time for the calculations of anomalous Hall conductivity using postw90.x and WannierBerri. We will take the example of bcc Fe and vary different parameters while using the same computational resource (see "Methods").

The computation consists of two phases. First, some preliminary operations are done. Those include reading the input files and performing Fourier transform from ab initio grid **q** to real-space vectors **R** : Eqs. (3), (6), and (7). This operation takes in WB (postw90.x) between 2 (3) seconds for the small **q** grid $4 \times 4 \times 4$ and 2 (3) minutes for a large grid of $16 \times 16 \times 16$. This time is mostly taken by reading the large formatted text file Fe.mmn, and it is done only once and does not scale with the density of the interpolation grid. In WB this is done in the constructor of the System_w90 class, and the object can be saved on disk using a pickle module so that this operation does not repeat for further calculations.

Next comes the interpolation part itself, for which the evaluation time scales linearly with the number of **k** points used. Further, the time for an interpolation grid $200 \times 200 \times 200$ is given, which is a rather good grid to make an accurate calculation for this material.

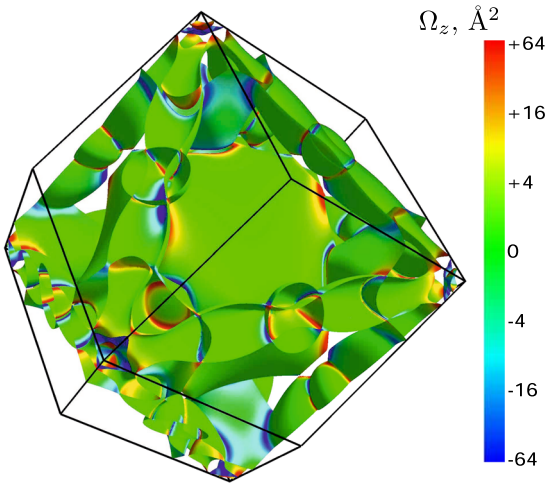

Fig. 2 Fermi surface of bcc iron. Color shows the Berry curvature Ω_z on a logarithmic scale. Figure produced using FermiSurfer[46] software.

We start with comparing time with the MDRS switched off and without the use of symmetries in WB. As can be seen in Fig. 3, for a small **q** grid $4 \times 4 \times 4$ WB is just slightly faster then postw90.x. However, for dense **q** grids the computational time of postw90.x grows linearly with the number of **q** points, while in WB it stays almost the same. This happens because in postw90.x the Fourier transform is the major time-consuming routine. On the other hand, in WB, although the cost of the mixed Fourier transform is expected to grow logarithmically with the ab initio grid, we do not see it because Fourier transform amounts only to ~10% of the computational time.

Next, we switch on the MDRS method and the computational time of postw90.x grows by a factor of 5. On the other hand, the computational time of WannierBerri does not change (not shown).

Finally, let's switch on the use of symmetries in WB. Thus the computational time decreases by a factor of 8. In the ultra-dense grid limit, one would expect the speedup to be approximately equal to the number of elements in the group—16 in the present example, due to exclusion of symmetry-equivalent **K** points. But this does not happen, because we use an FFT grid of $25 \times 25 \times 25$ $\boldsymbol{\kappa}$ points for all ab initio grids, hence the **K** grid is only $8 \times 8 \times 8$, and a considerable part of **K** points are at high-symmetry positions. Therefore they have less symmetric partners to be excluded from the calculation.

Thus we can see that the difference in computational time with postw90.x and WB reaches three orders of magnitude for this example. Note that the examples above were performed only for the pristine Fermi level. Now let's see what happens upon scanning the Fermi levels (Fig. 4). In WB, the computational time remains practically unchanged when we use up to $N_\epsilon \approx 1000$ Fermi levels, and only start to grow considerably at $N_\epsilon \sim 10^4$. On the other hand in postw90.x, the computational time significantly grows with N_ϵ, which is especially remarkable for small **q** grids, where the growth becomes linear already from $N_\epsilon \sim 10$. For denser **q** grids, the fixed amount of time (independent of N_ϵ) is larger, so the linear growth starts at higher N_ϵ.

In this section, we did not use the adaptive refinement procedure. However, when one starts from a rather large grid of **K** points, the new **K** points coming from the refinement procedure constitute only a small portion of the initial grid, and hence do not contribute much to computation time.

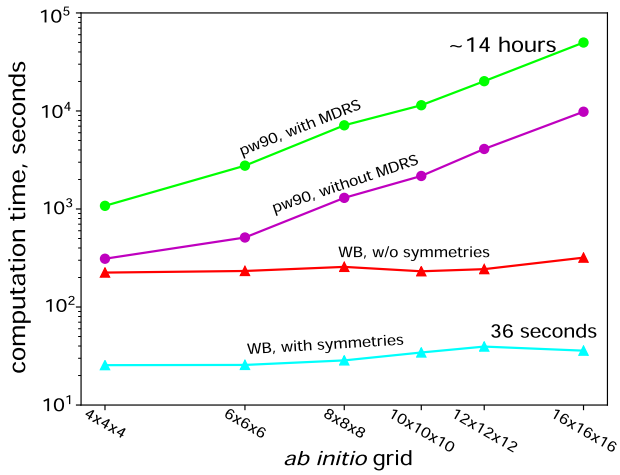

Fig. 3 Computational time for AHC. Time for calculations with WB (triangles) and postw90.x (circles) for different ab initio grids. For postw90.x, the calculations are done with (green) and without (purple) MDRS. For WB, the calculations are done with (cyan) and without (red) use of symmetries.

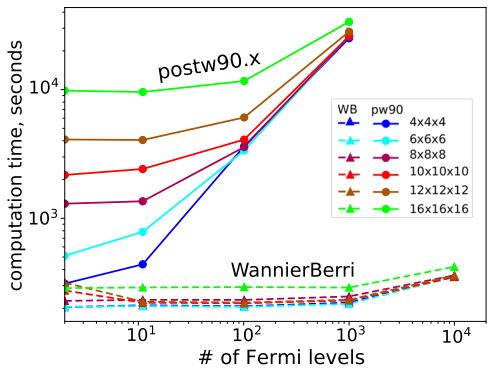

Fig. 4 Computational time for scanning multiple Fermi levels. Time for calculations with `WB` (dashed lines) and `postw90.x` (pw90, solid lines) are shown for different ab initio grids. MDRS method and symmetries are disabled here.

We have shown that the methods suggested in this article help to significantly reduce the computation time from days to minutes. However, bcc iron is a simple example with only one atom per unit cell, and only 18 WFs are needed. More complicated systems will require more time. For example, to obtain a converged value of anomalous Nernst conductivity in PrAlGe[47] using `w`annier19 (an early version of `WB`), the calculation took approximately 30 h on the same computation node. Estimates predict that the same calculation with `postw90.x` could take several months. Thus it is the case where the numerical advance not only saves time but also brings the calculation from the unreasonably time-consuming area, where most people would avoid working, to a reasonably feasible computation time.

The functionality implemented in `WannierBerri`

This appendix outlines the functionality that is currently implemented in `WannierBerri`. For more detailed and updated information, please refer to the online documentation.

The code may be used to evaluate the following quantities, represented as Brillouin zone integrals:

- "`ahc`" : intrinsic anomalous Hall conductivity $\sigma^{\mathrm{AHE}}_{\alpha\beta}$[48] via Eq. (11)
- Anomalous Nernst conductivity[49] $\alpha^{\mathrm{ANE}}_{\alpha\beta}$ may be obtained from $\sigma_{\alpha\beta}(\epsilon)^{\mathrm{AHE}}$ evaluated over a dense grid of Fermi levels ϵ

$$\alpha^{\mathrm{ANE}}_{\alpha\beta} = -\frac{1}{e}\int d\varepsilon \frac{\partial f}{\partial \varepsilon}\sigma^{\mathrm{AHE}}_{\alpha\beta}(\varepsilon)\frac{\varepsilon-\mu}{T}, \tag{27}$$

where $f(\varepsilon) = 1/\left(1+e^{\frac{\varepsilon-\mu}{k_B T}}\right)$;

- "`Morb`" : orbital magnetization

$$M^{\gamma}_{n}(\mathbf{k}) = \frac{e}{2\hbar}\mathrm{Im}\ \epsilon_{\alpha\beta\gamma}\int [d\mathbf{k}]\sum_{n}^{\mathrm{occ}}[\langle\partial_a u_{n\mathbf{k}}|H_{\mathbf{k}}+E_{n\mathbf{k}}-2E_{\mathrm{F}}|\partial_b u_{n\mathbf{k}}\rangle]; \tag{28}$$

- "`berry_dipole`": Berry curvature dipole

$$D_{\alpha\beta}(\mu) = \int [d\mathbf{k}]\sum_{n}^{\mathrm{occ}}\partial_\alpha \Omega^{\beta}_{n}, \tag{29}$$

which describes nonlinear Hall effect[44];

- "`gyrotropic_Korb`" and `gyrotropic_Kspin`: gyrotropic magnetoelectric effect (GME)[45] tensor (orbital and spin contributions):

$$K_{\alpha\beta}(\mu) = \int [d\mathbf{k}]\sum_{n}^{\mathrm{occ}}\partial_\alpha m^{\beta}_{n}; \tag{30}$$

- "`conductivity_Ohmic`": ohmic conductivity within the Boltzmann transport theory in constant relaxation time (τ) approximation:

$$\sigma^{\mathrm{Ohm}}_{\alpha\beta}(\mu) = \tau\int [d\mathbf{k}]\sum_{n}^{\mathrm{occ}}\partial_\alpha E_{n\mathbf{k}}\partial_\beta E_{n\mathbf{k}}\delta(E_{n\mathbf{k}}-\mu) = \tau\int [d\mathbf{k}]\sum_{n}^{E_{n\mathbf{k}}<\mu}\partial^2_{\alpha\beta}E_{n\mathbf{k}}; \tag{31}$$

- "`dos`": density of states $n(E)$;
- "`cumdos`": cumulative density of states

$$N(E) = \int_{-\infty}^{E} n(\epsilon)d\epsilon. \tag{32}$$

For a more complete and updated list, please refer to the documentation at [wberri-org].

Currently, the following quantities are available to tabulate:

- "`berry`": Berry curvature

$$\Omega^{\gamma}_{n}(\mathbf{k}) = -\epsilon_{\alpha\beta\gamma}\mathrm{Im}\ \langle\partial_\alpha u_{n\mathbf{k}}|\partial_\beta u_{n\mathbf{k}}\rangle; \tag{33}$$

- "`morb`": orbital moment of Bloch states

$$m^{\gamma}_{n}(\mathbf{k}) = \frac{e}{2\hbar}\epsilon_{\alpha\beta\gamma}\mathrm{Im}\ \langle\partial_\alpha u_{n\mathbf{k}}|H_{\mathbf{k}}-E_{n\mathbf{k}}|\partial_\beta u_{n\mathbf{k}}\rangle; \tag{34}$$

- "`spin`": the expectation value of the Pauli operator

$$\mathbf{s}_n(\mathbf{k}) = \langle u_{n\mathbf{k}}|\hat{\boldsymbol{\sigma}}|u_{n\mathbf{k}}\rangle; \tag{35}$$

- "`V`": the band gradients $\nabla_{\mathbf{k}}E_{n\mathbf{k}}$.

DISCUSSION

In this article, I have presented a series of methods that boost the performance of Wannier interpolation to a higher level. The methods are implemented in the Python code `WannierBerri`. It is important to note that the mixed Fourier transform and the optimization of MDRS method and Fermi-level iteration while giving a large computational advantage, do not affect the result within machine precision. Hence, `WannierBerri` can be easily benchmarked with the established `postw90.x` code. The code not only allows to perform high-speed and high-precision calculations of AHC and a palette of other properties but also serves as a platform for implementing more functionalities involving Wannier interpolation. Thus it has the potential to become a community code. Interestingly, the code uses the same routines to perform calculations both based on WFs and tight-binding models. Finally, in combination with recent advances in automated construction of WFs, it paves a way to high-throughput calculations of properties of solids that require Wannier interpolation.

METHODS

Computational details

The input files for `WannierBerri` are prepared by a combination of an ab initio code and Wannier90. In this article, we used the QuantumEspresso (QE) code[50], and closely followed the Tutorial#18 of Wannier90. First, we perform self-consistent calculations on a grid $16\times16\times16$ **q** points, fixing the magnetization along [001] axis. We use Generalized gradient approximation (GGA)[51] for exchange-correlation functional and a norm-conserving pseudopotential from the PseudoDojo library[52,53], and spin–orbit coupling is taken into account. Next, we perform a non-self-consistent calculations using a Γ-centered grid of $8\times8\times8$ **q** points. Further, we define the set of trial orbitals $sp_3d_2$2, d_{xy}, d_{xz}, and d_{yz}, which set the initial guess for 18 WFs describing the conduction band of Fe. And prepare the input files for Wannier90 by means of the `pw2wannier90.x` interface. Finally, the maximally localized Wannier functions are constructed by the Wannier90 code. The resulting files may be used to start calculations both by `postw90.x` and `WannierBerri`. Further, we repeat this procedure for a series of non-self-consistent **q** grids. The reader may refer to the documentation of Wannier90 and QE for more details.

Calculations were performed on identical 32-core virtual nodes of the ScienceCloud cluster at the University of Zürich. The nodes are based on

AMD EPYC 7702 64-Core Processors with frequency 2GHz and 128 GB RAM per node, and one node was used per task. The computation time reported in Figs. 3 and 4 refers to this computational setup.

External libraries used in the code

The `WannierBerri` code is developed in the object-oriented Python3 language and makes use of the popular numerical libraries, which include `NumPy`[54], `SciPy`[55], and `lazy-propery`. Fast Fourier transform is implemented via libraries `pyFFTW` (wrapper of FFTW3 library[56]) and `NumPy`, and the user has an option to choose between them, although no notable difference in performance has been observed. The calculations are performed in parallel over **K** points by means of the `multiprocessing` module and the parameter "numproc" specifies that a `Pool` of 16 worker processes is used.

Evaluation of additional matrix elements

Wannier interpolation starts from certain matrix elements defined on the ab initio (**q**) grid. Those matrix elements should be evaluated within the ab initio code, namely within its interface to Wannier90. However, only QuantumEspresso[50] has the most complete interface `pw2wannier90.x`. The other codes provide only the basic interface, which includes the eigenenergies $E_{n\mathbf{q}}$ (.eig file) and overlaps

$$M^{\mathbf{b}}_{mn}(\mathbf{q}) = \langle u_{m\mathbf{q}} | u_{n\mathbf{q}+\mathbf{b}} \rangle \tag{36}$$

(file .mmn), where **b** vector connects neighboring **q** points. This information allows to interpolate the band energies (and their derivatives of any order) as well as Berry connections[20] and Berry curvature[17]. However, to evaluate the orbital magnetization via Eq. (26) or the orbital moment of Bloch states (34), one needs matrix elements of the Hamiltonian[18] (.uHu file)

$$C^{\mathbf{b}_1,\mathbf{b}_2}_{mn}(\mathbf{q}) = \langle u_{m\mathbf{q}+\mathbf{b}_1} | \hat{H}_{\mathbf{q}} | u_{n\mathbf{q}+\mathbf{b}_2} \rangle . \tag{37}$$

The evaluation of Eq. (37) is very specific to the details of the ab initio code, and implemented only in `pw2wannier90.x` and only for norm-conserving pseudopotentials. To enable the study of properties related to the orbital moment with other ab initio codes, the following workaround may be employed. By inserting a complete set of Bloch states at a particular **q** point $\mathbb{1} = \sum_l^{\infty} |u_{l\mathbf{q}}\rangle\langle u_{l\mathbf{q}}|$, we can rewrite Eq. (37) as

$$C^{\mathbf{b}_1,\mathbf{b}_2}_{mn}(\mathbf{q}) \approx \sum_{l}^{l_{\max}} \left(M^{\mathbf{b}_1}_{lm}(\mathbf{q})\right)^* E_{l\mathbf{q}} M^{\mathbf{b}_2}_{ln}(\mathbf{q}). \tag{38}$$

This equation is implemented within the `wannierberri.mmn2uHu` submodule, which allows to generate the .uHu file out of .mmn and .eig files. The equality in Eq. (38) is exact only in the limit $l_{\max} \to \infty$ and infinitely large basis set for the wavefunctions representation. So in practice, one has to check convergence for a particular system. As an example, we calculate the orbital magnetization of bcc Fe using the `.uHu` file computed with `pw2wannier90.x` and using the `wannierberri.mmn2uHu` interface with different summation limit $l_{\max}$ in (38). As can be seen in Fig. 5 already $l_{\max} = 200$ (corresponding energy ~230 eV) yields a result very close to that of `pw2wannier90.x`. However, one should bear in mind that convergence depends on many factors, such as the choice of WFs and pseudopotentials. In particular, for tellurium, we observed[24] that including only a few bands above the highest *p*-band is enough to obtain accurate results. However for iron, using a pseudopotential shipped with the examples of Wannier90, we failed to reach convergence even with $l_{\max} = 600$.

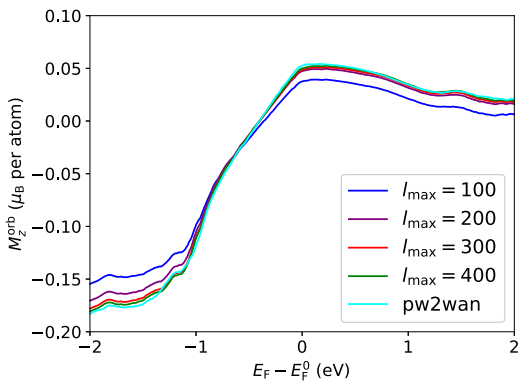

Fig. 5 Orbital magntization of bcc Fe. Given as a function of the Fermi level E_F relative to the pristine Fermi level E_F^0. Curves evaluated using the `.uHu` file computed with `pw2wannier90.x` (dashed line) and using the `wannierberri.mmn2uHu` interface (solid lines) with different summation limit $l_{\max}$ in (38).

To interpolate the spin operator expectation value, the matrix $s_{mn}(\mathbf{q}) = \langle u_{m\mathbf{q}} | \hat{\sigma} | u_{n\mathbf{q}} \rangle$ is needed. To facilitate study of spin-dependent properties within VASP[57] code, a submodule `wannierberri.vaspspn` is included, which computes $S_{mn\mathbf{q}}$ based on the normalized pseudo-wavefunction read from the `WAVECAR` file. Note that the use of pseudo-wavefunction instead of the full PAW[58] wavefunction is an approximation, which however in practice gives a rather accurate interpolation of spin.

The mmn2uHu and `vaspspn` modules were initially developed and used in ref. [24] as separate scripts, but were not published so far. Now they are included in the `WannierBerri` package with the hope of being useful to the community.

DATA AVAILABILITY

All input files needed to generate the Wannier functions and perform the example of the code, as well as the resulting output data plotted in the figures may be freely downloaded from the Materials Cloud Archive (https://doi.org/10.24435/materialscloud:1r-8w).

CODE AVAILABILITY

All software used and developed in this article are open-source and available for free. WannierBerri is available to install via PyPI repository https://pypi.org/project/wannierberri and open for contributions via GitHub https://github.com/stepan-tsirkin/wannier-berri. Updated documentation is available at http://wannier-berri.org/. External libraries used in `WB` are available via PyPI repository (https://pypi.org). Wannier90[28–30] is available at http://www.wannier.org/. QuantumEspresso[50] is available at https://www.quantum-espresso.org/. Figures were produced using `matplotlib`[59] (Figs. 1, 3–5) and FermiSurfer[46] (Fig. 2), and also with help of Inkscape vector graphics editor (https://inkscape.org).

Received: 5 October 2020; Accepted: 11 January 2021;
Published online: 19 February 2021

REFERENCES

1. Wannier, G. H. The structure of electronic excitation levels in insulating crystals. *Phys. Rev.* **52**, 191–197 (1937).
2. Marzari, N., Mostofi, A. A., Yates, J. R., Souza, I. & Vanderbilt, D. Maximally localized Wannier functions: theory and applications. *Rev. Mod. Phys.* **84**, 1419–1475 (2012).
3. Vanderbilt, D. & King-Smith, R. D. Electric polarization as a bulk quantity and its relation to surface charge. *Phys. Rev. B* **48**, 4442–4455 (1993).
4. King-Smith, R. D. & Vanderbilt, D. Theory of polarization of crystalline solids. *Phys. Rev. B* **47**, 1651–1654 (1993).
5. Resta, R. Macroscopic polarization in crystalline dielectrics: the geometric phase approach. *Rev. Mod. Phys.* **66**, 899–915 (1994).
6. Thonhauser, T., Ceresoli, D., Vanderbilt, D. & Resta, R. Orbital magnetization in periodic insulators. *Phys. Rev. Lett.* **95**, 137205 (2005).
7. Ceresoli, D., Thonhauser, T., Vanderbilt, D. & Resta, R. Orbital magnetization in crystalline solids: multi-band insulators, chern insulators, and metals. *Phys. Rev. B* **74**, 024408 (2006).
8. Xiao, D., Chang, M.-C. & Niu, Q. Berry phase effects on electronic properties. *Rev. Mod. Phys.* **82**, 1959–2007 (2010).
9. Soluyanov, A. A. & Vanderbilt, D. Computing topological invariants without inversion symmetry. *Phys. Rev. B* **83**, 235401 (2011).
10. Bradlyn, B. et al. Topological quantum chemistry. *Nature* **547**, 298–305 (2017).
11. Wu, Q., Zhang, S., Song, H.-F., Troyer, M. & Soluyanov, A. A. WannierTools: An open source software package for novel topological materials. *Comput. Phys. Commun.* **224**, 405–416 (2018).
12. Bouhon, A., Black-Schaffer, A. M. & Slager, R.-J. Wilson loop approach to fragile topology of split elementary band representations and topological crystalline insulators with time-reversal symmetry. *Phys. Rev. B* **100**, 195135 (2019).
13. Varnava, N., Souza, I. & Vanderbilt, D. Axion coupling in the hybrid Wannier representation. *Phys. Rev. B* **101**, 155130 (2020).
14. Souza, I., Marzari, N. & Vanderbilt, D. Maximally localized Wannier functions for entangled energy bands. *Phys. Rev. B* **65**, 035109 (2001).
15. Slater, J. C. & Koster, G. F. Simplified LCAO method for the periodic potential problem. *Phys. Rev.* **94**, 1498–1524 (1954).

16. Gosálbez-Martínez, D., Souza, I. & Vanderbilt, D. Chiral degeneracies and Fermi-surface Chern numbers in bcc Fe. *Phys. Rev. B* **92**, 085138 (2015).
17. Wang, X., Yates, J. R., Souza, I. & Vanderbilt, D. Ab initio calculation of the anomalous Hall conductivity by Wannier interpolation. *Phys. Rev. B* **74**, 195118 (2006).
18. Lopez, M. G., Vanderbilt, D., Thonhauser, T. & Souza, I. Wannier-based calculation of the orbital magnetization in crystals. *Phys. Rev. B* **85**, 014435 (2012).
19. Pizzi, G., Volja, D., Kozinsky, B., Fornari, M. & Marzari, N. BoltzWann: a code for the evaluation of thermoelectric and electronic transport properties with a maximallylocalized Wannier functions basis. *Comput. Phys. Commun.* **185**, 422–429 (2014).
20. Yates, J. R., Wang, X., Vanderbilt, D. & Souza, I. Spectral and Fermi surface properties from Wannier interpolation. *Phys. Rev. B* **75**, 195121 (2007).
21. Ibanez-Azpiroz, J., Tsirkin, S. S. & Souza, I. Ab initio calculation of the shift photocurrent by Wannier interpolation. *Phys. Rev. B* **97**, 245143 (2018).
22. Giustino, F., Cohen, M. L. & Louie, S. G. Electron-phonon interaction using Wannier functions. *Phys. Rev. B* **76**, 165108 (2007).
23. Poncé, S., Margine, E., Verdi, C. & Giustino, F. EPW: electron-phonon coupling, transport and superconducting properties using maximally localized Wannier functions. *Comput. Phys. Commun.* **209**, 116–133 (2016).
24. Tsirkin, S. S., Puente, P. A. & Souza, I. Gyrotropic effects in trigonal tellurium studied from first principles. *Phys. Rev. B* **97**, 035158 (2018).
25. Qiao, J., Zhou, J., Yuan, Z. & Zhao, W. Calculation of intrinsic spin Hall conductivity by Wannier interpolation. *Phys. Rev. B* **98**, 214402 (2018).
26. Ryoo, J. H., Park, C.-H. & Souza, I. Computation of intrinsic spin Hall conductivities from first principles using maximally localized Wannier functions. *Phys. Rev. B* **99**, 235113 (2019).
27. Marzari, N. & Vanderbilt, D. Maximally localized generalized Wannier functions for composite energy bands. *Phys. Rev. B* **56**, 12847 (1997).
28. Mostofi, A. A. et al. wannier90: a tool for obtaining maximally-localised Wannier functions. *Comput. Phys. Commun.* **178**, 685–699 (2008).
29. Mostofi, A. A. et al. An updated version of wannier90: a tool for obtaining maximally localised Wannier functions. *Comput. Phys. Commun.* **185**, 2309–2310 (2014).
30. Pizzi, G. et al. Wannier90 as a community code: new features and applications. *Matter* **32**, 165902 (2020).
31. Mustafa, J. I., Coh, S., Cohen, M. L. & Louie, S. G. Automated construction of maximally localized Wannier functions for bands with nontrivial topology. *Phys. Rev. B* **94**, 125151 (2016).
32. Damle, A., Lin, L. & Ying, L. Compressed representation of Kohn-Sham orbitals via selected columns of the density matrix. *J. Chem. Theory Comput.* **11**, 1463–1469 (2015).
33. Damle, A. & Lin, L. Disentanglement via entanglement: a unified method for Wannier localization. Multiscale model. *Sim* **16**, 1392–1410 (2018).
34. Vitale, V. et al. Automated high-throughput Wannierisation. *npj Comput. Mater.* **6**, 66 (2020).
35. Zhang, Z. et al. High-throughput screening and automated processing toward novel topological insulators. *J. Phys. Chem. Lett.* **9**, 6224–6231 (2018).
36. Garrity, K. F. & Choudhary, K. Database of Wannier Tight-binding Hamiltonians using high-throughput density functional theory. http://arxiv.org/abs/2007.01205 [cond-mat.mtrl-sci] (2020).
37. Garrity, K.F., Choudhary, K. JARVIS-WannierTB : database Wannier Tight-binding hamiltonian derived properties of 3D and 2D materials, online https://jarvis.nist.gov/jarviswtb/. Accessed 06/02/2021.
38. Cooley, J. W. & Tukey, J. W. An algorithm for the machine computation of the complex Fourier series. *Math. Comput.* **19**, 297–301 (1965).
39. Heideman, M., Johnson, D. & Burrus, C. Gauss and the history of the fast Fourier transform. *IEEE ASSP Mag.* **1**, 14–21 (1984).
40. Duhamel, P. & Vetterli, M. Fast fourier transforms: a tutorial review and a state of the art. *Signal Process.* **19**, 259–299 (1990).
41. Van Loan, C. *Computational Frameworks for the Fast Fourier Transform* (Society for Industrial and Applied Mathematics, 1992).
42. Yao, Y. et al. First principles calculation of anomalous Hall conductivity in ferromagnetic bcc Fe. *Phys. Rev. Lett.* **92**, 037204 (2004).
43. Sakuma, R. Symmetry-adapted Wannier functions in the maximal localization procedure. *Phys. Rev. B* **87**, 235109 (2013).
44. Sodemann, I. & Fu, L. Quantum nonlinear Hall effect induced by Berry curvature dipole in time-reversal invariant materials. *Phys. Rev. Lett.* **115**, 216806 (2015).
45. Zhong, S., Moore, J. E. & Souza, I. Gyrotropic magnetic effect and the magnetic moment on the Fermi surface. *Phys. Rev. Lett.* **116**, 077201 (2016).
46. Kawamura, M. FermiSurfer: Fermi-surface viewer providing multiple representation schemes. *Comput. Phys. Commun.* **239**, 197–203 (2019).
47. Destraz, D. et al. Magnetism and anomalous transport in the Weyl semimetal PrAlGe: possible route to axial gauge fields. *npj Quantum Mater.* **5**, 5 (2020).
48. Nagaosa, N., Sinova, J., Onoda, S., MacDonald, A. H. & Ong, N. P. Anomalous Hall effect. *Rev. Mod. Phys.* **82**, 1539–1592 (2010).
49. Xiao, D., Yao, Y., Fang, Z. & Niu, Q. Berry-phase effect in anomalous thermoelectric transport. *Phys. Rev. Lett.* **97**, 026603 (2006).
50. Giannozzi, P. et al. Quantum ESPRESSO toward the exascale. *J. Chem. Phys.* **152**, 154105 (2020).
51. Perdew, J. P., Burke, K. & Ernzerhof, M. Generalized gradient approximation made simple. *Phys. Rev. Lett.* **77**, 3865–3868 (1996).
52. van Setten, M. et al. The PseudoDojo: training and grading a 85 element optimized norm-conserving pseudopotential table. *Comput. Phys. Commun.* **226**, 39–54 (2018).
53. Hamann, D. R. Optimized norm-conserving Vanderbilt pseudopotentials. *Phys. Rev. B* **88**, 085117 (2013).
54. Oliphant, T. E. *A Guide to NumPy* (Trelgol Publishing, 2006).
55. Virtanen, P. et al. SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nat. Methods* **17**, 261–272 (2020).
56. Frigo, M. & Johnson, S. G. The design and implementation of FFTW3. In Fawwaz T. Ulaby (ed) *Proceedings of the IEEE* 93. Special issue on "Program Generation, Optimization, and Platform Adaptation", 216–231 (IEEE, 2005).
57. Kresse, G., et al. Vienna Ab initio Simulation Package (VASP) code. online https://www.vasp.at/. Accessed 06/12/2021.
58. Blöchl, P. E. Projector augmented-wave method. *Phys. Rev. B* **50**, 17953–17979 (1994).
59. Hunter, J. D. Matplotlib: a 2D graphics environment. *Comput. Sci. Eng.* **9**, 90–95 (2007).

ACKNOWLEDGEMENTS

I thank Ivo Souza and Cheol-Hwan Park for useful discussions and comments helping to improve the paper. I also acknowledge Xiaoxiong Liu, Miguel Ángel Jiménez Herrera, Patrick M. Lenggenhager, Jae-Mo Lihm, and Minsu Ghim for fixing bugs and further development of the code, which will be described elsewhere. I acknowledge support from the Swiss National Science Foundation (grant number: PP00P2_176877), the NCCR Marvel and the European Union's Horizon 2020 research and innovation program (ERC-StG-Neupert-757867-PARATOP).

AUTHOR CONTRIBUTIONS

The work was done by a single author, including the development of methodology, development of the core of the WannierBerri code, testing, and preparing the manuscript. Other researchers made contributions to the code, which are not used and not described in this paper.

COMPETING INTERESTS

The author declares no competing interests.

ADDITIONAL INFORMATION

Correspondence and requests for materials should be addressed to S.S.T.

Reprints and permission information is available at http://www.nature.com/reprints

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.