



ZStar

Reproducible polarization, Born effective charge, phonon, infrared, Raman, and dielectric-response workflows.

PyPI v0.2.0

Python 3.9+

License GPL-3.0

[English](#) | [简体中文](#) | [English PDF](#) | [中文 PDF](#)

Overview

ZStar is a Python workflow toolkit that connects ABACUS + PYATB, VASP, CP2K, Quantum ESPRESSO, and Phonopy calculations to reproducible polarization and dielectric-response results. Its main task is to turn atomic response data into symmetry-consistent Born effective charge (BEC) tensors, then use those tensors for phonon, infrared (IR), dielectric, and Raman analysis.

The toolkit keeps every stage visible: structures, solver inputs, band-gap gates, polarization values, charge-density data, tensor reconstruction reports, spectra, and progress records remain available for inspection and restart.

The numerical checks used for the current release are summarized in [docs/validation.md](#).

Main capabilities

- Forward and central finite-difference BEC calculations.
- Symmetry reduction, full-cell tensor reconstruction, and acoustic-sum-rule correction.
- A serial, resumable `0.no-move -> displaced structures` execution model.
- Reuse of the converged `0.no-move` charge density for every displacement.
- One-time insulating-state check after the reference SCF.
- Shell, Slurm, and Torque driver generation.
- Automatic compatibility with legacy and direct-static-response PyATB versions.
- A serial, resumable CP2K backend for Berry-phase BEC tensors and native APT checks.
- Three-dimensional, hybrid two-dimensional, and hybrid one-dimensional polarization/BEC analysis.
- Phonon generation, post-processing, mode classification, IR spectra, Raman spectra, and dielectric response.

- Auxiliary electrostatic-potential analysis for slabs and polar materials.
- A packaged, standards-compliant Agent Skill with JSON workspace preflight.
- A calculator-neutral response schema and backend plugin registry.
- Native Quantum ESPRESSO DFPT collection for molecular and bulk BEC/IR data.

The calculator-independent interface, physical `dim=0/1/2/3` contract, QE workflow, density adapters, Phonopy interchange, polarized Raman, optical constants, and dimensional response normalization are documented in the [calculator-independent guide](#).

Physical Scope

Three-dimensional crystals

For a periodic 3D crystal, ZStar evaluates

$$Z^*(\kappa, \alpha, \beta) = \Omega/e * dP_{\alpha} / du_{\kappa, \beta}$$

from Berry-phase polarization differences. Polarization branches are matched modulo the polarization quantum before the finite difference is taken.

One-dimensional wires and nanowires

For a `z`-periodic wire, ZStar combines PYATB Berry polarization along `z` with real-space ABACUS cube dipoles along the nonperiodic `x/y` directions. It reports vacuum-independent BEC tensors and `Angstrom^2` line polarizabilities, and supports Gamma-point IR and Raman spectra. A bulk NAC is explicitly rejected because finite-wavevector polar phonons require a genuine 1D Coulomb cutoff. See the [one-dimensional workflow](#).

Two-dimensional slabs

A slab requires separate treatment of in-plane and out-of-plane response:

- **In-plane polarization columns:** Berry-phase polarization is used while the full supercell remains insulating.
- **Out-of-plane polarization column:** the total slab dipole is integrated from the ABACUS charge-density cube, including ionic and electronic contributions.
- **Normalization:** in-plane BECs are made independent of vacuum height through the usual volume factor; 2D dielectric spectra are reported as sheet polarizability unless an effective thickness is supplied.

Canonical ZStar tensors store atomic displacement/force as rows and polarization/electric field as columns. Accordingly, a complete 2D BEC calculation needs `x`, `y`, and `z` displacements. The default `zstar bec pre --dim 2` workflow generates all three. The current hybrid implementation requires the slab normal to align with Cartesian `z`; a tilted slab is rejected explicitly.

One reference/displaced cube pair can be audited independently:

```
zstar polar2d --reference-cube reference.cube \  
  --displaced-cube atom_zplus.cube \  
  --displacement 0.01 --outdir slab_dipole_check
```

This writes the planar charge redistribution, dipole finite difference, effective charge, diagnostics, and PNG/PDF/SVG plots.

Installation

ZStar requires Python 3.9 or newer.

Install the released package:

```
pip install -U zstar
```

Or install a local checkout:

```
git clone https://github.com/xdzhu/zstar.git  
cd zstar  
pip install .
```

External programs are required only for the corresponding workflows:

- ABACUS for SCF, charge density, force, and sparse-matrix calculations.
- PyATB for Berry-phase polarization, band checks, and electronic dielectric response.
- Phonopy for displacement generation and phonon post-processing.
- CP2K for the optional CP2K finite-displacement BEC backend.
- VASP for native bulk BEC and mode-displaced dielectric-response workflows.
- Quantum ESPRESSO for the optional native DFPT BEC, dielectric, and IR route.

Verify the command:

```
zstar --version  
zstar --help
```

Configure calculator executables once per project (or add `--user` for a user configuration), then let generated drivers select `mpirun` or `srun`:

```
zstar config init  
zstar config set executables.abacus /opt/abacus/bin/abacus  
zstar config set executables.pyatb /opt/pyatb/bin/pyatb  
zstar config check  
zstar backend list --check
```

Agent Skill

Install the bundled, standards-compliant `$run-zstar-workflows` skill after installing ZStar:

```
zstar skill install
zstar skill preflight --root . --lane bec --dim bulk
```

Open a new agent session after installation. Use `--force` to refresh the skill after upgrading ZStar, or `--dest /path/to/skills` for a custom compatible skill directory. The skill encodes dimensional conventions, reference-first execution, restart behavior, scheduler authorization boundaries, and artifact-based completion checks. See [docs/agent_skill.md](#).

The canonical CLI, compatibility aliases, configuration precedence, and every public utility family are summarized in the [command-line reference](#).

Born Charge Workflow

1. Generate the reference and displacement folders

Run this in a directory containing `STRU`:

```
zstar bec pre --calculator abacus --stru STRU --pyatb --method forward --force
```

For a 2D slab:

```
zstar bec pre --calculator abacus --stru STRU --dim 2 --pyatb --method forward --force
```

For a `z`-periodic wire:

```
zstar bec pre --calculator abacus --stru STRU --dim 1 --pyatb --method central --force
```

The generated tree starts with `0.no-move`, followed by atom/direction folders such as `1.Ti/x+`. No per-displacement scheduler script is required.

Useful generation options:

Option	Meaning
<code>--method forward central</code>	One-sided or central finite difference.
<code>--reduce / --all</code>	Symmetry-reduced atoms (default) or every atom.
<code>--move "x y z"</code>	Explicit displacement directions.
<code>--displacement 0.01</code>	Finite-displacement half-step in Angstrom.

Option	Meaning
<code>--dim 0 1 2 3</code>	Molecular, one-dimensional, two-dimensional, or three-dimensional analysis.
<code>--input-mode</code> <code>abacus pyatb hamgnn custom</code>	Input preparation route.
<code>--input_sets FILES</code>	Extra files or directories copied into generated tasks.

2. Run the serial, resumable calculation

Local shell execution:

```
zstar bec run --root . \
  --abacus-command "mpirun -np 1 abacus" \
  --pyatb-command "mpirun -np 1 pyatb" \
  --omp-threads 28
```

For a wire or slab, use `--dim 1` or `--dim 2`, respectively. For an isolated molecule, use `--dim 0`; the workflow uses a Gamma-only supercell and reports atomic polar tensors (APT), not periodic-crystal BEC. For low-dimensional polarization/BEC runs, ZStar writes `out_chg 1 10` so real-space transverse dipole differences are not limited by ABACUS cube rounding.

The default execution order is:

1. Run `0.no-move` SCF and save charge density and sparse matrices.
2. Generate a normal PyATB high-symmetry band path with `pyatb_input --band`.
3. Stop before any displacement if the reference is metallic.
4. Calculate reference polarization and electronic dielectric response.
5. Copy the reference charge cube/restart into each target `OUT.<suffix>/.`
6. Run every displacement serially and calculate its polarization.
7. Record stage state under `.zstar/` so an interrupted run can resume.

The regular band path is the default lightweight gate. It can detect a gap closure on the sampled path but cannot exclude an off-path metallic pocket. A denser MP-grid check is explicit:

```
zstar bec run --root . --gap-mode mp --mp-density 0.08
```

Inspect progress at any time:

```
zstar bec stat --root .
```

3. Generate scheduler-specific drivers

Shell:

```
zstar bec job --root . --system shell
```

Slurm:

```
zstar bec job --root . --system slurm \  
  --queue compute --tasks 28 --cpus-per-task 1 --walltime 24:00:00
```

Torque/PBS:

```
zstar bec job --root . --system torque \  
  --queue batch --tasks 28 --cpus-per-task 1 --walltime 24:00:00
```

Backend-aware defaults use `mpirun -np N` for shell/Torque and `srun --ntasks=N` for Slurm. Add `--dry-run` to validate the generated script, environment, ordering, and resume records without launching a calculation. The three backends and their scheduler acceptance checks are recorded in [docs/validation.md](#).

Add `--submit` only when the generated script has been reviewed and the active environment provides the required executables.

4. Collect polarization and construct BEC tensors

Molecular APT from ABACUS + PYATB:

```
zstar bec pre --calculator abacus --dim 0 --method central --displacement 0.01 --pyatb  
zstar bec run --root . \  
  --abacus-command abacus --pyatb-command pyatb --omp-threads 20  
zstar bec post --root .
```

The molecular collector reconstructs symmetry-equivalent atoms, enforces translational invariance, and writes `molecular_apt.json` plus the normalized `zstar_response.json`. It also reconstructs small polarization signals from PYATB's separately printed ionic and electronic phases when the rounded final polarization line is insufficient.

Three-dimensional:

```
zstar bec post --root .
```

Two-dimensional hybrid treatment:

```
zstar bec post --root .
```

One-dimensional hybrid treatment for a `z`-periodic wire:

```
zstar bec post --root .
```

The manifest carries the selected dimensionality and difference method from `pre` into later actions. The old `gen/workflow/deal` commands remain accepted for compatibility.

Key outputs:

File	Meaning
<code>Z-BORN-reduced.out</code>	Raw tensors for explicitly calculated symmetry representatives.
<code>Z-BORN-symm.out</code>	Full-cell tensors reconstructed by symmetry and corrected by the acoustic sum rule.
<code>Z-BORN-reduced-neutral.out</code>	Reduced tensors after reconstruction and neutrality correction.
<code>BORN</code>	Electronic dielectric tensor plus Phonopy-order BEC tensors.
<code>BORN-for-phonopy.out</code>	Explicitly named copy of the Phonopy-compatible data.
<code>born_symmetry_report.json</code>	Machine-readable reconstruction and residual report.
<code>zstar_2d_bec.json</code>	Per-atom diagnostics for hybrid 2D BEC calculations.
<code>zstar_1d_bec.json</code>	Per-atom diagnostics for hybrid 1D BEC calculations.
<code>molecular_apt.json</code>	Molecular APT tensors, symmetry expansion, and translational-sum diagnostics.

CP2K BEC Backend

For a molecular (`--dim 0`) or three-dimensional insulating Gamma-point CP2K input, ZStar can build APT or BEC tensors directly from dipoles:

```
zstar bec pre --calculator cp2k --input input.inp --root cp2k_bec --dim 0 \
  --method central --displacement 0.005
zstar bec run --root cp2k_bec --cp2k-command cp2k.smp \
  --omp-threads 20 --data-dir /path/to/cp2k/data
zstar bec post --root cp2k_bec
```

The reference wavefunction is reused by every displacement, and interrupted runs resume from `.zstar/cp2k_bec_state.json`. CP2K 2025.2+ native `APT_FD` results can be generated and compared through `cp2k-bec native` and `cp2k-bec compare`. See the [complete CP2K guide](#) for tensor conventions, input restrictions, convergence checks, and direct-node validation.

VASP BEC Backend

ZStar can also drive VASP's native BEC implementations. Use `dfpt` for `LEPSILON` linear response or `finite-field` for `LCALCEPS`/PEAD:

```
zstar bec pre --calculator vasp --input-dir vasp_input --root vasp_bec --method dfpt
zstar bec run --root vasp_bec --vasp-command "mpirun -np 20 vasp_std"
zstar bec post --root vasp_bec
```

The reference SCF is checked for a finite band gap before the response stage, and completed stages are resumable. The collector writes normalized JSON, ZStar tensors, and a Phonopy-compatible **BORN** file. The [complete VASP guide](#) documents finite-field safeguards, cluster scripts, tensor conventions, and the VASP 6.3.2 SiC validation.

Phonons and Dielectric Response

1. Generate phonon calculations

In a phonon working directory containing **STRU**, **KPT**, and an **INPUT** with **cal_force 1**:

```
zstar phonon pre --root . --calculator abacus \
  --stru STRU --dim "2 2 2" --symprec 1e-3
zstar phonon run --root .
```

Run every generated **disp-*** force calculation with the local execution system. **zstar ph** does not require or duplicate an **abacus_x.sh**; if one is present it is copied only as an optional convenience.

2. Post-process forces and classify Gamma modes

```
zstar phonon stat --root .
zstar phonon post --root .
zstar phonon irrep --root . --file irreps.yaml --mode db
```

For non-analytical corrections, copy the BEC workflow's **BORN** into the phonon directory before post-processing:

```
cp ../polar/BORN .
zstar phonon post --root . --nac
```

3. Static and frequency-dependent dielectric response

Copy the full tensors as well:

```
cp ../polar/BORN .
cp ../polar/Z-BORN-symm.out .
```

Static response:

```
zstar dielectric static --qpoints qpoints.yaml --born Z-BORN-symm.out \  
--dielectric BORN --dim 3
```

Frequency-dependent response:

```
zstar dielectric freq --qpoints qpoints.yaml --born Z-BORN-symm.out \  
--dielectric BORN --dim 3
```

The command writes the zero-frequency tensor, real and imaginary response data, and PNG/PDF/SVG plots by default. Use `--no-plot` for data-only processing. Modes below 5 cm⁻¹ are excluded by default; change this with `--acoustic-cutoff`.

For 2D, omit `--thickness` to obtain a vacuum-independent sheet polarizability in angstroms:

```
zstar dielectric static --qpoints qpoints.yaml --born Z-BORN-symm.out \  
--dielectric BORN --dim 2
```

Provide `--thickness ANGSTROM` only when an effective 3D dielectric tensor is desired. The complete convention, bulk and two-dimensional examples, and output contract are documented in the [dielectric-response guide](#).

Infrared Spectrum

Calculate mode effective charges, oscillator strengths, broadened IR intensity, and dielectric/sheet response:

```
zstar spectra pre --calculator abacus --kind ir --root ir_spectrum \  
--qpoints qpoints.yaml \  
--born Z-BORN-symm.out --dielectric BORN \  
--dim 3  
zstar spectra post --root ir_spectrum
```

The retained low-level expert command exposes mode selection and plotting details when needed:

```
zstar ir --modes "4,5,8-10" --outdir ir_selected
```

Typical outputs are `ir_modes.csv`, `ir_spectrum.dat`, `ir_response_real.dat`, `ir_response_imag.dat`, `ir_spectrum.png`, `ir_spectrum.pdf`, `ir_spectrum.svg`, and `ir_summary.json`.

Raman Spectrum

ZStar obtains non-resonant Raman tensors by central finite differences of the electronic dielectric response along Gamma-point normal coordinates.

1. Prepare mode displacements

```
zstar spectra pre --calculator abacus --kind raman --root raman \  
  --stru STRU --qpoints qpoints.yaml \  
  --modes "4-12" --amplitude 0.02 \  
  --copy INPUT-scf --copy KPT
```

The amplitude is a normal-coordinate displacement in `angstrom * sqrt(amu)`.

2. Run, collect, and plot

```
zstar spectra run --root raman --reference 0.no-move \  
  --abacus-command "mpirun -np 1 abacus" \  
  --pyatb-command "mpirun -np 1 pyatb" \  
  --omp-threads 28  
zstar spectra stat --root raman  
zstar spectra post --root raman
```

The reference insulating gap is reused once; it is not repeated for every mode displacement. Each `plus/minus` stage reuses the reference charge density and records resumable state.

The low-level `zstar raman collect` and `zstar raman spectrum` commands remain available for expert reprocessing of an existing mode tree.

For 2D, `--dim 2` converts the vacuum-dependent dielectric derivative to a sheet-susceptibility derivative using the cell height stored in the phonon data.

Molecular IR and Raman

For the complete physical convention, workflow, outputs, and benchmark, see [Molecular IR and Raman spectroscopy](#).

An isolated molecule is represented in a sufficiently large periodic cell. Prepare its positive-frequency vibrational modes with the same central normal-coordinate displacements used for Raman calculations:

```
zstar spectra pre --calculator abacus --kind all --root raman --dim 0 \  
  --stru STRU --qpoints qpoints.yaml \  
  --acoustic-cutoff 100 --amplitude 0.02 \  
  --copy INPUT-scf --copy KPT
```

The complete resumable calculation produces both spectra:

```
zstar spectra run --root raman --reference 0.no-move \
  --abacus-command "mpirun -np 1 abacus" \
  --pyatb-command "mpirun -np 1 pyatb" \
  --spectrum-outdir raman_spectrum --ir-outdir ir_spectrum
zstar spectra post --root raman
```

Each displaced SCF is evaluated with two lightweight PYATB stages. The static dielectric response is converted to a molecular polarizability derivative, $d\alpha/dQ = V/(4\pi) * d(\epsilon_{\text{r}})/dQ$. Branch-wrapped Berry polarization is converted to a molecular dipole derivative, $d\mu/dQ = V * dP/dQ$. The normal-coordinate step Q is in $\text{\AA} * \sqrt{\text{amu}}$.

Existing polarization results can still be reprocessed with the low-level expert commands:

```
zstar ir --dim 0 --qpoints qpoints.yaml \
  --displacements raman --outdir ir_spectrum
zstar raman spectrum --dim 0 --qpoints qpoints.yaml \
  --raman-dir raman --outdir raman_spectrum
```

Molecular spectra are normalized for mode assignment and workflow validation. They are not reported as gas-phase integrated cross sections. Use a converged vacuum size, basis, displacement amplitude, and electronic-response grid for quantitative intensity work.

VASP and CP2K calculators

The calculator-neutral spectroscopy layer also supports VASP and CP2K:

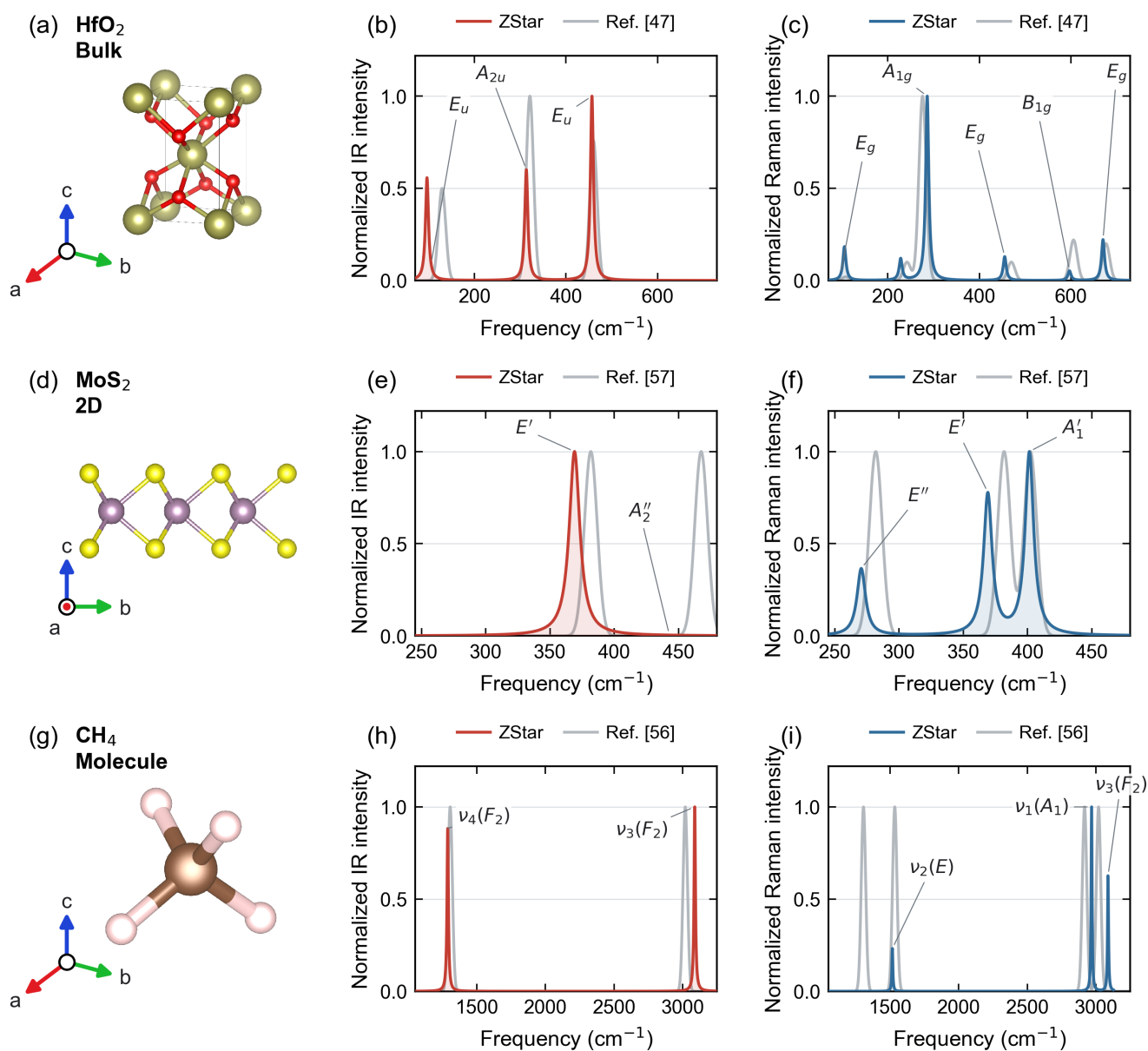
```
zstar spectra pre --calculator vasp --input-dir vasp_input \
  --modes-xml phonon/vasprun.xml --root vasp_spectra --dim 3
zstar spectra run --root vasp_spectra --command "mpirun -np 20 vasp_std"
zstar spectra post --root vasp_spectra

zstar spectra pre --calculator cp2k --input h2o.inp \
  --root cp2k_spectra --dim 0
```

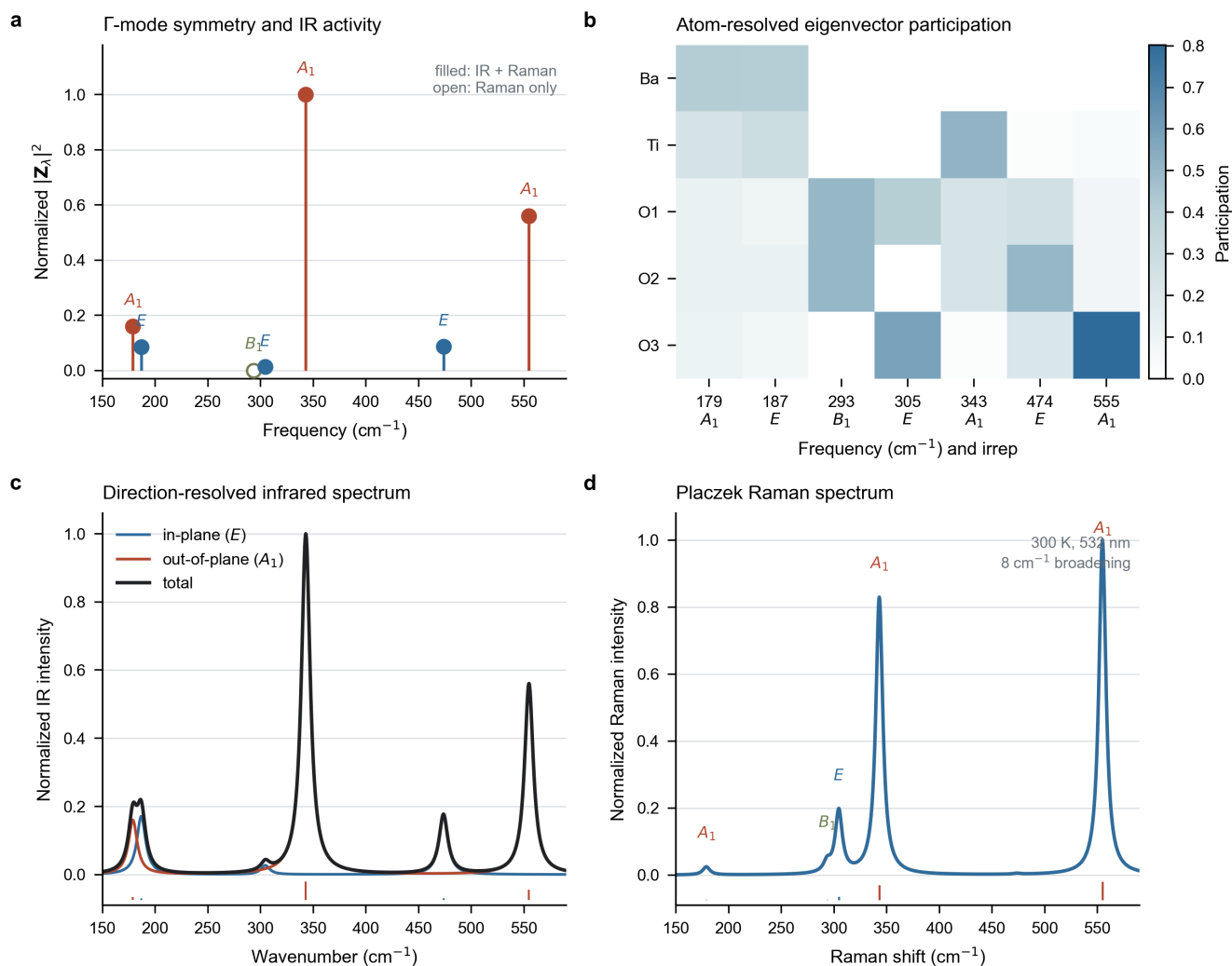
VASP uses central differences of native dielectric responses; CP2K uses native vibrational dipole and **LINRES/POLAR** intensities. See the [calculator spectroscopy guide](#).

Representative Validation Figures

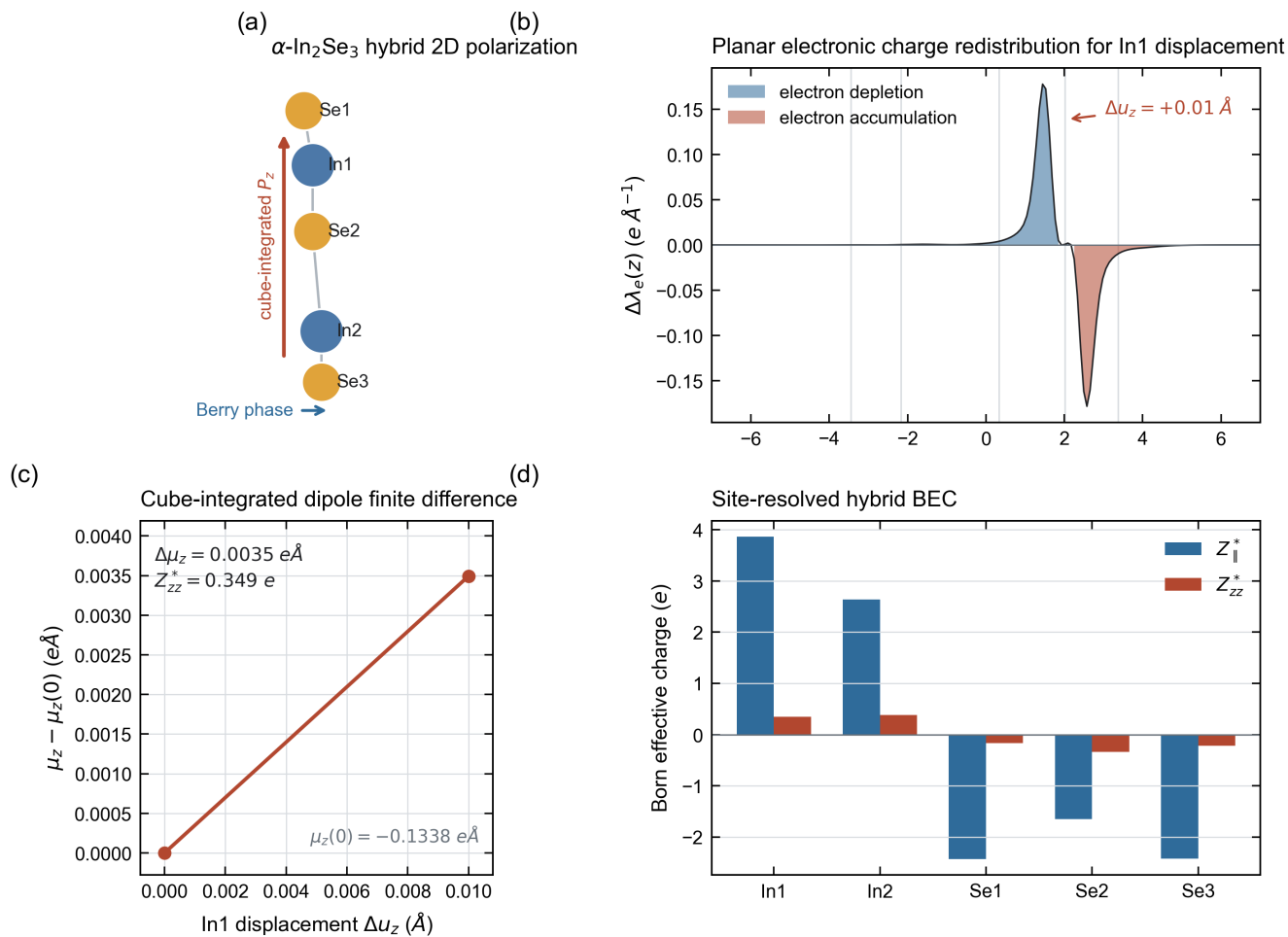
The compact source data, plotting script, vector files, and integrity manifest are archived in [docs/paper_figures](#).



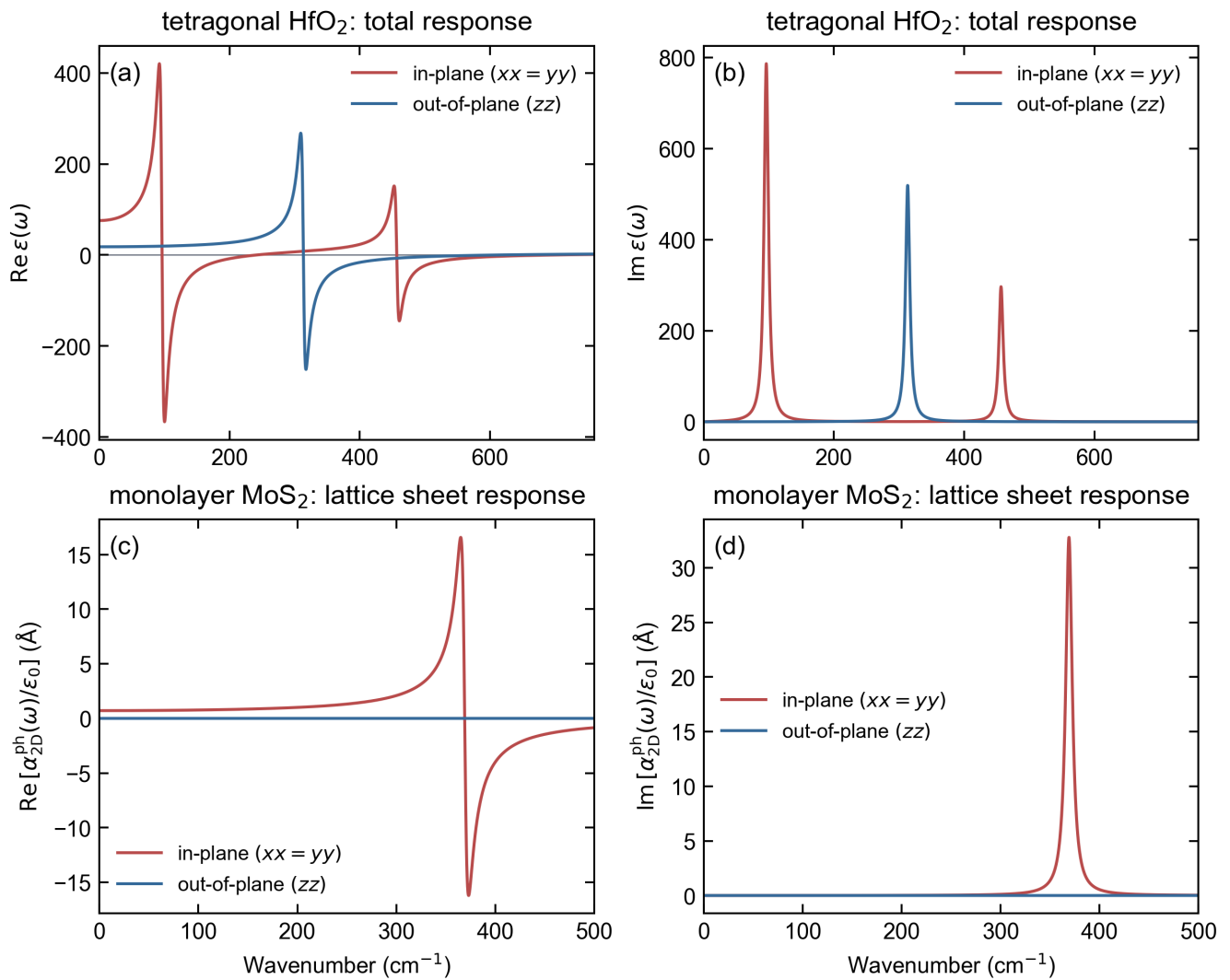
The three-row comparison follows the manuscript order: tetragonal HfO₂ (**Bulk**), monolayer MoS₂ (**2D**), and CH₄ (**Molecule**). The PBEsol HfO₂ row contains all 15 stable optical modes and 30 completed Raman response stages. The refreshed ABACUS/PBE-D3(BJ) MoS₂ row combines all six optical modes with production BEC-derived IR intensities and 12 completed central-difference Raman response stages.



The BTO validation includes all ten positive-frequency optical modes and 20 completed Raman finite-difference response tasks. The **B1** mode at 293.38 cm^{-1} is Raman active and IR silent.



The In₂Se₃ validation displays the Berry-phase/cube-integral split and the actual planar charge redistribution for an out-of-plane In displacement.



The HfO₂ panels show the total bulk response including the electronic background: the PBEsol/TZDP 9-au closure gives `epsilon(0) = diag(75.761034, 75.761034, 18.045191)`. The MoS₂ panels show the intrinsic lattice sheet polarizability; they are not a vacuum-dependent supercell dielectric constant.

PyATB Compatibility

ZStar probes the PyATB executable selected for the workflow:

- New builds with direct static response use `static_dielectric_only`.
- Older builds use a compact 0-30 eV optical grid at 0.1 eV spacing. This range was checked against the new direct-static intercept; the coarse spacing avoids an unnecessarily dense full optical spectrum.
- Both `static_dielectric_function.dat` and legacy `dielectric_function_real_part.dat` are accepted.

The selected mode and detected version are saved in `zstar_pyatb_compat.json`.

Electrostatic Potential

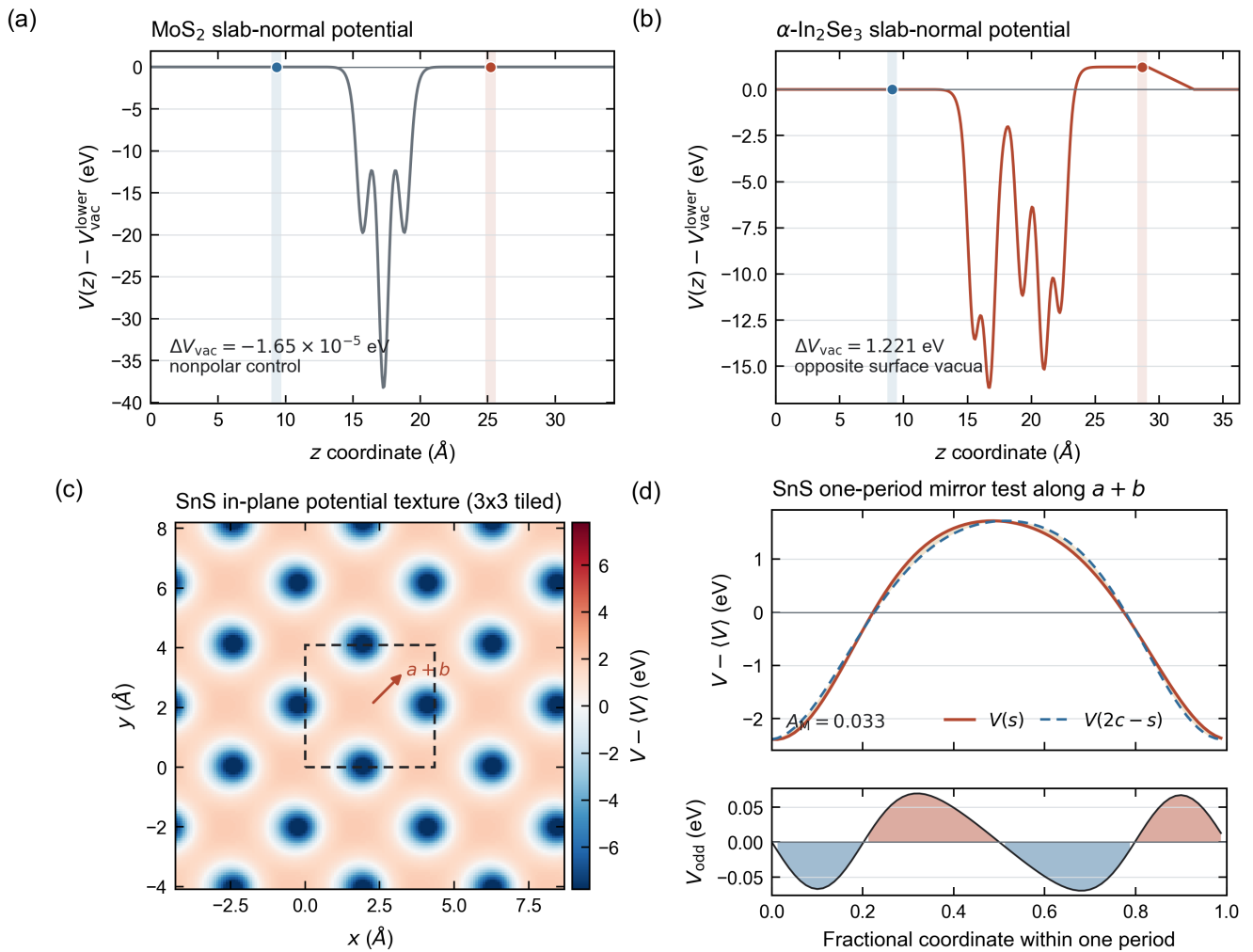
`zstar_pot` analyzes ABACUS `ElecStaticPot.cube` files:

```

zstar pot --cube OUT.ABACUS/ElecStaticPot.cube \
  --axes z --plane xy --plane-average --tile 5 5 \
  --vacuum-level --vacuum-sides --vacuum-window 0.75 \
  --direction a+b --mirror-test \
  --polar-arrow auto \
  --outdir potential

```

It can generate axis profiles, tiled planar maps, directional averages, one- or two-sided vacuum-level diagnostics, and an optimized one-period mirror asymmetry metric. Local side windows avoid mixing a dipole-correction reset into a polar-slab surface plateau. In the representative tests, MoS₂ has $\Delta V_{\text{vac}} = -1.65\text{e-}5 \text{ eV}$, while $\alpha\text{-In}_2\text{Se}_3$ has $\Delta V_{\text{vac}} = 1.220812 \text{ eV}$.



Commands, interpretation limits, and the SnS/SnSe/SnTe directional examples are collected in [docs/potential_examples.md](#).

Command Map

Command	Purpose
<code>zstar bec pre/run/job/stat/post</code>	Polarization, APT/BEC, BORN , resume state, and scheduler drivers.
<code>zstar phonon pre/run/job/stat/post/irrep</code>	Displacements, serial forces, force constants, frequencies, and irreps.
<code>zstar spectra pre/run/job/stat/post</code>	Calculator-aware IR and Raman workflows.
<code>zstar dielectric static/freq/optics</code>	Static, vibrational, and electronic dielectric response.
<code>zstar backend list</code>	List capabilities and optionally check executables/plugins.
<code>zstar config init/show/set/check</code>	Configure calculator executable paths.
<code>zstar response</code>	Validate and normalize calculator-neutral response data.
<code>zstar density</code>	Prepare density-export adapters and provenance sidecars.
<code>zstar stru convert/wyckoff</code>	Convert structures or inspect Wyckoff positions.
<code>zstar data qnep/db</code>	Export qNEP data or manage a traceable BEC/High-K database.
<code>zstar skill install/path/preflight</code>	Install the Agent Skill or inspect a workspace.
<code>zstar pot</code>	Plot potential profiles/maps, vacuum steps, and mirror asymmetry.

See the [complete CLI reference](#) for aliases, leaf actions, and executable-resolution rules.

Repository and Release Policy

- `examples/` contains local validation data and is intentionally ignored.
- `dist/` and `build/` are local build products and are not committed.
- `job_scripts/` contains reusable scheduler templates and is version controlled.
- `docs/how_to_update_pypi.md` records the release procedure.

The relative logo works for authenticated viewers of a private GitHub repository. PyPI requires a public HTTPS image URL, so `README_PYPI.md` deliberately omits private images.

Citation and License

If ZStar supports published work, please cite the ZStar software article or repository release together with the underlying electronic-structure and lattice-dynamics programs used in the calculation.

Machine-readable citation metadata for this release are provided in [CITATION.cff](#).

ZStar is distributed under the GNU General Public License v3.0.

Copyright (c) Xudong Zhu.