doped: Python toolkit for robust and repeatable charged defect supercell calculations

Seán R. Kavanagh
Harvard University

Defects are a universal feature of crystalline solids, dictating the key properties and performance of many functional materials. Given their crucial importance yet inherent difficulty in measuring experimentally, electronic structure methods are widely used to predict defect behaviour at the atomic level and the resultant impact on macroscopic properties. Here we report *doped*,[1] a Python package for the generation, pre-/post-processing and analysis of defect supercell calculations. *doped* has been built to implement the defect simulation workflow in an efficient, user-friendly yet powerful and fully-flexible manner, with the goal of providing a robust general-purpose platform for conducting reproducible calculations of solid-state defect properties.
Beyond the core functionality of *doped* in generating defect supercells and competing phases, writing calculation input files, parsing calculation outputs and analysing/plotting defect-related properties, some key advances include:

- Efficient supercell generation outperforming widely-used & established algorithms.
- Efficient charge-state estimation, outperforming the most widely-used current approaches in terms of accuracy and efficiency.
- Efficient competing phase selection based on controllable error tolerances.
- Automated symmetry & degeneracy handling for defect thermodynamics & analysis
- Automated compatibility checking & calculation sanity checking
- Streamlined thermodynamic analysis; formation energy diagrams, (non-)equilibrium Fermi level solving, doping analysis, Brouwer-type diagrams, 2D temperature & chemical potential heatmaps…
- Finite-size corrections; both the isotropic Freysoldt (FNV)[2] and anisotropic Kumagai (eFNV)[3]
- Reproducibility & tabulation; natively supports & encourages reproducibility (all I/O JSONable aiding provenance tracking, tabulated summary outputs…)
- Native compatibility with high-throughput architectures (AiiDA[4], atomate(2)[5]…) and other python-based defect analysis codes (ShakeNBreak[6], easyunfold[7], nonrad[8], CarrierCapture.jl[9], pydefect[10]…)

Some additional features include directional-dependent site displacement (local strain) analysis, deterministic & informative defect naming, molecule generation for gaseous competing phases, multiprocessing for expedited generation & parsing, shallow defect analysis (via *pydefect*[10]), Wyckoff site analysis (including arbitrary/interstitial sites), controllable defect site placement to aid visualisation and more. The functionality and recommended usage of *doped* is demonstrated in the tutorials (https://doped.readthedocs.io/en/latest/Tutorials.html) on the documentation website (https://doped.readthedocs.io/en/latest/).

1 S. R. Kavanagh, A. G. Squires, A. Nicolson, I. Mosquera-Lois, A. M. Ganose, B. Zhu, K. Brlec, A. Walsh and D. O. Scanlon, doped: Python toolkit for robust and repeatable charged defect supercell calculations, *Journal of Open Source Software*, 2024, **9**, 6433.
2 C. Freysoldt, J. Neugebauer and C. G. Van de Walle, Fully *Ab Initio* Finite-Size Corrections for Charged-Defect Supercell Calculations, *Phys. Rev. Lett.*, 2009, **102**, 016402.
3 Y. Kumagai and F. Oba, Electrostatics-based finite-size corrections for first-principles point defect calculations, *Phys. Rev. B*, 2014, **89**, 195205.
4 S. P. Huber, S. Zoupanos, M. Uhrin, L. Talirz, L. Kahle, R. Häuselmann, D. Gresch, T. Müller, A. V. Yakutovich, C. W. Andersen, F. F. Ramirez, C. S. Adorf, F. Gargiulo, S. Kumbhar, E. Passaro, C. Johnston, A. Merkys, A. Cepellotti, N. Mounet, N. Marzari, B. Kozinsky and G. Pizzi, AiiDA 1.0, a scalable computational infrastructure for automated reproducible workflows and data provenance, *Scientific Data*, 2020, **7**, 300.
5 K. Mathew, J. H. Montoya, A. Faghaninia, S. Dwarakanath, M. Aykol, H. Tang, I. Chu, T. Smidt, B. Bocklund, M. Horton, J. Dagdelen, B. Wood, Z.-K. Liu, J. Neaton, S. P. Ong, K. Persson and A. Jain, Atomate: A high-level interface to generate, execute, and analyze computational materials science workflows, *Computational Materials Science*, 2017, **139**, 140–152.
6 I. Mosquera-Lois, S. R. Kavanagh, A. Walsh and D. O. Scanlon, ShakeNBreak: Navigating the defect configurational landscape, *Journal of Open Source Software*, 2022, **7**, 4817.
7 B. Zhu, S. R. Kavanagh and D. Scanlon, easyunfold: A Python package for unfolding electronic band structures, *Journal of Open Source Software*, 2024, **9**, 5974.
8 M. E. Turiansky, A. Alkauskas, M. Engel, G. Kresse, D. Wickramaratne, J.-X. Shen, C. E. Dreyer and C. G. Van de Walle, Nonrad: Computing Nonradiative Capture Coefficients from First Principles, *arXiv:2011.07433 [cond-mat]*.
9 S. Kim, S. N. Hood, P. van Gerwen, L. D. Whalley and A. Walsh, CarrierCapture.jl: Anharmonic Carrier Capture, *Journal of Open Source Software*, 2020, **5**, 2102.
10 Y. Kumagai, N. Tsunoda, A. Takahashi and F. Oba, Insights into oxygen vacancies from high-throughput first-principles calculations, *Phys. Rev. Materials*, 2021, **5**, 123803.