




graph-pes: graph-based machine-learning models for potential-energy surfaces

John L. A. Gardner ¹ and Volker L. Deringer ¹

¹ Department of Chemistry, University of Oxford, Oxford, United Kingdom

DOI: [10.21105/joss.09329](https://doi.org/10.21105/joss.09329)

Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

Editor: [Yuanqing Wang](#)  

Reviewers:

- [@HectorMozo3110](#)

Submitted: 22 June 2025

Published: 27 February 2026

License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](#)).

Summary

We present graph-pes, an open-source toolkit for accelerating the development, training, and deployment of machine-learned interatomic potential (MLIP) models that act on graph representations of atomic structures. The toolkit comprises three components:

1. **The graph_pes Python package:** a modular framework containing all functionality required to build, train, and evaluate graph-based MLIPs. The package includes a mature data pipeline for converting atomic structures into graph representations (AtomicGraphs), a fully featured base class for MLIP implementations (GraphPESModel), and a suite of common data manipulation routines and model building blocks.
2. **The graph-pes-train command-line interface (CLI):** a convenience tool for training graph-based MLIPs on datasets of labelled atomic structures directly from the command line. The tool is compatible with any GraphPESModel (i.e., those defined in graph-pes, user-designed ones, and foundation models) and is designed to be easily extensible via custom loss functions, optimisers, datasets, and more.
3. **Molecular-dynamics drivers** that allow any GraphPESModel to be used in GPU-accelerated MD simulations. We currently provide a pair style for LAMMPS ([Thompson et al., 2022](#)), a GraphPESCalculator for ASE ([Larsen et al., 2017](#)), and an integration with the torch-sim package ([Cohen et al., 2025](#)).

Statement of need

In recent years, machine-learned PES models have become central tools for computational chemistry and materials science ([Deringer et al., 2019](#)). These models are trained on labels generated by quantum-mechanical methods, but scale much more favourably with system size, making it possible to simulate the dynamics of large systems over extended timescales.

Many flavours of MLIPs exist, and with them have arisen a variety of software packages, each typically tailored to training specific architectures (see below). Given their unique specialisations, these individual implementations do not normally conform to a common interface, making it difficult for practitioners to migrate training and validation pipelines between different architectures.

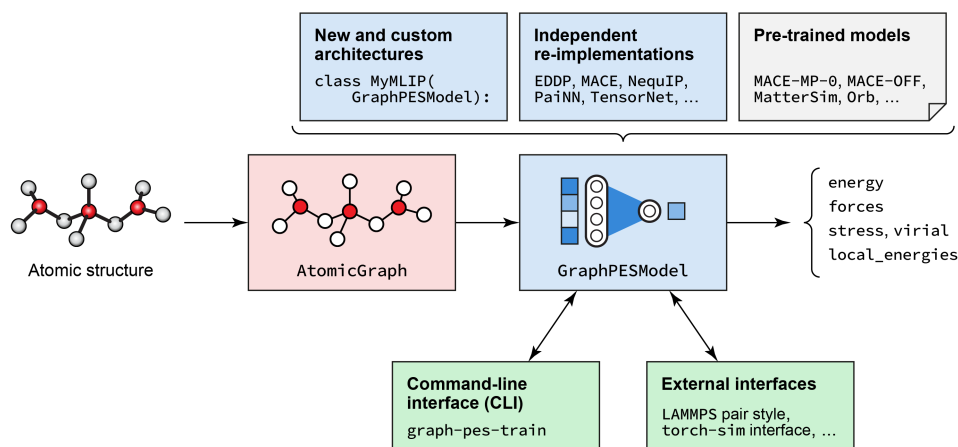


Figure 1: Schematic overview of the functionality of graph-pes. The core components are highlighted in colour. Red: The AtomicGraph class is used to represent atomic structures and incorporates the notion of locality via a neighbour list. Blue: The GraphPESModel class is the general base class for all graph-pes models. Green: graph-pes includes a CLI for easy training, and interfaces to multiple external simulation tools for evaluating MLIPs.

The graph-pes package provides a **unified interface and framework** for defining, training, and working with graph-based MLIP models, complementing existing software in the field (see “Related work” section below). This reduces the barrier to entry for researchers wanting to implement new MLIP architectures, and allows practitioners to easily explore different MLIP architectures: training scripts require as little as one line of code to swap between model architectures, while validation scripts can be written in an architecture-agnostic manner, with LAMMPS input scripts, ASE calculators, and torch-sim simulations requiring no changes other than pointing to a different model file.

Related work

graph-pes is beginning to drive projects within our research group, and we hope that it will be useful to many others. In recent work, we have described the use of graph-pes for fitting NequIP models to datasets created using autoplex (Liu et al., 2025), for assessing zero-shot performance of different graph-network MLIPs (Ben Mahmoud et al., 2025), and for distilling atomistic foundation models (Gardner et al., 2025).

A number of existing packages offer training and validation pipelines for particular ML-PES architectures, including schnetpack (Schütt et al., 2019, 2023), deepmd-kit (Wang et al., 2018; Zeng et al., 2023), nequip (Batzner et al., 2022), mace-torch (Batatia et al., 2022), torchmd-net (Pelaez et al., 2024), and fairchem (Shuaibi et al., 2025). These frameworks focus on their associated model families and do not share a common interface for training. While MatterTune (Kong et al., 2025) offers a unified interface for foundation model fine-tuning, it does not easily support training arbitrary models from scratch. In contrast to these, graph-pes is a general, model-agnostic framework, designed to enable exact side-by-side comparisons, easy implementation of arbitrary new architectures, and standardized training and evaluation workflows.

Features and implementation

Representing atomic structures with graphs

Graphs are a natural way to represent atomic structure, with nodes representing atoms and edges defining a local neighbourhood for each atom. The `AtomicGraph` class therefore serves as the base data structure in `graph-pes`, storing atomic positions, chemical identities, unit cell vectors (where applicable), and an edge list. Writing performant code for graph operations can be challenging; `graph-pes` therefore provides optimised implementations for accessing derived properties and performing common operations on both single and batched graph instances. This simplifies the implementation of new MLIP models and ensures forward passes remain readable. Full API details are available in the project documentation.

Model implementations

All MLIP models in `graph-pes` are implemented as subclasses of the `GraphPESModel` base class. Implementations need only define a forward pass that returns a local energy for each atom or a total energy for the structure; the framework handles the calculation of forces and stress tensors in a conservative manner via automatic differentiation. We also support models that return direct force and stress tensor predictions (e.g., `TensorNet` or `orb-v3-*` with their optional direct force readout heads).

Building on the `GraphPESModel` class, we provide independent re-implementations of popular MLIP architectures, including `PaiNN` (Schütt et al., 2021), `EDDP` (Pickard, 2022), `NequIP` (Batzner et al., 2022), `MACE` (Batatia et al., 2022), and `TensorNet` (Simeon & de Fabritiis, 2023). We use building blocks provided by `e3nn` (Geiger & Smidt, 2022) to implement models that act on spherical tensor decompositions.

Furthermore, we provide an `AdditionModel` implementation, which makes predictions as a sum over independent models. This allows `graph-pes` to add offset energies (`EnergyOffset`) and pair-repulsion terms (`LennardJones`, `Morse`, and `ZBLCoreRepulsion`) to any model implementation, as well as the creation and use of model ensembles.

Training and validation

We provide the `graph-pes-train` CLI tool for training any `GraphPESModel` on datasets of labelled atomic structures. As well as training from scratch, we also support the fine-tuning of existing models on new datasets, facilitating a variety of strategies, including synthetic pre-training (Gardner et al., 2024), foundation model fine-tuning (see below), and frozen transfer learning (Radova et al., 2025).

Under the hood, `graph-pes-train` builds upon the `PyTorch Lightning` (Falcon & The PyTorch Lightning team, 2019) training loop, allowing the user to configure a variety of common training features and callbacks. We also support the use of arbitrary, user-defined components, including custom loss functions, model architectures, optimisers, and datasets.

Because all models conform to the same interface, all training features can be used with any model architecture. Similarly, all downstream model uses can be written in an architecture-agnostic manner, allowing for MD, relaxations, and other scripts to be written once, and then used with any MLIP architecture, e.g. for extended validation beyond simple error metrics (Morrow et al., 2023).

Easy access to foundation models

A recent area of research is the development of “foundational” MLIPs that can describe the potential-energy surface of a wide range of systems. `graph-pes` integrates directly with the `mace-torch`, `mattersim`, and `orb-models` packages to provide access to, among others,

the MACE-MP (Batatia et al., 2025), MatterSim (Yang et al., 2024), orb-v2 (Neumann et al., 2024), MACE-OFF (Kovács et al., 2025), Egret-v1 (Mann et al., 2025), and orb-v3 (Rhodes et al., 2025) families of models. Each of these integrations generates GraphPESModels that are directly compatible with all relevant graph-pes features.

Acknowledgements

We thank Zoé Faure Beaulieu, Krystian Gierczak, and Daniel Thomas du Toit for early testing and feedback. J.L.A.G. acknowledges a UKRI Linacre - The EPA Cephalosporin Scholarship, support from an EPSRC DTP award [grant number EP/T517811/1], and from the Department of Chemistry, University of Oxford. This work was supported by UK Research and Innovation [grant number EP/X016188/1].

References

- Batatia, I., Benner, P., Chiang, Y., Elena, A. M., Kovács, D. P., Riebesell, J., Advincula, X. R., Asta, M., Avaylon, M., Baldwin, W. J., Berger, F., Bernstein, N., Bhowmik, A., Bigi, F., Blau, S. M., Cărare, V., Ceriotti, M., Chong, S., Darby, J. P., ... Csányi, G. (2025). A foundation model for atomistic materials chemistry. *The Journal of Chemical Physics*, 163(18), 184110. <https://doi.org/10.1063/5.0297006>
- Batatia, I., Kovacs, D. P., Simm, G. N. C., Ortner, C., & Csanyi, G. (2022, October). MACE: Higher Order Equivariant Message Passing Neural Networks for Fast and Accurate Force Fields. *Advances in Neural Information Processing Systems*.
- Batzner, S., Musaelian, A., Sun, L., Geiger, M., Mailoa, J. P., Kornbluth, M., Molinari, N., Smidt, T. E., & Kozinsky, B. (2022). E(3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials. *Nature Communications*, 13(1), 2453. <https://doi.org/10.1038/s41467-022-29939-5>
- Ben Mahmoud, C., El-Machachi, Z., Gierczak, K. A., Gardner, J. L. A., & Deringer, V. L. (2025). *Assessing zero-shot generalisation behaviour in graph-neural-network interatomic potentials* (No. arXiv:2502.21317). <https://doi.org/10.48550/arXiv.2502.21317>
- Cohen, O., Riebesell, J., Goodall, R., Kolluru, A., Falletta, S., Krause, J., Colindres, J., Ceder, G., & Gangan, A. S. (2025). TorchSim: An efficient atomistic simulation engine in PyTorch. *AI for Science*, 1(2), 025003. <https://doi.org/10.1088/3050-287X/ae1799>
- Deringer, V. L., Caro, M. A., & Csányi, G. (2019). Machine Learning Interatomic Potentials as Emerging Tools for Materials Science. *Advanced Materials*, 31(46), 1902765. <https://doi.org/10.1002/adma.201902765>
- Falcon, W., & The PyTorch Lightning team. (2019). *PyTorch Lightning* (Version 1.4). <https://doi.org/10.5281/zenodo.3828935>
- Gardner, J. L. A., Baker, K. T., & Deringer, V. L. (2024). Synthetic pre-training for neural-network interatomic potentials. *Machine Learning: Science and Technology*, 5(1), 015003. <https://doi.org/10.1088/2632-2153/ad1626>
- Gardner, J. L. A., Toit, D. F. T. du, Ben Mahmoud, C., Beaulieu, Z. F., Juraskova, V., Paşca, L.-B., Rosset, L. A. M., Duarte, F., Martelli, F., Pickard, C. J., & Deringer, V. L. (2025). *Distillation of atomistic foundation models across architectures and chemical domains* (No. arXiv:2506.10956). <https://doi.org/10.48550/arXiv.2506.10956>
- Geiger, M., & Smidt, T. (2022). *E3nn: Euclidean Neural Networks* (No. arXiv:2207.09453). <https://doi.org/10.48550/arXiv.2207.09453>
- Kong, L., Shoghi, N., Hu, G., Li, P., & Fung, V. (2025). *MatterTune: An Integrated, User-*

- Friendly Platform for Fine-Tuning Atomistic Foundation Models to Accelerate Materials Simulation and Discovery* (No. arXiv:2504.10655). <https://doi.org/10.48550/arXiv.2504.10655>
- Kovács, D. P., Moore, J. H., Browning, N. J., Batatia, I., Horton, J. T., Pu, Y., Kapil, V., Witt, W. C., Magdău, I.-B., Cole, D. J., & Csányi, G. (2025). *MACE-OFF: Transferable Short Range Machine Learning Force Fields for Organic Molecules* (No. arXiv:2312.15211). <https://doi.org/10.48550/arXiv.2312.15211>
- Larsen, A. H., Mortensen, J. J., Blomqvist, J., Castelli, I. E., Christensen, R., Duřak, M., Friis, J., Groves, M. N., Hammer, B., Hargus, C., Hermes, E. D., Jennings, P. C., Jensen, P. B., Kermode, J., Kitchin, J. R., Kolsbjerg, E. L., Kubal, J., Kaasbjerg, K., Lysgaard, S., ... Jacobsen, K. W. (2017). The atomic simulation environment—a Python library for working with atoms. *Journal of Physics: Condensed Matter*, 29(27), 273002. <https://doi.org/10.1088/1361-648X/aa680e>
- Liu, Y., Morrow, J. D., Ertural, C., Fragapane, N. L., Gardner, J. L. A., Naik, A. A., Zhou, Y., George, J., & Deringer, V. L. (2025). An automated framework for exploring and learning potential-energy surfaces. *Nature Communications*, 16(1), 7666. <https://doi.org/10.1038/s41467-025-62510-6>
- Mann, E. L., Wagen, C. C., Vandezande, J. E., Wagen, A. M., & Schneider, S. C. (2025). *Egret-1: Pretrained Neural Network Potentials For Efficient and Accurate Bioorganic Simulation* (No. arXiv:2504.20955). <https://doi.org/10.48550/arXiv.2504.20955>
- Morrow, J. D., Gardner, J. L. A., & Deringer, V. L. (2023). How to validate machine-learned interatomic potentials. *The Journal of Chemical Physics*, 158(12), 121501. <https://doi.org/10.1063/5.0139611>
- Neumann, M., Gin, J., Rhodes, B., Bennett, S., Li, Z., Choubisa, H., Hussey, A., & Godwin, J. (2024). *Orb: A Fast, Scalable Neural Network Potential* (No. arXiv:2410.22570). <https://doi.org/10.48550/arXiv.2410.22570>
- Pelaez, R. P., Simeon, G., Galvelis, R., Mirarchi, A., Eastman, P., Doerr, S., Thölke, P., Markland, T. E., & De Fabritiis, G. (2024). TorchMD-Net 2.0: Fast Neural Network Potentials for Molecular Simulations. *Journal of Chemical Theory and Computation*, 20(10), 4076–4087. <https://doi.org/10.1021/acs.jctc.4c00253>
- Pickard, C. J. (2022). Ephemeral data derived potentials for random structure search. *Physical Review B*, 106(1), 014102. <https://doi.org/10.1103/PhysRevB.106.014102>
- Radova, M., Stark, W. G., Allen, C. S., Maurer, R. J., & Bartók, A. P. (2025). *Fine-tuning foundation models of materials interatomic potentials with frozen transfer learning* (No. arXiv:2502.15582). <https://doi.org/10.48550/arXiv.2502.15582>
- Rhodes, B., Vandenhaute, S., Šimkus, V., Gin, J., Godwin, J., Duignan, T., & Neumann, M. (2025). *Orb-v3: Atomistic simulation at scale* (No. arXiv:2504.06231). <https://doi.org/10.48550/arXiv.2504.06231>
- Schütt, K. T., Hessmann, S. S. P., Gebauer, N. W. A., Lederer, J., & Gastegger, M. (2023). SchNetPack 2.0: A neural network toolbox for atomistic machine learning. *The Journal of Chemical Physics*, 158(14), 144801. <https://doi.org/10.1063/5.0138367>
- Schütt, K. T., Kessel, P., Gastegger, M., Nicoli, K. A., Tkatchenko, A., & Müller, K.-R. (2019). SchNetPack: A Deep Learning Toolbox For Atomistic Systems. *Journal of Chemical Theory and Computation*, 15(1), 448–455. <https://doi.org/10.1021/acs.jctc.8b00908>
- Schütt, K. T., Unke, O. T., & Gastegger, M. (2021). *Equivariant message passing for the prediction of tensorial properties and molecular spectra* (No. arXiv:2102.03150). <https://doi.org/10.48550/arXiv.2102.03150>
- Shuaibi, M., Das, A., Sriram, A., Misko, Barroso-Luque, L., Gao, R., Goyal, S., Ulissi, Z.,

- Wood, B., Xie, T., Yoon, J., Wander, B., Kolluru, A., Barnes, R., Sunshine, E., Tran, K., Xiang, Levine, D., Shoghi, N., ... Michel, K. (2025). *FAIRChem* (Version 2.2.0). <https://doi.org/10.5281/zenodo.15587498>
- Simeon, G., & de Fabritiis, G. (2023). *TensorNet: Cartesian Tensor Representations for Efficient Learning of Molecular Potentials* (No. arXiv:2306.06482). <https://arxiv.org/abs/2306.06482>
- Thompson, A. P., Aktulga, H. M., Berger, R., Bolintineanu, D. S., Brown, W. M., Crozier, P. S., in 't Veld, P. J., Kohlmeyer, A., Moore, S. G., Nguyen, T. D., Shan, R., Stevens, M. J., Tranchida, J., Trott, C., & Plimpton, S. J. (2022). LAMMPS - a flexible simulation tool for particle-based materials modeling at the atomic, meso, and continuum scales. *Computer Physics Communications*, 271, 108171. <https://doi.org/10.1016/j.cpc.2021.108171>
- Wang, H., Zhang, L., Han, J., & E, W. (2018). DeePMD-kit: A deep learning package for many-body potential energy representation and molecular dynamics. *Computer Physics Communications*, 228, 178–184. <https://doi.org/10.1016/j.cpc.2018.03.016>
- Yang, H., Hu, C., Zhou, Y., Liu, X., Shi, Y., Li, J., Li, G., Chen, Z., Chen, S., Zeni, C., Horton, M., Pinsler, R., Fowler, A., Zügner, D., Xie, T., Smith, J., Sun, L., Wang, Q., Kong, L., ... Lu, Z. (2024). *MatterSim: A Deep Learning Atomistic Model Across Elements, Temperatures and Pressures* (No. arXiv:2405.04967). <https://doi.org/10.48550/arXiv.2405.04967>
- Zeng, J., Zhang, D., Lu, D., Mo, P., Li, Z., Chen, Y., Rynik, M., Huang, L., Li, Z., Shi, S., Wang, Y., Ye, H., Tuo, P., Yang, J., Ding, Y., Li, Y., Tisi, D., Zeng, Q., Bao, H., ... Wang, H. (2023). DeePMD-kit v2: A software package for deep potential models. *The Journal of Chemical Physics*, 159(5), 054801. <https://doi.org/10.1063/5.0155600>