Support for advanced transition state search techniques in CASTEP

ARCHER2-eCSE02-4 Technical Report

Simone Sturniolo[1], Phil Hasnip[2], Peter Byrne[2], Paul Hodgkinson[3], Matt Probert[2] and James Kermode[4]

[1] Scientific Computing Department, Science & Technology Facilities Council, Rutherford Appleton Laboratory, Didcot OX11 0QX. United Kingdom
[2] Department of Physics, University of York, York YO10 5DD, United Kingdom
[3] Department of Chemistry, Durham University, Durham, DH1 3LE, UK
[4] Warwick Centre for Predictive Modelling, School of Engineering, University of Warwick, Coventry, CV4 7AL, United Kingdom

Abstract

The CASTEP density functional theory code is a UK flagship code, specialised for solid materials, and is heavily used on ARCHER2 (300-400 active users). While support for obtaining the ground-state electronic and atomic configurations is now very good, before this eCSE was completed computing transition states, reaction rates, and exploring free energy barriers with enhanced sampling were all still poorly supported in the CASTEP code, despite their importance for a wide range of chemistry and materials science applications. We have implemented two key new features in the CASTEP code to address these issues: (i) support for the extremely widely used nudged elastic band (NEB) transition state search tool, augmented by a state-of-the-art robust optimizer; (ii) an interface to the i-PI universal force engine which allows CASTEP to be connected efficiently to a wide range of external codes with enhanced sampling capabilities. Key beneficients include the CCP-NC, CCP9 and UKCP communities, where the new tools will aid in reconciling experimental observations with atomic-scale behaviour, helping to guide and interpret future experiments.

Introduction

Rare events, such as defect migration, are very hard to model directly with first principles methods because of the unfeasibly long simulation times that would be required to observe them in dynamical simulations. The nudged elastic band (NEB) method is very widely used to predict transition states and reaction rates from first principles calculations, with the key paper attracting over 1000 citations per year [1]. The algorithm works by relaxing a discretized chain of states joining reactants to products in order to produce an estimate for the saddle point or transition state and the associated minimum energy path.

Direct implementations of NEB were previously available in other DFT codes, but before this eCSE project, CASTEP's transition search capabilities were limited to the low accuracy linear/quadratic synchronous transit (LST/QST) method.

Serial implementation of the nudged elastic band algorithm

Implementation details

We first implemented the standard NEB algorithm of Refs. [1,2] directly within the CASTEP Fortran code. In our NEB implementation the update rule for atomic position is given by

$$x_n^{k+1} = x_n^k + \alpha^k \left[-\nabla^{\perp} V(x_n^k) + \boldsymbol{\eta}_n^k \right]$$

where x_n^k are the positions of image n at iteration k, α^k is the step size, which can either be fixed or chosen in an adaptive manner, $\nabla^{\perp} V(x_n^k)$ is the gradient of the total energy, projected to remove the component along the reaction path, and $\boldsymbol{\eta}_n^k = \boldsymbol{\eta}((x_n^k)', (x_n^k)'')$ is the NEB spring force.

As a consequence of the projection step, the NEB forces do not have a corresponding energy functional and hence linesearch is not possible, making quasi-Newton searches such as (L)BFGS inapplicable. Standard approaches are either to use simple gradient-only based methods or to add a maximum step size constraint in place of the linesearch. Both of these can exhibit slow convergence; a solution is instead to choose the step size adaptively using a custom time-stepping algorithm. A promising solution to this problem is the ODE12r adaptive scheme introduced in Ref. [3], which combines two distinct step selection criteria: one based on minimising the change in the residual force from one step to the next as is typically done to control error in adaptive ODE solvers, and a second based on minimising the residual itself. The key idea is that adaptive ODE step selection should be used in the pre-asymptotic regime, while minimising the residual is suitable in the asymptotic regime.

We therefore implemented three optimisation schemes for NEB paths, with the approach to be used controlled by the `TSSEARCH_NEB_METHOD` parameter: (i) the two point steepest descent scheme of Barzilai and Borwein [4]; (ii) the fast inertial relaxation engine (FIRE) [5] and (iii) the ODE12r scheme discussed above. In all three cases convergence is controlled by the previously existing `TSSEARCH_FORCE_TOL` parameter; a minor modification was made to set the default value of this parameter equal to the force tolerance used during geometry optimisation, `GEOM_FORCE_TOL`, to better fit user intuition.

We also provide three alternative algorithms to compute the tangents $(x_n^k)'$ and curvatures $(x_n^k)''$ at each image along the path, controlled by the `TSSEARCH_NEB_TANGENT_MODE` parameter. The three approaches are based on bisection, the improved tangent scheme of Ref [2], and cubic spline interpolation as used in Ref [3]. The latter was found to be the most robust so has been made the default.

Modifications of CASTEP, which have recently been incorporated into the main development repository[1] mainly consisted of changes to the following source files:

- `Functional/api.f90` - usage of Peter Byrne's common high-level API for CASTEP
- `Functional/tssearch_neb.f90` - new source file with implementation of NEB
- `Functional/tssearch_utils.f90` - addition of NEB to transition state tools

[1] https://bitbucket.org/castep/castep/pull-requests/217/ecse-project-neb-and-i-pi-socket

- `Fundamental/parameters.f90` - new user parameters, described below
- `Test/algor_test.f90` - continuous integration tests for spline routines
- `Test/tssearch_test.f90` - continuous integration tests for NEB functionality
- `Utilities/readts` - new Python library for parsing transition state outputs

Usage and parameters

Since CASTEP previously included support for some transition state search algorithms we did not need to make extensive changes to the user-controlled parameters to give access to our new functionality. The table below summarises all parameters related to NEB; this information is available in the CASTEP online help[2] and will in future be provided on the documentation website.

Parameter	Default	Level	Notes
TASK	SINGLEPOINT	Basic	Task should be set to TRANSITIONSTATESEARCH to perform a transition state calculation.
TSSEARCH_METHOD	LSTQST	Basic	The search method used to locate transition states. Must be set to NEB to perform a NEB calculation. Previous default retained for backwards compatibility. Modifiable: restart only Allowed values: LSTQST, NEB Default value : LSTQST
TSSEARCH_FORCE_TOL	Same as GEOM_FORCE_TOL	Basic	Tolerance for accepting convergence of the maximum |ionic force| during QST search. Modifiable: restart and on the fly
TSSEARCH_MAX_PATH_POINTS	20	Intermediate	The maximum number of path points for NEB search. Modifiable: restart and on the fly Allowed values: (any integer) > 0
TSSEARCH_NEB_METHOD	ODE12R	Intermediate	Method used to optimize the NEB shape. Modifiable: restart and on the fly Allowed values: GRAD_BB, FIRE, ODE12R
TSSEARCH_NEB_TANGENT_MODE	SPLINE	Basic	Method used to calculate the tangents of the NEB.

[2] e.g. by running the command `castep -help search tssearch`

			Modifiable: restart and on the fly Allowed values: NONE, BISECT, HIGH_E, SPLINE
TSSEARCH_NEB_SPRING_CONSTANT	0.1 eV/ang^2	Basic	Spring constant used between the images in NEB search. Modifiable: restart and on the fly Allowed values: (any) > 0.0
TSSEARCH_NEB_CLIMBING	FALSE	Basic	If TRUE then the central bead in NEB search climbs up the potential and TSSEARCH_MAX_PATH_POINTS must be odd (may be increased by +1). If FALSE then the central bead in NEB search slides down the potential. Modifiable: restart only Allowed values: TRUE or FALSE Default value: FALSE
TSSEARCH_NEB_MAX_ITER	20	Intermediate	The maximum number of steps during NEB search. Modifiable: restart and on the fly Allowed values: (any integer) > 0
TSSEARCH_NEB_NORMED	FALSE if TSSEARCH_NEB_TANGENT_MODE = SPLINE, TRUE otherwise	Expert	If TRUE then the spring forces applied along the tangents in the NEB are normed to the displacement between beads. If FALSE then the spring forces are projected along the tangents in the NEB and may result in null forces if displacements are orthogonal to the NEB tangents. Modifiable: restart only Allowed values: TRUE or FALSE

Verification and validation

The new serial NEB implementation was carefully verified with respect to a robust Python implementation within the Atomic Simulation Environment (ASE) [6] previously written by the PI of this eCSE project, James Kermode. Since we used an identical algorithmic approach, it was possible to exactly match the converged minimum energy path produced by the Python code for a test case of diffusion of a vacancy in an FCC lattice using the Morse interatomic potential, which is available both within CASTEP's pair-potential library and in ASE, giving confidence in our new implementation. Specimen input files are provided in Appendix A and available online with the supplementary information to this report.

We next verified that the implementation runs correctly when using DFT forces. We obtained accurate minimum energy paths for vacancy diffusion in copper and a gold atom diffusing on an aluminium surface, as illustrated in Figures 1 and 2, respectively. Validation was performed with respect to the existing ASE CASTEP interface, which is based on generating input files and parsing output files; this could not be used efficiently with complex workflows such as NEB, since reuse of electronic state from one step to the next is not possible without substantial I/O.

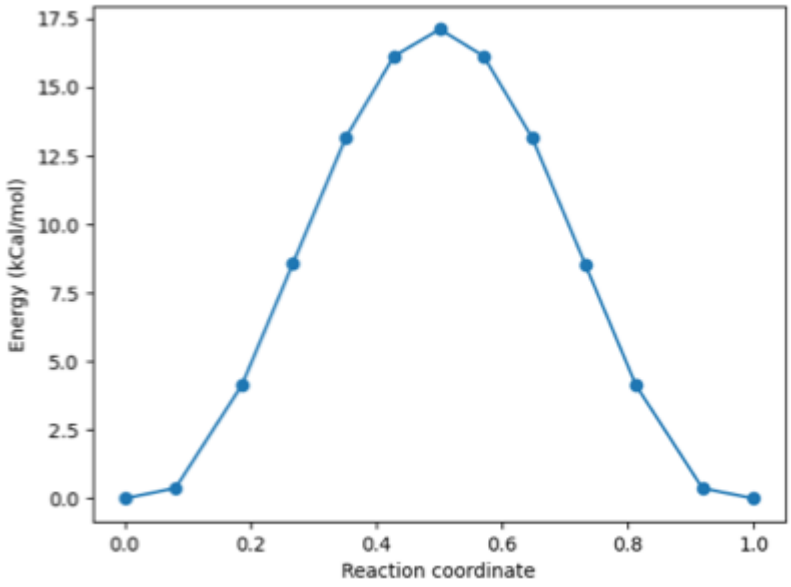

Figure 1. Converged NEB minimum energy path for vacancy diffusion in bulk copper. The energy barrier of around 0.025 eV is in good agreement with results obtained when driving CASTEP from ASE.

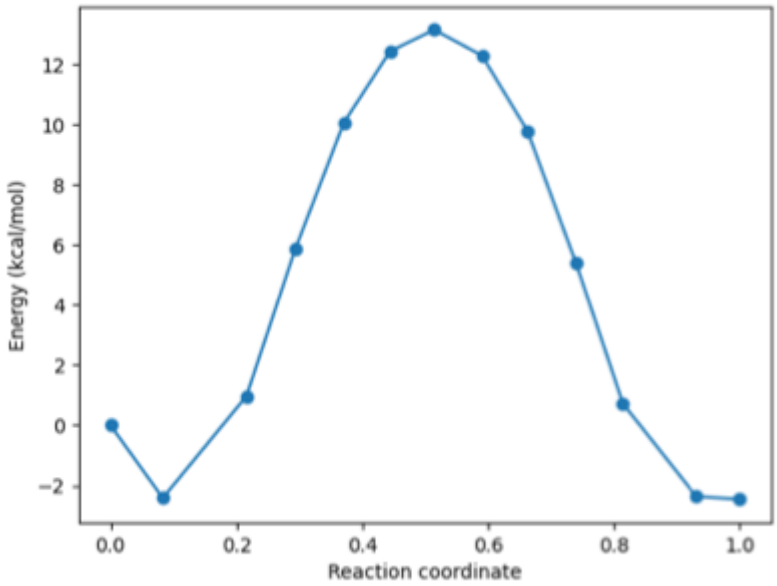

Figure 2. Converged NEB minimum energy path for a gold adatom diffusing on an aluminium surface. The kink close to a reaction coordinate value of 0.1 indicates the initial geometry optimisation should be slightly more tightly converged, since the image adjacent to it has moved downhill in energy; however, there are sufficient images to give a good MEP profile.

Parallelisation over NEB images

We next added top-level parallelisation over NEB images, modelled on the task-farm parallelisation of the existing path integral molecular dynamics (PIMD) over beads, a closely related technique. The number of farms to use is controlled by the existing NUM_FARMS parameter; typically this should be set to the number of NEB images.

We tested our implementation on a single 128 core node on ARCHER2. This leads to ideal weak scaling, as we validated on the 16 image system shown in Figure 3, where we used 1, 2, 4, 8 and 16 farms respectively, with each farm having 8 cores. (We in fact see a small speed up in the time for a single NEB step as the number of farms is increased; we attribute this to on average better initial guesses for the electronic ground state when there are more images and thus they are more closely spaced along the NEB path.)

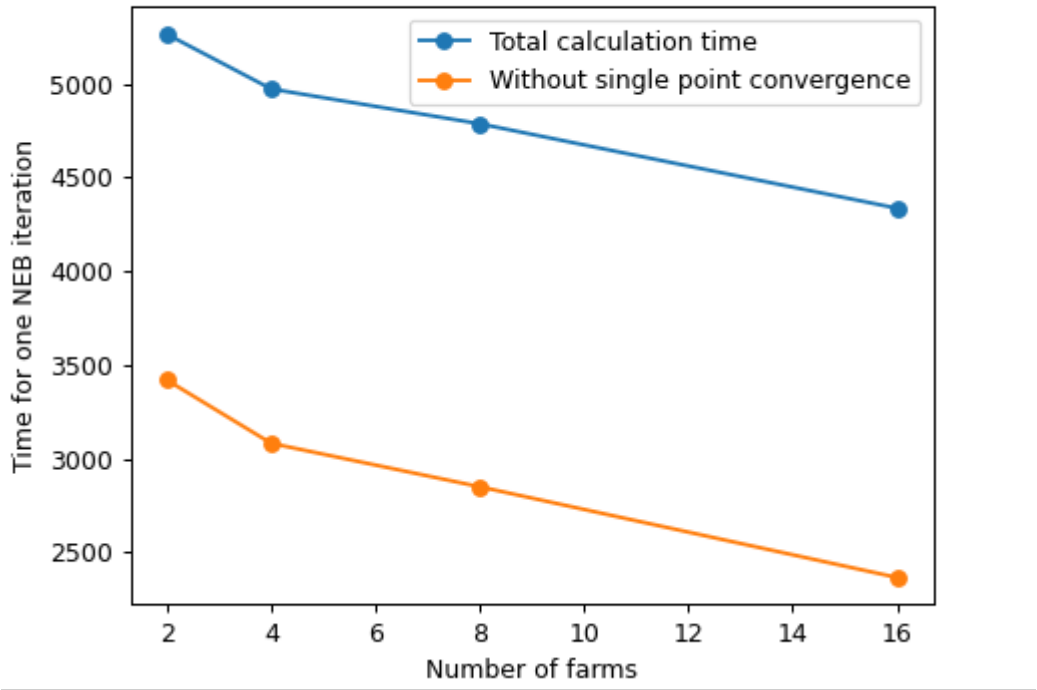

Figure 3. Weak scaling performance of the parallel NEB implementation. Time to solution for a single NEB update step is shown as a function of the number of compute farms present. The total time to solution actually falls as the number of farms increases due to improved initial guesses for the electronic minimisation.

Socket-based interface to the i-PI package

We next expanded the scope of the project to unlock the power of advanced optimisation and sampling algorithms more generally, by decoupling the problem of evolving the atomic positions from that of computing the interatomic forces. This challenge has already been addressed in a number of other electronic structure codes using the i-PI interface [7], which was already supported by CP2K, DFTB+, Lammps, Quantum ESPRESSO, Siesta, FHI-aims, Yaff, deMonNano, and TBE.

Notably the widely used plumed and ASE frameworks can also act as i-PI servers, adding support for advanced sampling and optimisation, respectively, with no further development

work. The approach is illustrated in Figure 4, where a number of independent CASTEP engines running on different MPI partitions act as clients of a central server which manages communication using POSIX sockets. The high-level algorithm can be implemented as part of the server in a code-independent manner. This work will be linked to a cross-code Python API initiative being developed as part of the ExCALIBUR *Materials and Molecular Modelling Exascale Design and Development Working Group* through PJH and PB.

A pre-existing prototype socket-based communication tool which hooks into the existing molecular dynamics routines within CASTEP was implemented by the PI, along with a asynchronous Python server. In this project we developed an alternative socket communications interface which uses the i-PI protocol, and allows CASTEP to be used as a drop-in force engine for a wide range of tasks.

Modifications of CASTEP consisted of changes to the following source files:

- `Functional/api.f90` - usage of Peter Byrne's common high-level API for CASTEP
- `Functional/socketdriver.f90` - high-level socket driver routines
- `Utility/csockets.c` and `Utility/f90sockets.f90` - C and Fortran bindings

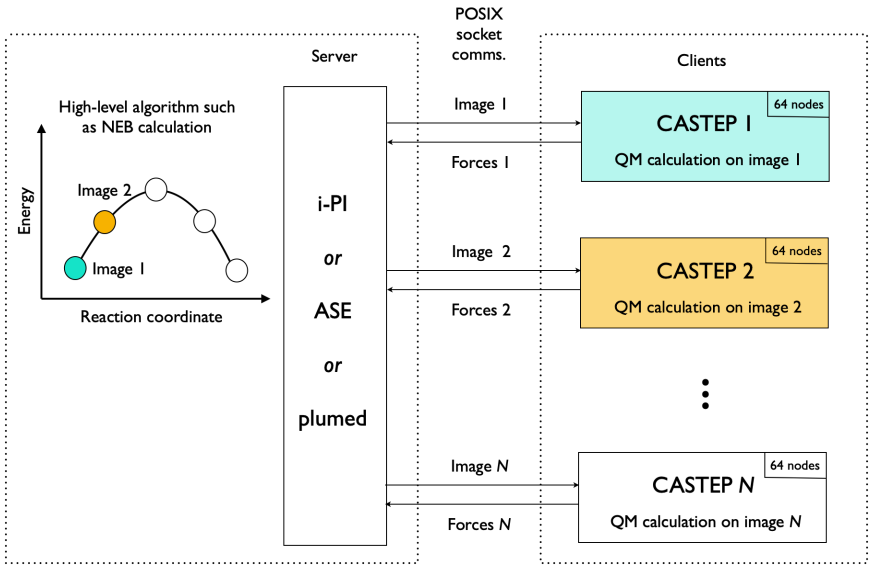

Figure 4: Schematic of the client/server configuration for socket-based communications. Each image runs within an MPI CASTEP run, linked by the i-PI socket protocol.

The i-PI interface has been implemented within the CASTEP code and allows efficient computational steering from Python codes. We also used it to validate the internal NEB implementation of objectives 1 and 2 with respect to the existing Python NEB implementation in the Atomic Simulation Environment (ASE), cf. Figures 1 and 2 above.

Usage is fairly straightforward, with two new parameters and a new option for the TASK parameter as detailed in the label below. Example input files for the socket interface can be found in Appendix B.

Parameter	**Default value**	**Notes**
TASK	SINGLEPOINT	Can now be set to SOCKET_DRIVER to enable socket communications mode
SOCKET_HOST	localhost	Address or IP address of the host to which the socket connection must be established. Modifiable: restart or on the fly Allowed values: up to 255 character address
SOCKET_PORT	3141	Port on which the socket connection must be established. Modifiable: restart or on the fly Allowed values: 0 < SOCKET_PORT < 65536

Extension objective on preconditioned minimisation

We originally hoped to deliver an extension objective of including a Fortran implementation of the efficient implementation of preconditioned minimum energy path finding of Ref. [3] or by adding a Fortran implementation building on components previously implemented in eCSE11-07.

Insufficient time was available to complete this extension objective directly within the Fortran codebase. However, we have demonstrated that our alternative approach of linking CASTEP to an existing Python implementation within the ASE package can now be done in two ways: (i) using our i-PI interface described above; (ii) the high-level Python cross-code API developed by project co-I Peter Byrne within the ExCALIBUR project. Since the latter project ran concurrently with this eCSE, we cooperated closely to use the same Fortran routines in our NEB and iPI implementations to avoid redundant code paths. We have validated the two approaches against one another and against our internal NEB implementation for the case of vacancy migration in copper (Figure 1).

Outlook

The code implementing our project objectives has been made available in the main CASTEP codebase, documented and supported through user tutorials and is now in production use in our research groups. Our pull request has been merged into the main CASTEP repository on BitBucket, which means the functionality will be incorporated in the 2022 release of CASTEP and made available on ARCHER2. We anticipate that these enhancements will enable computational steering of CASTEP by high-level algorithms in future projects. The PI is in the process of switching over his QM/MM projects, which currently rely on a combination of the previous proof-of-principle socket code and the less-efficient ASE file-based interface to use the new functionality.

Data Availability

The performance testing data supporting this report are available from Ref. [8]

Appendix A. Example NEB Calculation

Here we provide specimen input files for an example NEB calculation for vacancy migration in copper, equivalent to that performed in the validation tests reported above. These data are included with the supporting data for this report, along with Python scripts used to generate them

`CuVacNEB.cell` - CASTEP cell file, containing reactant, product and intermediate states

```
%BLOCK LATTICE_CART
 7.253359  0.000000  0.000000
 0.000000  7.253359  0.000000
 0.000000  0.000000  7.253359
%ENDBLOCK LATTICE_CART

%BLOCK POSITIONS_ABS
Cu -0.000016  0.000010  0.000001
Cu  7.253350  1.819280  1.819176
Cu  1.819274  7.253336  1.819210
Cu  1.819246  1.819258 -0.000004
Cu  0.000015 -0.000021  3.626677
Cu  7.253352  1.819280  5.434184
Cu  1.819272 -0.000017  5.434148
Cu  1.834521  1.834523  3.626682
Cu  7.253350  3.626673 -0.000001
Cu  7.253385  5.434078  1.819210
Cu  1.834487  3.626703  1.834479
Cu  1.819238  5.434103  0.000007
Cu  7.253378  3.626707  3.626680
Cu  7.253381  5.434078  5.434153
Cu  1.834487  3.626697  5.418876
Cu  1.834514  5.418838  3.626676
Cu  3.626691  0.000018  7.253363
Cu  3.626687  1.834493  1.834456
Cu  5.434085  0.000008  1.819175
Cu  5.434113  1.819250 -0.000001
Cu  3.626658 -0.000011  3.626677
Cu  3.626691  1.834498  5.418897
Cu  5.434087  0.000017  5.434183
Cu  5.418836  1.834512  3.626682
Cu  3.626685  3.626663  0.000003
Cu  3.626656  5.418862  1.834494
Cu  5.418870  3.626672  1.834446
Cu  5.434122  5.434112  0.000007
Cu  3.626658  5.418863  5.418870
Cu  5.418870  3.626662  5.418910
```

```
Cu  5.418844  5.418847  3.626677
%ENDBLOCK POSITIONS_ABS

FIX_COM: FALSE
KPOINT_MP_GRID: 4 4 4
%BLOCK POSITIONS_ABS_PRODUCT
ang
    Cu 0.006 0.006 -0.000
    Cu 0.021 1.813 1.834
    Cu 1.813 0.021 1.834
    Cu 1.813 1.813 7.253
    Cu 0.021 0.021 3.627
    Cu 0.021 1.813 5.419
    Cu 1.813 0.021 5.419
    Cu 3.606 3.606 3.627
    Cu 0.006 3.621 -0.000
    Cu 0.006 5.440 1.819
    Cu 1.813 3.606 1.834
    Cu 1.813 5.440 -0.000
    Cu 0.021 3.606 3.627
    Cu 0.006 5.440 5.434
    Cu 1.813 3.606 5.419
    Cu 1.813 5.440 3.627
    Cu 3.621 0.006 0.000
    Cu 3.606 1.813 1.834
    Cu 5.440 0.006 1.819
    Cu 5.440 1.813 7.253
    Cu 3.606 0.021 3.627
    Cu 3.606 1.813 5.419
    Cu 5.440 0.006 5.434
    Cu 5.440 1.813 3.627
    Cu 3.621 3.621 0.000
    Cu 3.621 5.440 1.819
    Cu 5.440 3.621 1.819
    Cu 5.440 5.440 0.000
    Cu 3.621 5.440 5.434
    Cu 5.440 3.621 5.434
    Cu 5.440 5.440 3.627
%ENDBLOCK POSITIONS_ABS_PRODUCT

%BLOCK POSITIONS_ABS_INTERMEDIATE
ang
    Cu 0.003 0.003 -0.000
    Cu 7.264 1.816 1.827
    Cu 1.816 7.264 1.827
    Cu 1.816 1.816 -0.000
    Cu 0.011 0.011 3.627
    Cu 7.264 1.816 5.427
    Cu 1.816 0.011 5.427
    Cu 2.720 2.720 3.627
    Cu 7.256 3.624 -0.000
    Cu 7.256 5.437 1.819
    Cu 1.824 3.616 1.834
    Cu 1.816 5.437 0.000
    Cu 7.264 3.616 3.627
    Cu 7.256 5.437 5.434
    Cu 1.824 3.616 5.419
    Cu 1.824 5.429 3.627
    Cu 3.624 0.003 7.253
    Cu 3.616 1.824 1.834
    Cu 5.437 0.003 1.819
    Cu 5.437 1.816 0.000
    Cu 3.616 0.011 3.627
    Cu 3.616 1.824 5.419
    Cu 5.437 0.003 5.434
    Cu 5.429 1.824 3.627
    Cu 3.624 3.624 0.000
    Cu 3.624 5.429 1.827
    Cu 5.429 3.624 1.827
    Cu 5.437 5.437 0.000
    Cu 3.624 5.429 5.427
    Cu 5.429 3.624 5.427
```

```
    Cu 5.429 5.429 3.627
%ENDBLOCK POSITIONS_ABS_INTERMEDIATE
```

CuVacNEB.param - CASTEP parameter file

```
TASK: transitionstatesearch
TSSEARCH_METHOD: neb
CUT_OFF_ENERGY: 500.0
WRITE_CELL_STRUCTURE: TRUE
XC_FUNCTIONAL: PBE
MAX_SCF_CYCLES: 100
TSSEARCH_NEB_MAX_ITER: 100
TSSEARRCH_FORCE_TOL: 0.1 eV/ang
TSSEARCH_MAX_PATH_POINTS: 11
TSSEARCH_NEB_CLIMBING: TRUE
```

Appendix B. Example i-PI Calculation

CuVacSocket.cell - CASTEP cell file

```
%BLOCK LATTICE_CART
10.818734  0.000000  0.000000
 0.000000 10.818734  0.000000
 0.000000  0.000000 10.818734
%ENDBLOCK LATTICE_CART

%BLOCK POSITIONS_ABS
Cu -0.005669  0.000000  0.000000
Cu -0.007079  1.796044  1.796044
Cu  1.796684  0.000000  1.794700
Cu  1.796684  1.794700  0.000000
Cu -0.008422  0.000000  3.599807
Cu -0.009155  1.793968  5.409367
Cu  1.788924 -0.000000  5.409367
Cu  1.792694  1.788876  3.595816
Cu -0.008422  0.000000  7.218927
Cu -0.007079  1.796044  9.022690
Cu  1.796684  0.000000  9.024034
Cu  1.792694  1.788876  7.222918
Cu -0.008422  3.599807  0.000000
Cu -0.009155  5.409367  1.793968
Cu  1.792694  3.595816  1.788876
Cu  1.788924  5.409367  0.000000
Cu -0.014247  3.595816  3.595816
Cu -0.039124  5.409367  5.409367
Cu  1.791141  3.594263  5.409367
Cu  1.791141  5.409367  3.594263
Cu -0.014247  3.595816  7.222918
Cu -0.009155  5.409367  9.024766
Cu  1.792694  3.595816  9.029858
Cu  1.791141  5.409367  7.224470
Cu -0.008422  7.218927  0.000000
Cu -0.007079  9.022690  1.796044
Cu  1.792694  7.222918  1.788876
Cu  1.796684  9.024034  0.000000
Cu -0.014247  7.222918  3.595816
Cu -0.009155  9.024766  5.409367
Cu  1.791141  7.224470  5.409367
Cu  1.792694  9.029858  3.595816
Cu -0.014247  7.222918  7.222918
Cu -0.007079  9.022690  9.022690
Cu  1.792694  7.222918  9.029858
Cu  1.792694  9.029858  7.222918
Cu  3.606245  0.000000  0.000000
```

```
Cu  3.606245  1.793968  1.793968
Cu  5.415805  0.000000  1.794700
Cu  5.415805  1.794700  0.000000
Cu  3.606245 -0.000000  3.592046
Cu  3.606245  1.763999  5.409367
Cu  5.423565 -0.000000  5.409367
Cu  5.419795  1.788876  3.595816
Cu  3.606245 -0.000000  7.226688
Cu  3.606245  1.793968  9.024766
Cu  5.415805 -0.000000  9.024034
Cu  5.419795  1.788876  7.222918
Cu  3.606245  3.592046  0.000000
Cu  3.606245  5.409367  1.763999
Cu  5.419795  3.595816  1.788876
Cu  5.423565  5.409367 -0.000000
Cu  3.606245  3.594263  3.594263
Cu  5.421348  3.594263  5.409367
Cu  5.421348  5.409367  3.594263
Cu  3.606245  3.594263  7.224470
Cu  3.606245  5.409367  9.054735
Cu  5.419795  3.595816  9.029858
Cu  5.421348  5.409367  7.224470
Cu  3.606245  7.226688 -0.000000
Cu  3.606245  9.024766  1.793968
Cu  5.419795  7.222918  1.788876
Cu  5.415805  9.024034  0.000000
Cu  3.606245  7.224470  3.594263
Cu  3.606245  9.054735  5.409367
Cu  5.421348  7.224470  5.409367
Cu  5.419795  9.029858  3.595816
Cu  3.606245  7.224470  7.224470
Cu  3.606245  9.024766  9.024766
Cu  5.419795  7.222918  9.029858
Cu  5.419795  9.029858  7.222918
Cu  7.218159  0.000000  0.000000
Cu  7.219568  1.796044  1.796044
Cu  9.015611  0.000000  1.797453
Cu  9.015611  1.797453  0.000000
Cu  7.220911 -0.000000  3.599807
Cu  7.221644  1.793968  5.409367
Cu  9.015611  0.000000  5.409367
Cu  9.015611  1.794700  3.599807
Cu  7.220911 -0.000000  7.218927
Cu  7.219568  1.796044  9.022690
Cu  9.015611  0.000000  9.021281
Cu  9.015611  1.794700  7.218927
Cu  7.220911  3.599807 -0.000000
Cu  7.221644  5.409367  1.793968
Cu  9.015611  3.599807  1.794700
Cu  9.015611  5.409367  0.000000
Cu  7.226736  3.595816  3.595816
Cu  7.251613  5.409367  5.409367
Cu  9.015611  3.592046  5.409367
Cu  9.015611  5.409367  3.592046
Cu  7.226736  3.595816  7.222918
Cu  7.221644  5.409367  9.024766
Cu  9.015611  3.599807  9.024034
Cu  9.015611  5.409367  7.226688
Cu  7.220911  7.218927  0.000000
Cu  7.219568  9.022690  1.796044
Cu  9.015611  7.218927  1.794700
Cu  9.015611  9.021281  0.000000
Cu  7.226736  7.222918  3.595816
Cu  7.221644  9.024766  5.409367
Cu  9.015611  7.226688  5.409367
Cu  9.015611  9.024034  3.599807
Cu  7.226736  7.222918  7.222918
Cu  7.219568  9.022690  9.022690
Cu  9.015611  7.218927  9.024034
Cu  9.015611  9.024034  7.218927
%ENDBLOCK POSITIONS_ABS
```

CuVacSocket.param - CASTEP parameter file

```
TASK: socketdriver
CUT_OFF_ENERGY: 500.0
XC_FUNCTIONAL: PBE
```

References

[1] G. Henkelman, B. P. Uberuaga, and H. Jonsson, *A Climbing Image Nudged Elastic Band Method for Finding Saddle Points and Minimum Energy Paths*, J. Chem. Phys. **113**, 9901 (2000).

[2] G. Henkelman and H. Jónsson, *Improved Tangent Estimate in the Nudged Elastic Band Method for Finding Minimum Energy Paths and Saddle Points*, J. Chem. Phys. **113**, 9978 (2000).

[3] S. Makri, C. Ortner, and J. R. Kermode, *A Preconditioning Scheme for Minimum Energy Path Finding Methods*, J. Chem. Phys. **150**, 094109 (2019).

[4] J. Barzilai and J. M. Borwein, *Two-Point Step Size Gradient Methods*, IMA J. Numer. Anal. **8**, 141 (1988).

[5] E. Bitzek, P. Koskinen, F. Gahler, M. Moseler, and P. Gumbsch, *Structural Relaxation Made Simple*, Phys. Rev. Lett.

[6] A. H. Larsen, J. J. Mortensen, J. Blomqvist, I. E. Castelli, R. Christensen, M. Dułak, J. Friis, M. N. Groves, B. Hammer, C. Hargus, E. D. Hermes, P. C. Jennings, P. B. Jensen, J. Kermode, J. R. Kitchin, E. L. Kolsbjerg, J. Kubal, K. Kaasbjerg, S. Lysgaard, J. B. Maronsson, T. Maxson, T. Olsen, L. Pastewka, A. Peterson, C. Rostgaard, J. Schiøtz, O. Schütt, M. Strange, K. S. Thygesen, T. Vegge, L. Vilhelmsen, M. Walter, Z. Zeng, and K. W. Jacobsen, *The Atomic Simulation Environment—a Python Library for Working with Atoms*, J. Phys. Condens. Matter **29**, 273002 (2017).

[7] V. Kapil, M. Rossi, O. Marsalek, R. Petraglia, Y. Litman, T. Spura, B. Cheng, A. Cuzzocrea, R. H. Meißner, D. M. Wilkins, B. A. Helfrecht, P. Juda, S. P. Bienvenue, W. Fang, J. Kessler, I. Poltavsky, S. Vandenbrande, J. Wieme, C. Corminboeuf, T. D. Kühne, D. E. Manolopoulos, T. E. Markland, J. O. Richardson, A. Tkatchenko, G. A. Tribello, V. Van Speybroeck, and M. Ceriotti, *I-PI 2.0: A Universal Force Engine for Advanced Molecular Simulations*, Comput. Phys. Commun. **236**, 214 (2019).

[8] S. Sturniolo and J. Kermode, *stur86/neb-Perftests: ARCHER2 Tests* (2022). http://dx.doi.org/10.5281/zenodo.5913303