UNIVERSITY OF CAMBRIDGE

Sampling Configurational Energy Landscapes

Matthew Griffiths

Department of Chemistry

This dissertation is submitted for the degree of
Doctor of Philosophy

Trinity College

May 2019

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This dissertation contains fewer than 60,000 words including appendices, bibliography, footnotes, tables and equations, and has fewer than 150 figures.

Matthew Griffiths
May 2019

Abstract

The computational analysis of high dimensional surfaces is a fundamental problem across a wide range of scientific fields, for example in the study of models of clusters of atoms, glasses, self-assembling systems and biomolecules; machine learning; physics; and other fields. This work presents a variety of novel methods developed to aid the computational study of the structures, dynamics and thermodynamics of systems described by these surfaces, traditionally termed *energy landscapes*.

When studying molecular systems, it is important to be able to quantify measures of similarity or difference between a pair of structures generated from an energy landscape. These measures are needed to make predictions of the properties of a given molecular structure from the known properties of similar others. Equivalently a pair of structures can be aligned into similar orientations to allow an interpolated pathway to be generated between them which can be used to identify the transition states between the pair which is a key limiting step in discrete path sampling. The efficiency of the transition state search is strongly dependent on the quality of the initial interpolation and so the alignment methods used. In this work two novel alignment algorithms are presented and benchmarked against existing algorithms for aligning pairs of structures for both periodic and isolated clusters of atoms. The algorithms respectively demonstrate superior performance for either periodic or isolated structures.

The efficient evaluation of the global thermodynamic properties of an *in silico* system, or analogously, the evidence in Bayesian inference, is a challenge for many high-dimensional systems due to a phenomenon known as broken ergodicity. This problem occurs when the energy barriers between different regions of the energy landscape make it difficult to sample both uniformly. In this work a novel superposition based approach that is embarrassingly parallel, based on the athermal method nested sampling, is introduced and benchmarked against a model system exhibiting broken ergodicity. It is shown that the method reproduces the key features of the heat capacity.

Acknowledgements

First and foremost I would like to thank my supervisor, Prof. David Wales FRS for all his help, support, guidance, suggestions and corrections. In particular I am most grateful for the freedom he gave me to explore and grow through my (often silly) ideas and projects. I must also thank Dr Chris Forman for introducing me to the world of energy landscapes and guiding me through the formative years of my PhD.

The theory sector of the chemistry department, and the Wales group, has one of the friendliest academic environments in which I have ever worked, and I must thank everyone there I have had the pleasure of getting to know. Notably, I must thank Sam Niblett for his collaboration and always being ready to provide thoughtful feedback and suggestions and Konstantin Röder for his Teutonic insights into all things related to work, research, beer and life.

I was funded by the Engineering and Physical Sciences Research Council Centre for Doctoral Training in Nanoscience and Nanotechnology (NanoDTC). I would like to thank all the academics, students and friends involved with the NanoDTC for creating an excellent environment to explore and make friends across a wide range of fields of science. I would also like to thank the Department of Chemistry and the Trinity College Rouse Ball fund for funding travel to conferences.

In my time at Trinity College I have been extremely lucky to have made so many friends that have all helped me in their own ways. The many members of the BA Society and Boat Club have been of welcome support and companionship. In particular, I would like to thank Peter Ford for his friendship, suggestions, corrections and willingness to talk 'shop' long after anyone else would have given up. My fellow residents of Marshall Road, Sam Bell, John Grenfell-Shaw, Preeyan Parmar and Imogen Grant have all been invaluable in their own ways, for which I am eternally grateful. Specifically, I must thank Imogen for her unwavering support and reminders that commas are not, always, necessary.

I dedicate this thesis to John.

Table of contents

List of figures

1 Introduction

> "Young man, in mathematics you don't understand things. You just get used to them."
>
> — John von Neumann

The curse of dimensionality is a fundamental problem faced by any method attempting to study complex systems. As the number of free variables and dimensions increase, the structure of the space being studied becomes exponentially sparser. Essentially the ratio of the fraction of space that is interesting compared to the fraction of space that is not decreases exponentially quickly, which means that it becomes much more difficult to sample, cluster or search for points of interest.

The study of energy landscapes, pioneered in chemical physics, has developed a variety of methods to enable the study of *in silico* models of complex chemical systems, in particular clusters, glasses and biomolecules [1]. These methods have been extended to a wide variety of fields, such as machine learning [2–6], self-assembly [7, 8] and many others.

In the energy landscapes approach a potential energy surface (PES) is defined over the space of all possible configurations of the system, and the properties of the system are related to this energy surface.

The curse of dimensionality manifests itself in a variety of ways.

- As the potential energy increases the number of local minima rises exponentially [9–11].
- It is impossible to enumerate all the minima that contribute to the properties of the system.
- The vast majority of the configuration space corresponds to unphysically high energies.

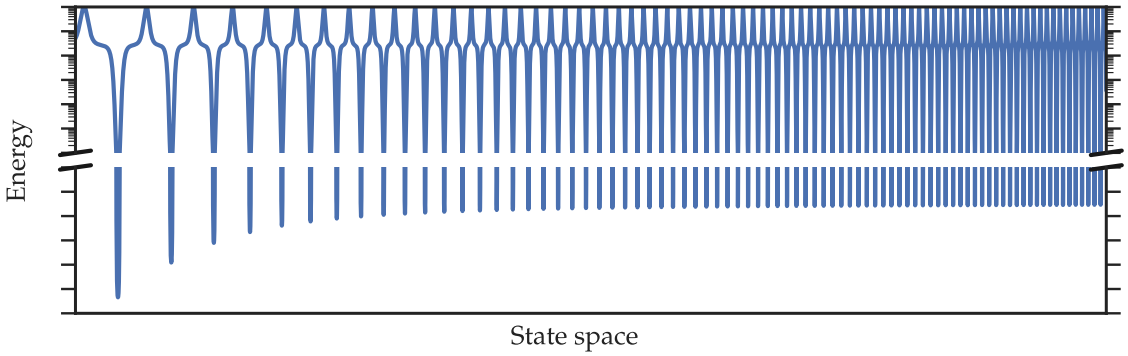


Fig. 1.1 A schematic of the structure of a high-dimensional energy surface. The vast majority of the configuration volume is high energy, which is not physically relevant, whilst the regions of low energy, which are of physical interest, are sparse.

- Only very small moves that sample from a local volume in configuration space will remain low energy. Larger moves will generally end up in the unphysically high energy region, which occupies the vast majority of the total configuration space.

These behaviours are illustrated in fig. 1.1.

One of the most efficient ways to survey an energy landscape is the basin-hopping algorithm, a stochastic global optimisation approach [12–14] described in section 2.1. It can be used to produce a database of low energy minima from which it is possible to estimate the low-temperature thermodynamics of the system using the harmonic superposition approximation, as explained in section 2.4.2.

It is not possible to understand the dynamics of the system from just a database of minima, because the rearrangement pathways between the states associated with different minima must also be considered. Discrete path sampling is a framework that identifies the transition states connecting a database of minima which can be used to study the dynamics of the system [15–17]. Standard techniques from uni-molecular rate theory [18], or rare event methodology using explicit dynamics [19] can then be used to calculate rate constants for individual minimum-to-minimum transitions and hence construct kinetic transition networks [20–24].

The initial pathway obtained between distant end points may be the union of many individual minimum-transition state-minimum paths [25], and is likely to require extensive refinement to locate kinetically relevant routes. Substantial gains in efficiency are likely if the end points can be aligned to improve the interpolation and reduce the initial path length [26, 27]. Finding connections between designated end points may also be helpful for some path sampling approaches that employ explicit

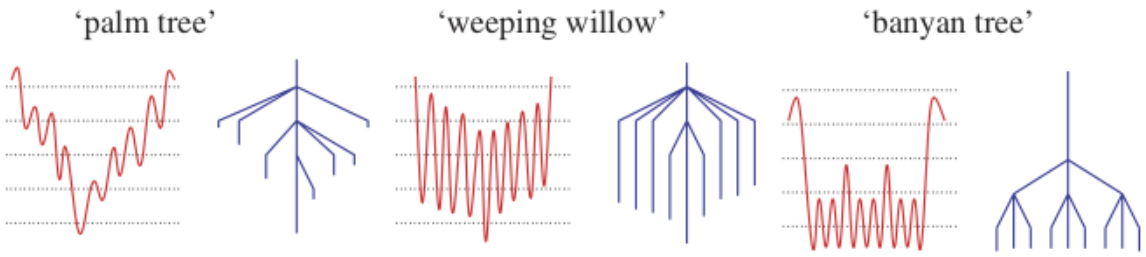


Fig. 1.2 Disconnectivity graphs of several one-dimensional potential energy functions. The dashed lines indicate the energies where the superbasins were calculated, figure reproduced with permission from Wales [17].

dynamics. Unfortunately, alignment can require optimisation of permutational isomers as well as translational and rotational degrees of freedom. Finding the best solution then becomes a global optimisation problem in its own right. In section 2.3 we review existing alignment algorithms, in chapters 3 and 4 we introduce two new alignment algorithms, and in chapter 5 we demonstrate how these algorithms exhibit improved accuracy and efficiency over existing algorithms.

The database of minima and transition states can be used to construct a disconnectivity graph (DG), as illustrated in fig. 1.2, which shows the energy level above which minima become *connected*. The exact definition of what being connected means can vary, leading to different DGs, which is explored in section 6.7.2. For the standard definition, two minima are said to be connected at a given energy threshold if there exists a sequence of transition states connecting them which are all below the threshold. The structure of a DG gives insights into the dynamical behaviour of the system being studied [17].

In addition to understanding the dynamics of a system, we are also often interested in understanding its global thermodynamic properties. Evaluating these thermodynamic properties amounts to performing integrals over the PES, which can in general only be attempted by Monte Carlo sampling techniques, due to the high dimensionality. These types of integral are also important for calculating the *evidence* in Bayesian inference, see section 2.5.3.

For these sampling problems, the curse of dimensionality tends to manifest in the form of *broken ergodicity*. Any sampling technique must ensure that it samples uniformly over all regions of the same energy. However if there are large energy barriers between two *disconnected* regions, then it can take random walks exploring the space an extremely long time to cross the barrier, so the convergence time of the sampling will become extremely long. Methods that have been developed to

evaluate these integrals and the approaches used to deal with broken ergodicity are reviewed in section 2.4. In chapter 6 we introduce a new approach, which is embarrassingly parallel, and a new method for sampling from hard constraints and benchmark them on a model system of a cluster of 31 Lennard-Jones atoms.

In chapter 7, we summarise the main findings of this work and suggest avenues for future work.

2 Methods

2.1 Global optimisation

Basin-hopping (BH) is a global optimisation technique [12–14] where any point in configuration space is associated with the local minimum of the corresponding basin of attraction, of a given minimisation method [14]. The basin of attraction is defined as the set of points that minimise to the same minimum for a given minimisation method. Basins of attraction are only guaranteed to be contiguous when steepest-descent minimisation is used. The boundary between two contiguous basins of attraction is a watershed [28] or transition surface. See fig. 2.1 for a schematic view of the BH transformation which allows the algorithm to take moves to high energy configurations, whilst still sampling the low energy states of interest (see fig. 1.1). The key point about this transformation is that it removes the downhill barriers between local minima which can trap alternative algorithms. We can see how in fig. 2.1 a multiminimum function (multimodal in terms of likelihood) has been turned into a broad funnel. This transformed landscape is comparatively much easier to explore as it is possible to make efficiently much larger moves.

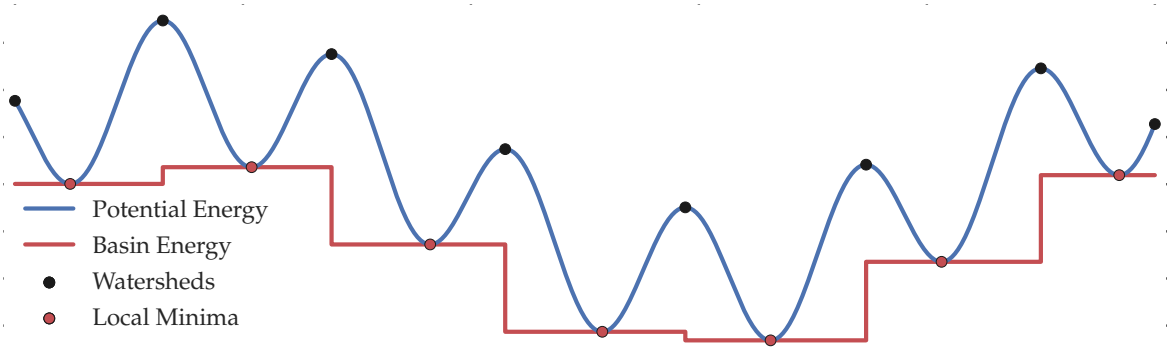


Fig. 2.1 Illustration of how the basin-hopping algorithm produces a stepped landscape. Every point in the basin of attraction of a minimum is associated with the energy of that minimum.

Algorithm 1 Basin-Hopping

Input: $\boldsymbol{R}_{\text{start}}$, T, d ▷ Starting configuration, fictitious temperature, starting stepsize
Output: $\boldsymbol{V}^Q_{\text{min}}$ and $\boldsymbol{R}_{\text{min}}$ ▷ List of energy and configuration of minima
$\boldsymbol{V}^Q_{\text{min}} = \{\}$, $\boldsymbol{R}_{\text{min}} = \{\}$ ▷ Initialise empty lists
$V^Q_\mu = \min\{V(\boldsymbol{R})\}$, $\boldsymbol{R}_\mu = \arg\min\{V(\boldsymbol{R})\}$ ▷ Minimise initial configuration $\boldsymbol{R}$
add V^Q_μ, $\boldsymbol{R}_\mu$ to $\boldsymbol{V}^Q_{\text{min}}$, $\boldsymbol{R}_{\text{min}}$
repeat
  $\boldsymbol{R}_{\text{new}} = \boldsymbol{R}_\mu + \boldsymbol{s}(d)$ ▷ Make random move to generate new configuration
  $V^Q_{\text{new}} = \min\{V(\boldsymbol{R}_{\text{new}})\}$ ▷ Minimise $\boldsymbol{R}_{\text{new}}$
  if Accept$(V^Q_{\text{new}}, V_\mu, T)$ **then** ▷ Test acceptance criterion
    $\boldsymbol{R}_\mu = \arg\min\{V(\boldsymbol{R}_{\text{new}})\}$ ▷ Move to accepted minimum
    $V^Q_\mu = V^Q_{\text{new}}$ ▷ Change acceptance energy
    if $\boldsymbol{R}_\mu \notin \boldsymbol{R}_{\text{min}}$ **then** ▷ Check whether minimum has already been found
      add V^Q_μ, $\boldsymbol{R}_\mu$ to $\boldsymbol{V}^Q_{\text{min}}$, $\boldsymbol{R}_{\text{min}}$
    end if
  end if
  if Adjustable T or d **then**
    Adjust T, d ▷ To keep acceptance rate approximately fixed
  end if
until Termination condition

For a potential energy surface, $V(\boldsymbol{R}) : \mathbb{R}^{3N} \to \mathbb{R}$, the basin-transformation can be defined as $V(\boldsymbol{R}) \to \widetilde{V}(\boldsymbol{R}) = \min\{V(\boldsymbol{R})\}$, where $\min\{V(\boldsymbol{R})\}$ is the potential energy resulting from a local minimisation or *quench* of $V(\boldsymbol{R})$ starting at $\boldsymbol{R}$ and $\arg\min\{V(\boldsymbol{R})\} = \boldsymbol{R}_m$ returns the location of the minimum associated with this quench.

A simple overview of the BH algorithm is shown in algorithm 1, where repeated random steps are taken corresponding to coordinate perturbations followed by minimisations. A pseudo-temperature is used to allow the acceptance rate of these moves to be kept approximately constant, and the size of the random jumps can also be varied to adjust the acceptance rate, with larger steps being less likely.

The Metropolis criterion is a commonly used acceptance criterion, though other criteria can be used. A simple step displacement function, $\boldsymbol{s}(d)$, involves randomly displacing each coordinate in the range $[-d, d]$, although a multivariate Gaussian or uniform hypersphere could also be used. We can adjust the pseudo-temperature and/or the step size to keep the acceptance rate close to a target value. The algorithm returns a list of minima encountered, $\boldsymbol{R}_{\text{min}}$, and their energies, $\boldsymbol{V}^Q_{\text{min}}$, where the superscript Q indicates that these are *quenched* energies.

The standard BH algorithm cannot be used to obtain thermodynamic information as the stepsize and pseudo-temperature are updated during the simulation, and the location of the replica is moved to the location of the minimum, which all break detailed balance, hence the distribution of points that are encountered is also unknown. A variety of extensions to BH have been developed using the superposition approach [1, 10, 29–32] , which enable the density of states to be extracted.

Statistically valid samples from the canonical distribution of $\widetilde{V}(\boldsymbol{R})$ can be generated if BH is performed at fixed temperature and stepsize, though at the cost of decreased global optimisation efficiency.

2.2 The Lennard-Jones potential

The Lennard-Jones (LJ) potential is a simple representation for the energy of a pair of atoms:

$$V_{\mathrm{LJ}}(r) = 4\,\epsilon_{\mathrm{LJ}}\left[\left(\frac{\sigma_{\mathrm{LJ}}}{r}\right)^{12} - \left(\frac{\sigma_{\mathrm{LJ}}}{r}\right)^{6}\right], \tag{2.1}$$

where ϵ_{LJ} and $2^{1/6}\sigma_{\mathrm{LJ}}$ are respectively the pair equilibrium depth of the potential well and separation. When applied to homoatomic systems both ϵ_{LJ} and σ_{LJ} can be set equal to unity to make the potential dimensionless without loss of generality. Its computational simplicity means that it has been extensively studied as a model system. Isolated clusters of n interacting atoms (LJ_n) have been extensively studied at a range of cluster sizes [33] making LJ clusters useful model systems for benchmarking various methods.

2.3 Structural comparison

Quantifying the difference or similarity between two structures is of broad relevance. In a chemical context we may be interested in using measures of structural similarity among a set of chemical structures to predict various chemical properties, for example in quantitative structure-activity relationships (QSAR) where statistical models for predicting the chemical and biological activities of new structures are generated from data about known structures [34]. In this field many structural comparison protocols utilise additional information about the structures, so that the comparison is performed primarily on the chemically active region of the structure.

A related field is that of machine learning of chemical properties where a variety of approaches have been developed. Here it is of particular importance that the methods effectively capture the structural information in ways that it is easy for the machine learning algorithms to learn from [35–38].

Quantifying the similarity between structures can allow two configurations to be aligned to match each other as closely as possible. This alignment is particularly useful in DPS [15–17] which identifies pathways and transition states between local minima on the energy landscape. The initial pathway obtained between distant minima may be the union of many individual minimum-transition state-minimum paths [25], and is likely to require extensive refinement to locate kinetically relevant routes. Substantial gains in efficiency are likely if the end points can be aligned to improve the initial interpolation [26, 27] and reduce the corresponding path length.

Optimal alignment for connection purposes usually corresponds to minimising the Euclidean distance between the two end points in $3N$-dimensional configuration space where N is the number of atoms. However, there are cases where the minimum distance results in incorrect local permutational alignment, producing artificially high barriers if the permutations are not corrected [26]. For large biomolecules a local permutational alignment procedure was introduced to solve this problem [26], combining translational and orientational degrees of freedom with the shortest augmenting path algorithm [39] for each group of permutable atoms and an adjustable number of atoms from the immediate environment.

The Euclidean distance in configuration space is simply related to the root-mean-square distance (RMSD) by a factor of $\sqrt{N}$, so we can use these quantities interchangeably. RMSD is the most commonly used metric for comparing two different structures. However, the exhaustive, deterministic calculation of the minimal RMSD with respect to translational, rotational and permutational symmetries scales combinatorially with the number of identical atoms in the system [40]. The difficulties associated with using RMSD have led to the development of a wide variety of alternative metrics for quantifying the dissimilarity between structures [41].

The problem of 3D point set registration in computer vision is analogous to structure alignment in chemical systems. It is used in many different applications, for example in 3D surface reconstruction [42], alignment of magnetic resonance images and computer aided tomography scans [43], optical character recognition [44], and range image matching [45]. In computer vision the correspondence between point sets usually does not need to be one-to-one, in contrast to most chemical alignment problems.

While it is possible to compare periodic structures with different lattice parameters [46], many algorithms (including the algorithms introduced in this work) assume that the structures being compared have the same lattice parameters.

In this section we review the existing methods used to compare structures, with a particular focus on the alignment algorithms, both in the computer vision and chemical physics literature.

2.3.1 Root-mean-square distance

Two structures (or in computer vision terms, point sets), p and q, can each be defined by N atomic coordinates, $\boldsymbol{R}^p = \{\boldsymbol{r}_1^p, \boldsymbol{r}_2^p, ..., \boldsymbol{r}_N^p\} \in \mathbb{R}^{3N}$ and $\boldsymbol{R}^q = \{\boldsymbol{r}_1^q, \boldsymbol{r}_2^q, ..., \boldsymbol{r}_N^q\} \in \mathbb{R}^{3N}$. The generalised Euclidean distance (norm)

$$|\boldsymbol{R}^p - \boldsymbol{R}^q| = \left(\sum_{j=1}^{N} |\boldsymbol{r}_j^p - \boldsymbol{r}_j^q|^2 \right)^{1/2}, \tag{2.2}$$

is not a good metric, because it is not invariant to symmetries of the Hamiltonian, in particular the rigid-body motions and permutations of identical atoms. For an isolated cluster in the absence of external fields the energy is invariant to overall translation and rotation, and to permutations of identical atoms. Similarly, for a periodic system, point group symmetries, permutations, and global translations leave the energy unchanged. The RMSD between two structures is better defined as the minimum of the Euclidean norm with respect to all these symmetries. For an isolated cluster this definition becomes

$$\mathrm{RMSD}(p,q) = \frac{1}{\sqrt{N}} \min_{\boldsymbol{M},\boldsymbol{P},\boldsymbol{D}} |\boldsymbol{R}^p - \boldsymbol{P}(\boldsymbol{R}^q \boldsymbol{M}^\top - \boldsymbol{D})|, \tag{2.3}$$

where $\boldsymbol{P}$ is a $3N \times 3N$ permutation matrix of the atomic coordinates, $\boldsymbol{D} \in \mathbb{R}^{3N}$ contains N copies of $\boldsymbol{d} \in \mathbb{R}^3$ the global displacement vector of the structure, $\boldsymbol{M} \in \mathbb{R}^{3N \times 3N}$ is a block diagonal matrix, containing N copies of a rotation matrix $\mathbf{m} \in \mathrm{SO}(3)$ and $\boldsymbol{M}^\top$ indicates the matrix transpose of $\boldsymbol{M}$ [1].

Similarly, for a periodic system we can define

$$\mathrm{RMSD}(p,q) = \frac{1}{\sqrt{N}} \min_{\boldsymbol{L},\boldsymbol{P},\boldsymbol{D},\boldsymbol{S}} |\boldsymbol{R}^p - \boldsymbol{P}(\boldsymbol{R}^q \boldsymbol{S} - \boldsymbol{D} - \boldsymbol{L})|, \tag{2.4}$$

where $\boldsymbol{S} \in \mathbb{R}^{3N \times 3N}$ is a block diagonal matrix containing N copies of a 3×3 matrix corresponding to symmetry operations of the periodic supercell, $\boldsymbol{L} = (\boldsymbol{l}_1, \boldsymbol{l}_2, ..., \boldsymbol{l}_N) =$

$L\boldsymbol{J} \in \mathbb{R}^{3N}$ is a set of lattice vectors, with $\boldsymbol{J} \in \mathbb{Z}^{3N}$ and L the length of the unit cell. In the present work we consider a cubic supercell, but the above definition is easily generalised.

Calculating the RMSD is therefore a global optimisation problem, requiring the identification of the relative lattice vectors and/or rotation, permutation and translation that define the global minimum. Locating this global minimum is equivalent to finding the optimal alignment of two structures where the total squared displacement between them is minimised. Henceforth, we will refer to finding the minimal RMSD between two structures as aligning them.

2.3.2 Existing methods

A variety of methods have been developed to calculate the minimal RMSD. They are either heuristic or are not guaranteed to locate the global minimum RMSD in polynomial time. A variety of algorithms have also been developed in the computer vision literature that attempt to minimise alternative metrics, as discussed below.

Partial algorithms

Various algorithms have been developed that, in polynomial time, will find the global minimum for one of the symmetries over which we are minimising.

Translational Alignment For an isolated cluster it can easily be shown that the best alignment will always occur when the centres of coordinates coincide, independent of the permutations, rotations and the number of different chemical species present [40]. This result does not apply to a periodic system, because the centre of coordinates is not well defined, although the mean displacement between the two structures must be zero when the RMSD between them is minimised.

Permutational Alignment If the minimisation is restricted to permutations then the optimal permutation can be found in polynomial time using the Hungarian algorithm [47] which scales approximately as $O(N^{2.5})$ and the shortest-augmenting path algorithm which is faster and scales as $O(N^2)$ [39]. Both these algorithms are forms of primal-dual methods that perform a simultaneous primal constrained maximisation and dual constrained minimisation; when both problems are satisfied the optimal solution has been found [39].

Rotational Alignment Finding the optimal rotational alignment for a fixed permutation has an analytic solution, $O(N)$, and can be achieved using quaternions [48] or Lagrange multipliers [49].

Lattice Vectors For a given displacement and permutation the lattice vector that minimises the RMSD between the two structures can be found in $O(N)$ operations, by rounding the relative displacement vector between pair of atoms to the closest lattice vector. Finding the global translation and set of lattice vectors that minimises the RMSD can be found by setting the mean displacement between the structures to zero after finding the optimal lattice vectors in $O(N)$ operations

Point Group Symmetries The RMSD should also be minimal with respect to the point group symmetries of the periodic structure which can be enumerated. For isolated structures the inverted structure ($\boldsymbol{R}^q \to -\boldsymbol{R}^q$) may also be relevant.

Unfortunately, iteratively minimising each of these symmetries in turn does not guarantee that the global minimum RMSD will be found. Guaranteeing this condition would require testing every possible permutation. Since there are $N!$ possible permutations for a homoatomic system this approach is prohibitively expensive for all but the smallest systems.

Full algorithms

Monte Carlo alignment Sadeghi et al. [40] developed a Monte Carlo algorithm for calculating the global minimum RMSD for both clusters and periodic systems [50]. In this method an initial permutational alignment is performed by either matching the principal axes of the moment of inertia, or by matching atoms with similar local environments. Random permutations followed by a rotational or displacement alignment are then applied, and the new alignment is accepted if the RMSD is less than the old RMSD plus a small adjustable parameter. This parameter is changed dynamically during the simulation to keep the acceptance rate around 50%. The number of MC iterations required to find the global minimum RMSD for this method scales approximately exponentially for atomic Lennard-Jones clusters [40].

Iterative closest point This is one of the most commonly used algorithms in computer vision to align point sets. This algorithm iteratively pairs up the closest points in the two point clouds (the computer vision equivalent to a structure) and then

minimises the distance squared between them until no further improvement appears [45]. Variants of this method have been developed, incorporating the expectation-maximisation algorithm [51] or using the Levenberg-Marquardt approach [44]. These local minimisation methods can be combined with a branch and bound scheme to determing the global minimum of the cost function [52–54]. However, because the cost function measures the nearest neighbour distance, these algorithms will not necessarily find the global RMSD.

Kernel correlation An alternative approach developed for point set registration is based on maximising the kernel correlation between two points sets, p and q. For a kernel function, $K(\boldsymbol{r}, \boldsymbol{r}')$, we can define the kernel correlation between two vectors, $\{\boldsymbol{r}_j^p, \boldsymbol{r}_{j'}^q\} \in \mathbb{R}^3$, as

$$KC(\boldsymbol{r}_j^p, \boldsymbol{r}_{j'}^q) = \int K(\boldsymbol{r}, \boldsymbol{r}_j^p) K(\boldsymbol{r}, \boldsymbol{r}_{j'}^q)\, \mathrm{d}\boldsymbol{r}, \tag{2.5}$$

and the total kernel correlation between two structures as

$$KT(p, q) = \sum_j \sum_{j'} KC(\boldsymbol{r}_j^p, \boldsymbol{r}_{j'}^q). \tag{2.6}$$

The registration of two point sets is achieved by performing a non-linear optimisation of the total kernel correlation [55]. This method is directly analogous to the extended Gaussian image approach [56]. This approach was used by Makadia et al. [57] to align point sets with very little overlap, optimising rotations with a discrete $SO(3)$ Fourier transform (SOFT) [58] to find the best correlation between discrete histograms of the extended Gaussian images. The SOFT has also been used to identify binding regions between proteins [59, 60].

Branch and bound RMSD Hong et al. [61] developed a branch and bound based method for deterministically calculating the RMSD between two configurations of identical atoms. The algorithm works by progressively bounding the RMSD between subsets of permutations of the atoms in both structures. By bounding the lowest possible RMSD for each subset the algorithm can eliminate those that give poorer alignments, removing that region of search space. The algorithm exhibited better than $O(N^2)$ performance for aligning identical structures generated randomly, with permuted components.

Our own analysis and implementation of this algorithm suggest that the performance is not competitive for alignment of different structures. The number of

permutation subsets with a lower bound below a given distance scales approximately exponentially with the distance which means that the computational complexity scales approximately with the exponential of the minimum RMSD.

Methods implemented in Cambridge energy landscape software PERMDIST, ATOMMATCHFULL and ATOMMATCHDIST are heuristic algorithms for estimating the global RMSD which have been developed and implemented in the public domain programs GMIN [62] and OPTIM [63]. These algorithms are described below:

PERMDIST This algorithm applies a successive set of permutational alignments, using the shortest augmenting path algorithm [39], each one followed by an overall rotational or translational alignment [26]. The procedure is repeated until a minimum RMSD in permutational space is reached. Because this process is not guaranteed to give the global RMSD it is restarted from multiple random initial rotations/displacements. This approach has much in common with the iterative closest point based algorithms.

ATOMMATCHFULL This algorithm was developed to identify structural isomers of periodic systems by successively superimposing every pair of atoms and then checking how many other atoms in one structure are within a certain distance of an atom in the second structure (in which case the two atoms are said to "match") [64]. An exhaustive search is performed, superimposing all pairs of atoms within the smallest permutable group which allows us to fix the global translation. Once the global translation is found the permutational assignment problem can be solved to get the full permutation. Because this algorithm attempts to maximise the number of matches it does not necessarily find the global RMSD. It scales approximately as $O(N^4)$, so for large systems it is computationally expensive.

ATOMMATCHDIST This algorithm is based on ATOMMATCHFULL, but reduces the computational expense by exiting some of the loops over atoms early if the current trial superposition does not give enough matches [64]. This strategy sometimes gives significantly poorer alignments than ATOMMATCHFULL.

Methods implemented in KPLOT Two algorithms have been developed for use in the structure visualisation and analysis program KPLOT [65] for identifying

isostructural similarities between structures. These methods do not attempt to minimise the RMSD, but are included for reference.

CMPZ This is a method developed for comparing crystallographic structures. It looks for the set of affine transformations that map atoms from the rescaled unit cell of one crystal structure onto atoms in the rescaled unit cell of the second structure; it then performs the inverse transform to check that the mapping is bijective. If all the atoms map to within a certain tolerance of each other then the structures are described as equivalent [46]. If the unit cells have different specifications the algorithm will detect whether they are equivalent, whether one unit cell is a supercell of the other, and/or whether one structure is a substructure of the other. This algorithm is one of the few algorithms capable of comparing structures with different lattice parameters.

CCL This approach extends the CMPZ algorithm to clusters, by seeking the affine transformation that maps one set of atoms onto another, and it can be used to identify structural isomers. It will also identify whether a cluster forms a smaller part of a larger system [66].

Alternative metrics

Due to several difficulties associated with calculating and using the RMSD a variety of alternative metrics and descriptors have been developed. A number of issues motivated these developments:

- Calculating the global minimum RMSD can be computationally difficult.
- The RMSD changes continuously but not smoothly as the coordinates of one structure are smoothly varied, because there are discontinuities in the gradient when the optimal permutation changes.
- The RMSD does not always accurately capture the degree of (dis)similarity between two structures in the most useful way [26, 41].
- The RMSD can only be used to compare structures with the same number of equivalent atoms.

We now briefly review some of the metrics and methods that are related to the approaches developed in the present work.

Gaussian kernels A variety of algorithms (including those we employ below) are based on the definition of a density function using a sum of Gaussian kernels of width σ_G,

$$\rho_p(\boldsymbol{r}) = \sum_{j=1}^{N} \exp\left(\frac{|\boldsymbol{r} - \boldsymbol{r}_j^p|^2}{2\sigma_G^2}\right), \qquad \rho_q(\boldsymbol{r}) = \sum_{j=1}^{N} \exp\left(\frac{|\boldsymbol{r} - \boldsymbol{r}_j^q|^2}{2\sigma_G^2}\right). \tag{2.7}$$

These densities are equivalent to the kernel functions used in the kernel correlation point set registration methods [55–57]. The properties of the overlap integral with respect to a set of rigid body motions, $T : \mathbb{R}^3 \to \mathbb{R}^3$,

$$\Omega^{pq}(T) = \iiint \rho_p(\boldsymbol{r})\rho_q(T(\boldsymbol{r}))\mathrm{d}\boldsymbol{r}, \tag{2.8}$$

are considered. The Gaussian kernel is one of the more common kernels used to generate a density function from a list of coordinates, but others have been proposed [67–69]. This functional representation of the densities is permutationally invariant and smooth.

Smooth Overlap of Atomic Positions (SOAP) This descriptor compares the local environment of two atoms by centring the density functions on two specific atoms and then evaluating the overlap integral with respect to all possible rotations [67]. The calculation can be performed efficiently by expressing the densities as truncated sums of spherical harmonics whose integrals can be evaluated analytically to obtain density functions. This procedure allows the local environment of different atoms to be compared. There are a variety of ways that the local similarity metrics can be combined to determine the global similarity of two structures [41]. This method has been used to improve potential energy surface fitting within the Gaussian approximation potentials (GAP) framework [70, 71].

Maximum overlap of kernels The global maximum value of eq. (2.8) with respect to rotations has been used as an alternative metric for clusters through searches based on simulated annealing [68].

Fingerprint functions Fingerprint functions produce vectors of structural properties that are invariant to symmetries of the Hamiltonian. These properties are often based on the eigenvalues of various matrices associated with the structure, such as Coulomb matrices [72], or kernel overlap matrices [40, 73]. The norm of

the ordered eigenvalues can be used as a metric, and if the vector is larger than the number of degrees of freedom it can provide a unique identifier for the structure [40]. Properties of the interatomic distance matrix have also been used to construct descriptors and metrics [74].

Other metrics A variety of other metrics have been developed, based on a number of properties including bond-order parameters [70, 75–77], 'similarity functions'[78], bond network graphs [79], localised Coulomb representations (related to Coulomb matrices in the same way as SOAP relates to kernel overlap matrices) [80], and radial distribution functions [81]. The similarity of proteins has been calculated by projecting the shape of the protein as an expansion of Wigner-D functions and calculating the correlation between these expansions [82].

2.4 Sampling energy landscapes

In this section we review the methods and challenges associated with the calculation of the global thermodynamic properties of an energy landscape. These properties are determined by integrals of the form,

$$\mathcal{I}_{\Phi_0}[f] = \int_{\Phi_0} f(V(\boldsymbol{R}))\,\mathrm{d}\boldsymbol{R}, \tag{2.9}$$

where Φ_0 is the domain of the integral and $\mathcal{I}_{\Phi_0}[f]$ is a functional of the integral of f over Φ_0. Due to the curse of dimensionality these integrals generally have to be estimated numerically using stochastic methods. These approaches can generally be classified under two broad classes, *thermal* or *athermal*.

2.4.1 Thermal methods

Thermal methods directly generate samples from probability distributions related to $f(V(\boldsymbol{R}))$, often using Markov chain Monte Carlo (MCMC), and usually sampling from the canonical distribution. More advanced thermal methods may sample from a set of related probability distributions simultaneously [33, 83–90]. Due to the 'curse of dimensionality' the vast majority of configuration space will be extremely high in energy, and to generate statistically valid samples thermal methods must obey detailed balance. Almost all large moves will land in high energy regions and so will be rejected. Hence these sampling methods must take short local moves to

have a reasonable acceptance rate, increasing the time taken to simulate large-scale rearrangements. The convergence is dominated by the time taken to simulate such rearrangements which can lead to broken ergodicity. The performance may sometimes be improved by making more efficient moves, for example using Hamiltonian Monte Carlo [91], molecular dynamics [90], and/or by replica exchange between different probability distributions [83–87] (see section 2.4.1).

Markov Chain Monte Carlo

Unless a mapping to the uniform distribution can be found then it is not generally possible to directly sample from an arbitrary probability distribution. In these cases Markov Chain Monte Carlo (MCMC) is normally used as the basis to generate samples from a target probability distribution, $\mathrm{Pr}^0(\boldsymbol{R})$.

The samples are generated by constructing a Markov chain whose equilibrium distribution is the same as the target distribution. This approach is based on the principle of detailed balance,

$$T^0(\boldsymbol{R} \to \boldsymbol{R}')\mathrm{Pr}^0(\boldsymbol{R}) = T^0(\boldsymbol{R}' \to \boldsymbol{R})\mathrm{Pr}^0(\boldsymbol{R}'), \tag{2.10}$$

where $T^0(\boldsymbol{R} \to \boldsymbol{R}')$ is the probability of transitioning from state $\boldsymbol{R}$ to state $\boldsymbol{R}'$, so the rate of flow from $\boldsymbol{R}$ to $\boldsymbol{R}'$ is the same as the reverse flow. When simulating a Markov process an instance of a Markov process is commonly referred to as a *replica*. A Markov process is uniquely defined by the transition probability $T^0(\boldsymbol{R} \to \boldsymbol{R}')$. Provided that a Markov process satisfies eq. (2.10), and there exists a unique stationary distribution, the stationary distribution of the Markov process will correspond to $\mathrm{Pr}^0(\boldsymbol{R})$.

While it is not known how to directly sample from Pr^0, it is generally straightforward to define a Markov process that satisfies eq. (2.10). The transition probability is normally split into the product of an acceptance distribution, $A^0(\boldsymbol{R} \to \boldsymbol{R}')$, and proposal distribution, $h^0(\boldsymbol{R} \to \boldsymbol{R}')$,

$$T^0(\boldsymbol{R} \to \boldsymbol{R}') = A^0(\boldsymbol{R} \to \boldsymbol{R}')h^0(\boldsymbol{R} \to \boldsymbol{R}'). \tag{2.11}$$

The proposal distribution is used to generate moves, so it is commonly a Gaussian or uniform distribution. The acceptance distribution determines the probability of accepting or rejecting these proposed moves. In standard MCMC the proposal

distribution is chosen to be symmetric so

$$h^0(\boldsymbol{R} \to \boldsymbol{R}') = h^0(\boldsymbol{R}' \to \boldsymbol{R}) \tag{2.12}$$

which means that the acceptance distribution must satisfy,

$$\frac{A^0(\boldsymbol{R} \to \boldsymbol{R}')}{A^0(\boldsymbol{R}' \to \boldsymbol{R})} = \frac{\mathrm{Pr}^0(\boldsymbol{R}')}{\mathrm{Pr}^0(\boldsymbol{R})}. \tag{2.13}$$

The Metropolis criterion

$$A^0(\boldsymbol{R} \to \boldsymbol{R}') = \min\left(1, \frac{\mathrm{Pr}^0(\boldsymbol{R}')}{\mathrm{Pr}^0(\boldsymbol{R})}\right) \tag{2.14}$$

satisfies this condition, and defines the Metropolis–Hastings algorithm. As the acceptance criterion only depends on the ratio of the probabilities, the probability distribution does not need to be normalised [90].

Molecular dynamics/Hamiltonian Monte Carlo

Perhaps the most obvious way to generate representative samples of a molecular configuration is to directly simulate the behaviour *in silico*. Assuming that the thermal energy of the system is high enough to render quantum effects irrelevant, then it is possible to directly model the molecular dynamics (MD) classically [90]. Provided the MD simulation is run for long enough at a given temperature, it can capture the required properties. Closely related to MD is Hamiltonian Monte Carlo (HMC) which extends MD simulation methods and ideas to modelling a wider range of probability distributions [91, 92].

In MD and HMC the size of the state space is effective doubled to include both the configuration and the velocity (or momentum). While this factor incurs additional computational cost, including the velocity can facilitate more efficient exploration of the available state space by guiding the MC moves to stay in the region of interest [91].

NUTS

One challenge when using HMC is choosing the appropriate length trajectory to simulate, as a poor choice leads can lead to a dramatic reduction in the efficiency of the sampler. The No U-Turn Sampler (NUTS) avoids having to specify this parameter

by recursively doubling a slice of points along a HMC trajectory and terminating once the trajectory starts doubling back on itself (hence the no u-turn). This method has the additional advantage of making the sampling from HMC exact [91].

NUTS still requires a stepsize to chosen for the HMC simulation, however this can be chosen slightly more straightforwardly by adaptively adjusting the stepsize to target an effective acceptance rate [93].

Replica exchange

Replica exchange is one of the most commonly used approaches to accelerate convergence and overcome broken ergodicity in MC simulations. This technique enables samples from different probability distributions to be coupled together in a single simulation, which can be advantageous as it may be easier to sample from certain distributions which may be analytic, have lower barriers, or faster converging MC averages (for example sampling at a higher temperature). Coupling alternative probability distributions to systems that exhibit broken ergodicity can allow simulations to cross barriers faster, accelerating convergence.

Replica exchange enables MC simulations of two probability distributions, $\mathrm{Pr}^0(\boldsymbol{R})$ and $\mathrm{Pr}^1(\boldsymbol{R})$, to swap their current replicas, $\boldsymbol{R}^a$ and $\boldsymbol{R}^b$ [90]. For unbiased sampling the reverse swap must be equally likely, so it must obey detailed balance,

$$T^{RX}(\{\boldsymbol{R}^a,\boldsymbol{R}^b\}\to\{\boldsymbol{R}^b,\boldsymbol{R}^a\})\mathrm{Pr}^0(\boldsymbol{R}^a)\mathrm{Pr}^1(\boldsymbol{R}^b) = T^{RX}(\{\boldsymbol{R}^b,\boldsymbol{R}^a\}\to\{\boldsymbol{R}^a,\boldsymbol{R}^b\})\mathrm{Pr}^0(\boldsymbol{R}^b)\mathrm{Pr}^1(\boldsymbol{R}^a), \quad (2.15)$$

where we have indicated the joint state of $\boldsymbol{R}^a$ and $\boldsymbol{R}^b$ being associated with Pr^0 and Pr^1 respectively, by $\{\boldsymbol{R}^a,\boldsymbol{R}^b\}$. The generalised Metropolis acceptance criterion,

$$A^{RX}(\{\boldsymbol{R}^a,\boldsymbol{R}^b\}\to\{\boldsymbol{R}^b,\boldsymbol{R}^a\}) = \min\left(1,\frac{\mathrm{Pr}^0(\boldsymbol{R}^b)\mathrm{Pr}^1(\boldsymbol{R}^a)}{\mathrm{Pr}^0(\boldsymbol{R}^a)\mathrm{Pr}^1(\boldsymbol{R}^b)}\right), \quad (2.16)$$

will satisfy eq. (2.15), assuming that the jump probability is symmetric, and so can be used to exchange replicas between different MCMC simulations, or to seed an individual MCMC walk with samples from an alternate distribution [83–87] .

Parallel tempering

The canonical example of replica exchange is parallel tempering (PT) where MC simulations are run at different temperatures simultaneously and replicas are peri-

odically exchanged according to eq. (2.16). A key challenge associated with PT and its associated variants is choosing the set of temperatures over which to perform the simulation. If they are too widely spaced then the average energy differences between adjacent replicas will be too large, and exchanges between temperatures will not happen quickly enough whilst having too many replicas will be inefficient. When PT is applied to MD simulations it is normally referred to as molecular dynamics replica exchange (MD-REX) [83–86] .

Hamiltonian replica exchange

It is also possible to exchange replicas between different potential energy distributions [33, 87]. Commonly, harmonic potentials are used as a reference, as sampling from a harmonic distribution corresponds to sampling from a normal distribution. At energies close to the lowest energy minimum the potential function can be approximated fairly accurately by a multidimensional harmonic well. Coupling the MC simulations to the analytic harmonic potential allows replicas to 'tunnel' through the energy barriers between the lowest basins for the actual potential. This harmonic approximation has been successfully applied to calculate heat capacities of various Lennard-Jones clusters using PT [33, 87] and nested sampling [94].

2.4.2 Principle of superposition

It is common practice to split the configuration space into different regions and then tackle the configuration integral in each region independently. Often different regions are associated with different minima, using the basin of attraction of a minimisation algorithm [1, 95]. The total density of states can then be described by the sum over basins of attraction of the minima to give the superposition partition function [1, 10, 30–32, 96]:

$$\Omega(V) = \sum_{\mu \in \boldsymbol{R}_{\min}} P_\mu \Omega_\mu(V), \tag{2.17}$$

where P_μ is the number of distinguishable permutation-inversion isomers of minimum μ, and $\Omega_\mu(V)$ is the density of states for the corresponding basin of attraction (see section 2.4.3).

This approach can be useful, because in many situations determining $\Omega_\mu(V)$ is more straightforward than determining the full density of states. At energies close to the minimum, the potential function can be well approximated by a harmonic

potential, with an analytic density of states. Additionally, there are no barriers within a basin of attraction.

Harmonic superposition approximation

Given a database of minima, the fastest method for estimating the density of states is the harmonic superposition approximation (HSA) [29] where the density of states,

$$\Omega_\mu(V^I) \propto \theta(V^I - V_\mu^Q)(V^I - V_\mu^Q)^{\kappa/2-1}/\overline{\nu}_\mu, \tag{2.18}$$

and configuration volume,

$$\Phi_\mu(V^I) \propto \theta(V^I - V_\mu^Q)(V^I - V_\mu^Q)^{\kappa/2}/\overline{\nu}_\mu, \tag{2.19}$$

of each individual minimum are approximated by a harmonic potential with known analytic form where θ is the Heaviside step function, V_μ^Q is the energy of minimum μ associated with the basin, κ is the number of vibrational degrees of freedom (the number of non-zero eigenvalues of the Hessian), and $\overline{\nu}_\mu$ is the geometric mean of the vibrational normal modes. The harmonic superposition partition function is simple to calculate and accurate at low temperatures, but at high temperatures anharmonic vibrational effects become pronounced introducing systematic errors [29].

Basin-sampling

The harmonic superposition principle can be extended to work at higher temperatures by fitting an anharmonic density of states to a two-dimensional histogram using basin-sampling with PT (BSPT) [89], though the original method used Wang–Landau sampling [88]. This method enables HSA to be coupled to a high-temperature PT simulation. HSA efficiently models the low-temperature thermodynamics, circumventing broken ergodicity whilst PT efficiently samples the high-temperature thermodynamics where broken ergodicity is no longer a problem. The anharmonic correction involves a smooth interpolation between these two schemes.

2.4.3 Athermal methods

Athermal methods attempt to determine the density of states,

$$\Omega(V) = \frac{d\Phi(V)}{dV}, \tag{2.20}$$

where

$$\Phi(V) = \int_{V(\boldsymbol{R})<V} \mathrm{d}\boldsymbol{R} \tag{2.21}$$

is the configuration volume, so eq. (2.9) can then be expressed as,

$$\mathcal{I}_{\Phi_0}[f] = \int_{-\infty}^{\infty} f(V)\Omega(V)\,\mathrm{d}V = \int_{0}^{\Phi(\infty)} f(V)\,\mathrm{d}\Phi(V). \tag{2.22}$$

The density of states can be determined by discretising into a set of energy bins (see section 2.4.3) or determined afterwards by nested sampling [2] (see section 2.4.4).

Bin-based methods

Almost all bin-based methods generate samples by performing a random walk with a *flat-histogram* acceptance criterion,

$$A^{\text{hist}}(j \to j') = \min\left(1, \frac{\Omega_j^{\text{H}}}{\Omega_{j'}^{\text{H}}}\right), \tag{2.23}$$

where Ω_j^{H} is the configuration volume associated with the histogram bin j, so the probability of accepting a move into a given bin is *inversely* proportional to the volume of the bin. This criterion in theory ensures that every bin is visited an equal number of times. The key challenge associated with these methods is that the relative size of adjacent histogram bins can be extremely large if the bin ranges are not carefully chosen, so the probability of any move landing in the smaller bins becomes too low and the simulation will not converge.

Additionally, calculating the optimal acceptance probability requires the density of states to be known which is the required goal. Most histogram based algorithms therefore employ bootstrapping where an estimate of the density of states is progressively refined during the simulation whilst also guiding the sampling. As the estimated density of states changes the acceptance probabilities of the moves during the simulation vary, so these simulations are non-Markovian.

There are several bin-based athermal methods.

Wang–Landau sampling Every MC step moving from bin j to j' updates the WL density of states estimate by a multiplicative factor of $f^{\text{WL}} > 1$. Over the course of the simulation the value of f^{WL} is reduced closer to unity. Convergence is diagnosed when the number of visits to each potential bin is approximately equal.

Transition matrix Monte Carlo An alternative approach to determining the density of states histogram is to treat the histogram bins as discrete states of a Markov process, and sample the transition state matrix between these states. To sample the full transition matrix efficiently, a criterion like eq. (2.23) is needed to ensure that all the bins are sampled approximately equally. The relative configuration volumes of the histogram can then be calculated by a non-linear least squares fit of the logarithmic volumes to the transition matrix [97].

Statistical temperature Monte Carlo Statistical temperature Monte Carlo (STMC) can be viewed as a generalisation of the Wang–Landau approach that effectively interpolates the density of states between the bin ranges. The formulation of the method is different because the quantity of interest is the statistical temperature, $T(V) = (d \ln \Omega(V)/dV)^{-1}$, rather than the density of states [98].

Because STMC and its variants perform an interpolation between the bins, they avoid many of the issues associated with the finite bin widths in the WL approach. Hence STMC may converge more efficiently compared to WL sampling. STMC has been extended to work with both PT [99] and MD [100] simulations.

2.4.4 Nested sampling

Nested Sampling (NS) is an approach that was developed by Skilling [2] to efficiently calculate the evidence in Bayesian inference, as discussed in section 2.5.3, equivalent to the evaluation of eq. (2.9).

Skilling's insight was to reformulate the density of states integral, eq. (2.22), as a Lebesgue integral,

$$\Pr(D|M) = \int \Pr\left(D|M(\boldsymbol{\theta})\right) \Pr\left(M(\boldsymbol{\theta})\right) \, \mathrm{d}\boldsymbol{\theta} = \int_0^\infty \lambda \, \mathrm{d}X^{\mathrm{NS}}(\lambda), \tag{2.24}$$

where $X^{\mathrm{NS}}(\lambda)$ is the *prior volume* enclosed within likelihood contour $\Pr\left(D|M(\boldsymbol{\theta})\right) > \lambda$, and

$$X^{\mathrm{NS}}(\lambda) = \int_{\Pr(D|M(\boldsymbol{\theta}))>\lambda} \Pr\left(M(\boldsymbol{\theta})\right) \, \mathrm{d}\boldsymbol{\theta}, \tag{2.25}$$

Associating $\boldsymbol{\theta} \equiv \boldsymbol{R}$, $\Pr\left(D|M(\boldsymbol{\theta})\right) \equiv f(V(\boldsymbol{R}))$, and taking $\Pr\left(M(\boldsymbol{\theta})\right)$ to be uniform over all of the available configuration space, eq. (2.22) is recovered exactly, and the prior volume becomes proportional to the configuration volume, eq. (2.21). Hence it is possible to define nested sampling in terms of configuration volumes which is how we will describe it.

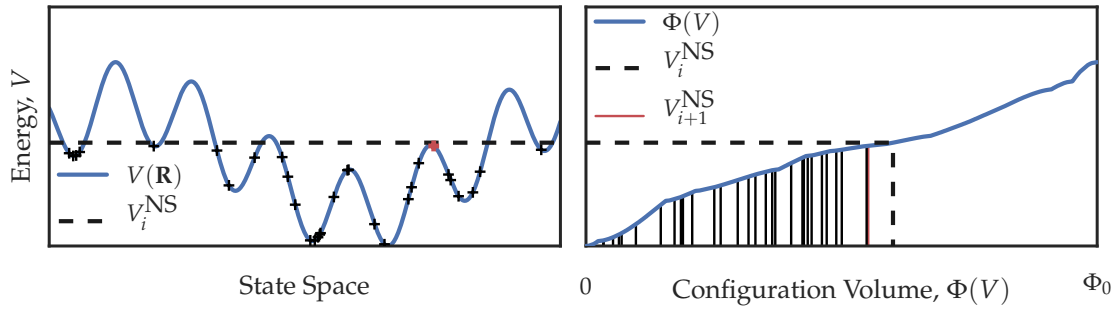


Fig. 2.2 A schematic representation of how nested sampling can calculate the relative difference in configuration volume between two energy thresholds. n^{live} live points are randomly distributed over $\boldsymbol{R}$ such that they all have energy less than V_i^{NS}. The live point with the highest energy, V_{i+1}^{NS} is highlighted in red. As the live points are randomly distributed over $\boldsymbol{R}$ (and so $\Phi(V)$) we know that $\Phi(V_i^{\mathrm{NS}})/\Phi(V_{i+1}^{\mathrm{NS}}) \equiv t_i^{\mathrm{NS}} \sim \mathcal{B}(n^{\mathrm{live}}, 1)$.

Nested sampling works by determining the ratio in configuration volume for decreasing thresholds $\boldsymbol{V}^{\mathrm{NS}} = \{V_1^{\mathrm{NS}}, ..., V_{N^{\mathrm{NS}}}^{\mathrm{NS}}\}$. This list is generated by maintaining a set of n^{live} independent replicas (also known as *live points*) and then iteratively removing the highest energy replica (whereupon it becomes a *dead point*).

A new live point is generated randomly and uniformly within the configuration volume enclosed by the energy contour of the most recently removed dead point. This process means that the ratio of configuration volumes enclosed by the energy contours of successively removed dead points can be modelled by a set of independent beta-distributed variables (see section 2.5.1 and eq. (2.37)),

$$t_j^{\mathrm{NS}} = \frac{\Phi(V_j^{\mathrm{NS}})}{\Phi(V_{j+1}^{\mathrm{NS}})} \sim \mathcal{B}(n^{\mathrm{live}}, 1). \tag{2.26}$$

An overview of the basic nested sampling algorithm is shown in algorithm 2, and a schematic for one step of a nested sampling run is shown in fig. 2.2.

For nested sampling convergence is normally considered to have occurred when the value of the evidence integral (configuration integral) contained by the live points is less than some specified fraction of the total evidence (the partition function), or the energy difference between the highest and lowest live points is less than some tolerance.

If the likelihood is set to be the canonical probability of a configuration, and $\boldsymbol{\theta}$ is the configuration, then the evaluation of the evidence in eq. (2.24) is equivalent to the calculation of the partition function.

Formulating the integral as a Lebesgue integral enables NS to effectively choose the optimal bin width during the course of a NS calculation.

Algorithm 2 Basic nested sampling

Input: n^{live} ▷ Number of replicas to generate
Output: $\boldsymbol{V}^{\max}$ ▷ List of likelihoods of dead points
Initialise empty list $\boldsymbol{V}^{\max} = \{\}$
generate n^{live} independent replicas of live points with sorted energies $\boldsymbol{V} = \{V_1 < V_2 < ... < V_{n^{\text{live}}}\}$
repeat
  Remove dead point $V_{n^{\text{live}}}$ from $\boldsymbol{V}$ and append to $\boldsymbol{V}^{\max}$
  Generate new replica uniformly from prior with energy $V_{\text{new}} < V_{n^{\text{live}}}$
  Insert new energy, V_{new}, into sorted list, $\boldsymbol{V}$, of energies of live points
until Convergence

Dynamic nested sampling During nested sampling it is possible to dynamically change the number of replicas being sampled by increasing the number of replicas in the energy/likelihood ranges that contribute most to the observables, which can be useful to improve the accuracy. The ratio of volumes enclosed by successive energies/likelihoods will still be distributed as eq. (2.34), but the number of live points can now change. This process is known as dynamic nested sampling [101]. Dynamic nested sampling runs can be combined by merging all the energies/likelihoods into a single sorted list. The ratio of volumes enclosed will again be beta distributed, with n^{live} equal to the sum of the live points in all the nested sampling runs considered.

In this framework every step of a nested sampling run can be viewed as the removal of the highest energy live point, followed by the addition of some number of live points sampled uniformly below the energy of the point removed. If no live points are added their number will decrease as the live point with the highest energy is successively removed which is equivalent to removing multiple points at the same time. Here we will assume that in every step of a nested sampling run exactly one live point is removed, but there can be a dynamic number of live points.

Challenges The key computational challenge associated with nested sampling is generating replicas uniformly from the configuration space (the prior) subject to the constraint that the energy/likelihood of the replicas is less/greater than a given cut-off [102, 103]. Three key factors complicate this process.

- The configuration (prior) volume of interest is normally a tiny fraction of the total.
- The potential energy landscape has an exponentially large number of local minima, all corresponding to maxima of the likelihood.
- As the potential energy constraint decreases during the nested sampling simulation, regions in configuration space will become disconnected, and sampling across them becomes challenging.

A variety of approaches have been developed to tackle these problems.

- A hard constraint variant of HMC known as Galilean sampling which exploits isolikelihood contours/potential gradient information [104, 105] to enable long-range directed moves.
- MULTINEST [106] fits a set of intersecting ellipsoidal contours to the set of live points and then performs rejection sampling within the contours though the efficiency of this method will decrease with the dimensionality of the system.
- Diffusive nested sampling [107] uses MCMC to explore a variety of nested sampling distributions to overcome barriers between basins.
- POLYCHORD [102] extends the slice sampling algorithms to multimodal distributions.
- Most approaches discard information when generating new replicas. Importance nested sampling [108] and nested importance sampling [109] allow the inclusion of discarded data points. These possibilities have been used to improve estimates generated by the MULTINEST algorithm.
- Superposition enhanced nested sampling (SENS) [94] uses a population of low energy minima (corresponding to high likelihood) obtained using a global optimisation algorithm, such as BH [12–14], to propose moves to cross barriers that the MCMC walks cannot overcome. In exact-SENS, replicas are generated via Hamiltonian replica exchange; in inexact-SENS new samples are generated at low energies by approximating the potential function as a set of harmonic wells. Both of these approaches enable SENS to significantly improve the accuracy of the calculated density of states at lower energies, whilst needing fewer replicas than in a standard NS simulation.

Landscape charts The results from nested sampling have been used to determine when different regions in configuration space become disconnected during the simulation [110]. A shape descriptor was used to group together configurations generated from the simulation to detect when different regions have become separated which was then used to generate energy landscape charts.

Integration Using nested sampling we can calculate an estimate of eq. (2.9) and its associated uncertainty [103]. This estimation can be done by calculating the first and second moments of $\mathcal{I}_{\Phi_0}[f]$. Suppose we have performed nested sampling and generated N^{NS} nested sampling points where the jth point was sampled with n_j^{NS} live points present. We can approximate the integral eq. (2.22) by the sum

$$\mathcal{I}_{\Phi_0}[f] \approx \Phi_0 \sum_j^{N^{\mathrm{NS}}} V_j^{\mathrm{NS}}(\Phi(V_{j+1}^{\mathrm{NS}}) - \Phi(V_j^{\mathrm{NS}})) = \Phi_0 \sum_j^{N^{\mathrm{NS}}} f_j(1-t_j)\prod_{k=1}^{j-1} t_k, \tag{2.27}$$

where $f_j = f(V_j^{\mathrm{NS}})$. Assuming the quadrature error is negligible [2] and as all the volume ratios are independent, the expected value of $\mathcal{I}_{\Phi_0}[f]$ can be calculated straightforwardly from eq. (2.36),

$$\mathbb{E}_{\mathrm{NS}}[\mathcal{I}_{\Phi_0}[f]] = \Phi_0 \sum_{j=1}^{N^{\mathrm{NS}}} f_j \frac{1}{n_j^{\mathrm{NS}}} \prod_{k=1}^{j} \frac{n_k^{\mathrm{NS}}}{n_k^{\mathrm{NS}}+1}. \tag{2.28}$$

$\mathbb{E}_{\mathrm{NS}}[\mathcal{I}_{\Phi_0}[f]^2]$ can be found by adapting the method used by Keeton [103] to estimate the uncertainty of estimates obtained using nested sampling runs with a fixed number of live points,

$$\mathbb{E}_{\mathrm{NS}}[\mathcal{I}_{\Phi_0}[f]^2] = \sum_{l=1}^{N^{\mathrm{NS}}} \left[\frac{2f_l}{n_l^{\mathrm{NS}}} \left(\prod_{k=1}^{l} \frac{n_k^{\mathrm{NS}}}{n_k^{\mathrm{NS}}+1} \right) \left(\sum_{j=1}^{l} \left(\frac{f_j}{n_j^{\mathrm{NS}}+1} \prod_{k'=1}^{j} \frac{n_{k'}^{\mathrm{NS}}+1}{n_{k'}^{\mathrm{NS}}+2} \right) \right) \right], \tag{2.29}$$

which can be calculated in $O(N^{\mathrm{NS}})$ operations. The statistical uncertainty for the estimate of $\mathcal{I}_{\Phi_0}[f]$ can be calculated as,

$$\sigma^2_{\mathcal{I}_{\Phi_0}[f]} = \mathbb{E}_{\mathrm{NS}}[\mathcal{I}_{\Phi_0}[f]^2] - \mathbb{E}_{\mathrm{NS}}[\mathcal{I}_{\Phi_0}[f]]^2. \tag{2.30}$$

2.5 Mathematical Methods

In this section we give a brief overview of the mathematical methods that are used in the rest of this work.

2.5.1 Beta distribution

To model a binomial event, such as flipping a coin, we need to be able to specify a probability distribution of the *probability* of obtaining heads, a probability of a *probability*. The probability of observing a certain number of heads, N^{heads}, after a fixed number of flips, N^{flips}, with a fixed probability of obtaining heads, p_{heads}, is modelled by the binomial distribution,

$$N^{\text{heads}} \sim \text{Bin}(N^{\text{flips}}, p_{\text{heads}}) \tag{2.31}$$

with probability,

$$\Pr(N^{\text{heads}}|N^{\text{flips}}, p^{\text{heads}}) = \binom{N^{\text{flips}}}{N^{\text{heads}}} {p_{\text{heads}}}^{N^{\text{heads}}} (1 - p_{\text{heads}})^{N^{\text{flips}} - N^{\text{heads}}}, \tag{2.32}$$

where $\binom{n}{m} = \frac{n!}{m!(n-m)!}$. The beta distribution is the *conjugate prior* to the binomial distribution (see section 2.5.3), so it is the most straightforward way to define a distribution over

$$p_{\text{heads}} \sim \mathcal{B}(\alpha_{\mathcal{B}}, \beta_{\mathcal{B}}), \tag{2.33}$$

with probability density,

$$\begin{aligned} \Pr(p_{\text{heads}}) &= \begin{cases} \dfrac{\Gamma(\alpha_{\mathcal{B}} + \beta_{\mathcal{B}})}{\Gamma(\alpha_{\mathcal{B}})\Gamma(\beta_{\mathcal{B}})} p_{\text{heads}}^{\alpha_{\mathcal{B}}-1} (1 - p_{\text{heads}})^{\beta_{\mathcal{B}}-1} & \text{when } 0 < p_{\text{heads}} < 1 \\ 0 & \text{otherwise} \end{cases} \\ &\equiv \mathcal{B}(p_{\text{heads}}|\alpha_{\mathcal{B}}, \beta_{\mathcal{B}}), \end{aligned} \tag{2.34}$$

where $\alpha_{\mathcal{B}}$, $\beta_{\mathcal{B}}$ are the shape parameters of the beta distribution. The normalisation constant can also be written as a beta function,

$$B(\alpha_{\mathcal{B}}, \beta_{\mathcal{B}}) = \frac{\Gamma(\alpha_{\mathcal{B}} + \beta_{\mathcal{B}})}{\Gamma(\alpha_{\mathcal{B}})\Gamma(\beta_{\mathcal{B}})}. \tag{2.35}$$

The moments of the beta distribution are

$$\int_0^1 t_{\mathcal{B}}^a (1-t_{\mathcal{B}})^{b'} \mathcal{B}(t_{\mathcal{B}}|\alpha_{\mathcal{B}},\beta_{\mathcal{B}})\,\mathrm{d}t_{\mathcal{B}} = \frac{\Gamma(\alpha_{\mathcal{B}}+\beta_{\mathcal{B}})}{\Gamma(\alpha_{\mathcal{B}})\Gamma(\beta_{\mathcal{B}})}\frac{\Gamma(\alpha_{\mathcal{B}}+a)\Gamma(\beta_{\mathcal{B}}+b')}{\Gamma(\alpha_{\mathcal{B}}+a+\beta_{\mathcal{B}}+b')}, \tag{2.36}$$

so in the case of eqs. (2.28) and (2.29) $t_k = t_{\mathcal{B}}$, $\alpha_{\mathcal{B}} = 1$ and $\beta_{\mathcal{B}} = n_k^{\mathrm{NS}}$, so we would find that $\mathbb{E}[t_{\mathcal{B}}] = n_k^{\mathrm{NS}}/(n_k^{\mathrm{NS}}+1)$, $\mathbb{E}[t_{\mathcal{B}}^2] = n_k^{\mathrm{NS}}/(n_k^{\mathrm{NS}}+2)$, and $\mathbb{E}[(1-t_{\mathcal{B}})] = 1/(n_k^{\mathrm{NS}}+1)$.

The beta distribution can also be used to model the *order statistics* of samples from the uniform distribution, if $n^{\mathcal{U}}$ samples have been drawn independently from the uniform distribution, then the $k^{\mathcal{U}}$th order statistic,

$$\mathcal{U}_{k^{\mathcal{U}}} \sim \mathcal{B}(k^{\mathcal{U}}, n^{\mathcal{U}} - k^{\mathcal{U}} + 1), \tag{2.37}$$

the distribution of the $k^{\mathcal{U}}$th highest value of $n^{\mathcal{U}}$ independent samples from the uniform distribution, which derives naturally from considering the binomial likelihood of observing $k^{\mathcal{U}}$ successes out of $n^{\mathcal{U}}$ trials with probability $\mathcal{U}_{k^{\mathcal{U}}}$. It is in this sense that NS uses the beta distribution as the set of live points can be modelled as uniformly distributed samples over $\Phi(V)$.

Fitting beta distributions

It is possible to fit a beta distribution, $\mathcal{B}(a_{\mathrm{fit}}, b_{\mathrm{fit}})$, to some other random variable, $t_{\mathrm{fit}} \sim q_{\mathrm{fit}}$, by matching the first and second moments as the beta distribution has only two degrees of freedom,

$$a_{\mathrm{fit}} = \mathbb{E}_{q_{\mathrm{fit}}}[t_{\mathrm{fit}}]\left(\frac{\mathbb{E}_{q_{\mathrm{fit}}}[t_{\mathrm{fit}}]\left(\mathbb{E}_{q_{\mathrm{fit}}}[t_{\mathrm{fit}}]-1\right)}{\mathbb{E}_{q_{\mathrm{fit}}}[t_{\mathrm{fit}}^2]-\mathbb{E}_{q_{\mathrm{fit}}}[t_{\mathrm{fit}}]^2}-1\right), \tag{2.38}$$

$$b_{\mathrm{fit}} = \left(\mathbb{E}_{q_{\mathrm{fit}}}[t_{\mathrm{fit}}]-1\right)\left(\frac{\mathbb{E}_{q_{\mathrm{fit}}}[t_{\mathrm{fit}}]\left(\mathbb{E}_{q_{\mathrm{fit}}}[t_{\mathrm{fit}}]-1\right)}{\mathbb{E}_{q_{\mathrm{fit}}}[t_{\mathrm{fit}}^2]-\mathbb{E}_{q_{\mathrm{fit}}}[t_{\mathrm{fit}}]^2}-1\right), \tag{2.39}$$

where $\mathbb{E}_{q_{\mathrm{fit}}}[t_{\mathrm{fit}}]$ is the first moment of t_{fit} and $\mathbb{E}_{q_{\mathrm{fit}}}[t_{\mathrm{fit}}^2]$ is the second moment of t_{fit} with respect to q_{fit}.

2.5.2 Dirichlet distribution

The binomial distribution and the beta distribution can be generalised to model multiple probabilities, for example, studying the rolling of a k^{die}-sided die. The results, $\boldsymbol{N}^{\mathrm{die}} = \{N_1^{\mathrm{die}}, ..., N_{k^{\mathrm{die}}}^{\mathrm{die}}\} \sim \mathrm{Mult}(\boldsymbol{p}^{\mathrm{die}})$ of rolling the die with probabilities, $\boldsymbol{p}^{\mathrm{die}} = \{p_1^{\mathrm{die}}, ..., p_{k^{\mathrm{die}}}^{\mathrm{die}}\}$ will be distributed according to the multinomial distribution,

$\boldsymbol{N}^{\text{die}} \sim \text{Mult}(\boldsymbol{p}^{\text{die}})$, with probability,

$$\text{Mult}(\boldsymbol{N}^{\text{die}}|\boldsymbol{p}^{\text{die}}) = \frac{\left(\sum_{j=1}^{k^{\text{die}}} N_j^{\text{die}}\right)!}{\prod_{k=1}^{N} N_k^{\text{die}}!} \prod_{k=1}^{N} {p_k^{\text{die}}}^{N_k^{\text{die}}}. \tag{2.40}$$

The conjugate prior of the multinomial distribution is the Dirichlet distribution, parametrised by an N-dimensional parameter vector, $\boldsymbol{\alpha}^{\text{die}} = (\alpha_1^{\text{die}}, ..., \alpha_N^{\text{die}})$. For $\boldsymbol{p}^{\text{die}} \sim \text{Dir}(\boldsymbol{\alpha}^{\text{die}})$ the probability distribution of $\boldsymbol{p}^{\text{die}}$ will be,

$$\Pr(\boldsymbol{p}^{\text{die}}|\boldsymbol{\alpha}^{\text{die}}) = \frac{\Gamma\left(\sum_{j=1}^{N} \alpha_j^{\text{die}}\right)}{\prod_{j=1}^{N} \Gamma(\alpha_j^{\text{die}})} \prod_{k=1}^{N} {p_k^{\text{die}}}^{\alpha_k^{\text{die}}-1}, \tag{2.41}$$

where $\sum_{j=1}^{N} p_j^{\text{die}} = 1$.

2.5.3 Bayesian inference

In Bayesian statistics, probabilities are used to encode beliefs about some phenomenon, whereas in frequentist statistics probabilities are used to model frequencies of events. At the heart of Bayesian statistics is Bayes' rule,

$$\Pr(M(\boldsymbol{\theta})|D) = \frac{\Pr(D|M(\boldsymbol{\theta}))\Pr(M(\boldsymbol{\theta}))}{\Pr(D|M)}, \tag{2.42}$$

which allows the *prior* belief/distribution, $\Pr\left(M(\boldsymbol{\theta})\right)$, for the parameters, $\boldsymbol{\theta}$, of some model, $M(\boldsymbol{\theta})$, to be updated to give a *posterior* belief/distribution, $\Pr(M|D(\boldsymbol{\theta}))$, for the parameters, given some observed data, D. This update is performed by taking the product of the *likelihood*, $\Pr(D|M(\boldsymbol{\theta}))$, of observing the data given the model with the prior. This product is normalised by the *evidence*, $\Pr(D|M)$, the probability of observing the data given all possible instances of the model, calculated by integrating the product of the likelihood and prior or *marginalising* over all possible parameters,

$$\Pr(D|M) = \int \Pr\left(D|M(\boldsymbol{\theta})\right) \Pr\left(M(\boldsymbol{\theta})\right) \, \mathrm{d}\boldsymbol{\theta}, \tag{2.43}$$

noting that we have dropped the model's argument of $\boldsymbol{\theta}$ as the evidence is not a function of the parameters.

When using the beta distribution to model p_{heads}, we can specify an uninformative prior on the coin toss, $p_{\text{heads}} \sim \mathcal{B}(\alpha_p, \beta_p)$, where α_p and β_p specify our prior belief of p_{heads}. For an uninformative prior $\alpha_p = \beta_p = 1/2$. We can use Bayes' rule to update

our belief of p_{heads},

$$\begin{aligned}\Pr(p_{\text{heads}}|N^{\text{heads}}, N^{\text{flips}}) &= \frac{\text{Bin}(N^{\text{heads}}|N^{\text{flips}}, p_{\text{heads}})\mathcal{B}(p_{\text{heads}}|\alpha_p, \beta_p)}{\Pr(N^{\text{heads}}|N^{\text{flips}})} \\ &= \mathcal{B}(p_{\text{heads}}|N^{\text{heads}} + \alpha_p, N^{\text{flips}} - N^{\text{heads}} + \beta_p), \end{aligned} \tag{2.44}$$

here we see why the parameters of the beta distribution are commonly viewed as pseudocounts, since they can be viewed as representing the number of observations of the event happening or not happening.

Model comparison

The evidence is useful as it allows different models to be compared. Given two models, M_1 and M_2, and some data, the probability of the hypothesis that the first model is correct, $\mathcal{H}_1$, can be calculated as,

$$\Pr(\mathcal{H}_1) = \frac{\Pr(M_1|D)\Pr(M_1)}{\Pr(M_1|D)\Pr(M_1) + \Pr(M_2|D)\Pr(M_2)} = \frac{K^{\text{BF}}}{1 + K^{\text{BF}}}, \tag{2.45}$$

where the relative evidence between the two models is known as the *Bayes factor*,

$$K^{\text{BF}} = \frac{\Pr(M_1|D)}{\Pr(M_2|D)}\frac{\Pr(M_1)}{\Pr(M_2)}, \tag{2.46}$$

and $\Pr(M_1)$ and $\Pr(M_2)$ are the prior belief of whether $\mathcal{H}_1$ or $\mathcal{H}_2$ is true. If *a priori* both models are viewed equally likely then $\Pr(M_1) = \Pr(M_2) = 1/2$.

When $K^{\text{BF}} > 1$, $\mathcal{H}_1$ is more likely. Conversely if $K^{\text{BF}} < 1$, $\mathcal{H}_2$ is more likely given the data. In the next section it will be shown how to compare binomial distributions using Bayesian model comparison.

Comparing binomial distributions

Consider an example where we want to compare examination pass rates, p_a and p_b, between two different departments, a and b. Suppose we observe that there are n_a^{pass} passes and n_a^{fail} fails from department a and the equivalent for b, and $n_{a/b}^{\text{students}} = n_{a/b}^{\text{pass}} + n_{a/b}^{\text{fail}}$. We are interested in testing the hypothesis that the pass rates are the same, $\mathcal{H}_0 : p_a = p_b$, or different, $\mathcal{H}_1 : p_a \neq p_b$. We can compare these hypotheses by Bayesian model comparison, from section 2.5.3. So we can calculate

the evidence of model 1,

$$\begin{aligned}
&\Pr(p_a = p_b | n_a^{\text{pass}}, n_a^{\text{fail}}, n_b^{\text{pass}}, n_b^{\text{fail}}) \\
&\quad = \int \text{Bin}(n_a^{\text{pass}} | n_a^{\text{students}}, p_a) \text{Bin}(n_b^{\text{pass}} | n_b^{\text{students}}, p_a) \mathcal{B}(p_a | \alpha^{\text{pass}}, \beta^{\text{pass}}) \, \mathrm{d}p_a \\
&\quad = \binom{n_a^{\text{students}}}{n_a^{\text{pass}}} \binom{n_b^{\text{students}}}{n_b^{\text{pass}}} B(n_a^{\text{pass}} + n_b^{\text{pass}} + \alpha^{\text{pass}}, n_a^{\text{fail}} + n_b^{\text{fail}} + \alpha^{\text{fail}}), \quad (2.47)
\end{aligned}$$

and 2,

$$\begin{aligned}
&\Pr(p_a \neq p_b | n_a^{\text{pass}}, n_a^{\text{fail}}, n_b^{\text{pass}}, n_b^{\text{fail}}) \\
&= \iint \text{Bin}(n_a^{\text{pass}} | n_a^{\text{students}}, p_a) \text{Bin}(n_b^{\text{pass}} | n_b^{\text{students}}, p_b) \mathcal{B}(p_a | \alpha^{\text{pass}}, \beta^{\text{pass}}) \mathcal{B}(p_b | \alpha^{\text{pass}}, \beta^{\text{pass}}) \, \mathrm{d}p_a \, \mathrm{d}p_b \\
&\quad = \binom{n_a^{\text{students}}}{n_a^{\text{pass}}} \binom{n_b^{\text{students}}}{n_b^{\text{pass}}} B(n_a^{\text{pass}} + \alpha^{\text{pass}}, n_a^{\text{fail}} + \alpha^{\text{fail}}) B(n_b^{\text{pass}} + \alpha^{\text{pass}}, n_b^{\text{fail}} + \alpha^{\text{fail}}), \quad (2.48)
\end{aligned}$$

where α^{pass} and α^{fail} encode the prior pseudocounts on the pass rate. The Bayes factor comparing the two hypotheses can be calculated

$$\begin{aligned}
&K_{\mathcal{B}}^{\text{BF}}(n_a^{\text{pass}}, n_a^{\text{fail}}, n_b^{\text{pass}}, n_b^{\text{fail}}) \\
&\qquad = \frac{B(n_a^{\text{pass}} + n_b^{\text{pass}} + \alpha^{\text{pass}}, n_a^{\text{fail}} + n_b^{\text{fail}} + \alpha^{\text{fail}})}{B(n_a^{\text{pass}} + \alpha^{\text{pass}}, n_a^{\text{fail}} + \alpha^{\text{fail}}) B(n_b^{\text{pass}} + \alpha^{\text{pass}}, n_b^{\text{fail}} + \alpha^{\text{fail}})}. \quad (2.49)
\end{aligned}$$

In addition we can generalise eq. (2.47) to more than two departments, with pass rates, $\boldsymbol{p}^{\text{pass}} = \{p_1, ..., p_{N_{\text{departments}}}\}$, observed passes counts $\boldsymbol{n}^{\text{pass}} = \{n_1^{\text{pass}}, ..., n_{N_{\text{departments}}}^{\text{pass}}\}$ and fail counts $\boldsymbol{n}^{\text{fail}} = \{n_1^{\text{fail}}, ..., n_{N_{\text{departments}}}^{\text{fail}}\}$ then we can calculate the Bayes factor for them having the same pass rate,

$$\begin{aligned}
&\Pr\left((p_j = p_k) \forall p_j, p_k \in \boldsymbol{p}^{\text{pass}} | \boldsymbol{n}^{\text{pass}}, \boldsymbol{n}^{\text{fail}}\right) \\
&\quad = B\left(\alpha^{\text{pass}} + \sum_{j=1}^{N_{\text{departments}}} n_j^{\text{pass}}, \alpha^{\text{fail}} + \sum_{j=1}^{N_{\text{departments}}} n_j^{\text{fail}}\right) \prod_{j=1}^{N_{\text{departments}}} \binom{n_j^{\text{pass}} + n_j^{\text{fail}}}{n_j^{\text{pass}}}. \quad (2.50)
\end{aligned}$$

3 Kernel Correlation Alignment

In this chapter we describe the FASTOVERLAP algorithm, a variant of kernel correlation [55] based alignment methods. It uses a fast Fourier transform (FFT) or discrete $SO(3)$ Fourier transform (SOFT) [58] to find deterministically the maximum correlation/overlap between kernel/density representations of either periodic structures or clusters. FASTOVERLAP is also related to the methods proposed by Bartók et al. [67], Ferré et al. [68] and Makadia et al. [57].

In sections 3.1 and 3.3 we demonstrate how the kernel correlation/overlap can be used to estimate the global minimum RMSD efficiently for structures that are reasonably close for periodic systems and isolated clusters of atoms; for alignments with large RMSDs the inherent approximations will break down. The maximum correlation displacement/rotation can be used as a starting point for the PERMDIST algorithm (see section 2.3.2). The run time for FASTOVERLAP is not very sensitive to the RMSD for a given system.

A comparison of the performance of this algorithm compared to existing methods is given in chapter 5.

3.1 RMSD estimation by Gaussian overlap

Under certain circumstances it is possible to estimate the RMSD between two closely aligned structures by calculating the overlap integral of a set of Gaussian functions centred on the atomic coordinates. Consider a pair of atoms, specified by two Gaussian kernels with width σ_G, centred at positions $\boldsymbol{r}_0$ and $\boldsymbol{r}_1$,

$$\rho_0(\boldsymbol{r}) = \exp\left(-\frac{|\boldsymbol{r}-\boldsymbol{r}_0|^2}{2\sigma_G^2}\right), \qquad \rho_1(\boldsymbol{r}) = \exp\left(-\frac{|\boldsymbol{r}-\boldsymbol{r}_1|^2}{2\sigma_G^2}\right). \tag{3.1}$$

The overlap integral of these two Gaussians is,

$$\begin{aligned}\iiint \rho_0(\boldsymbol{r})\rho_1(\boldsymbol{r})\mathrm{d}\boldsymbol{r} &= \iiint \exp\left(-\frac{|\boldsymbol{r}_0-\boldsymbol{r}_1|^2}{4\sigma_G^2}\right)\exp\left(-\frac{|\boldsymbol{r}-\frac{\boldsymbol{r}_0+\boldsymbol{r}_1}{2}|^2}{\sigma_G^2}\right)\mathrm{d}\boldsymbol{r}\\ &= \exp\left(-\frac{|\boldsymbol{r}_0-\boldsymbol{r}_1|^2}{4\sigma_G^2}\right)(\pi\sigma_G^2)^{3/2}.\end{aligned} \tag{3.2}$$

When $|\boldsymbol{r}_0-\boldsymbol{r}_1| \ll \sigma_G$ we can approximate eq. (3.2) as,

$$\iiint \rho_0(\boldsymbol{r})\rho_1(\boldsymbol{r})\mathrm{d}\boldsymbol{r} \approx (\pi\sigma_G^2)^{3/2} - (\pi\sigma_G^2)^{3/2}\frac{|\boldsymbol{r}_0-\boldsymbol{r}_1|^2}{4\sigma_G^2} + o\left(\left(\frac{|\boldsymbol{r}_0-\boldsymbol{r}_1|}{\sigma_G}\right)^4\right). \tag{3.3}$$

Hence the overlap integral is proportional to the squared displacement between the atoms when they are close relative to σ_G. We can extend this result to estimate the RMSD of two closely aligned periodic structures, p and q, by defining the density functions

$$\rho_p(\boldsymbol{r}) = \sum_{\boldsymbol{l}\in\boldsymbol{L}}\sum_{j=1}^{N}\exp\left(\frac{|\boldsymbol{r}-\boldsymbol{r}_j^p+\boldsymbol{l}|^2}{2\sigma_G^2}\right), \rho_q(\boldsymbol{r}-\boldsymbol{d}) = \sum_{\boldsymbol{l}\in\boldsymbol{L}}\sum_{j=1}^{N}\exp\left(\frac{|\boldsymbol{r}-\boldsymbol{r}_j^q-\boldsymbol{l}-\boldsymbol{d}|^2}{2\sigma_G^2}\right). \tag{3.4}$$

Recall that $\boldsymbol{d}$ is the global displacement vector and $\boldsymbol{l}$ is a particular lattice vector. Using eq. (3.2) we can calculate the overlap integral or *kernel correlation* of these densities

$$\Omega^{pq}(\boldsymbol{d}) = \iiint_0^L \rho_p(\boldsymbol{r})\rho_q(\boldsymbol{r}-\boldsymbol{d})\mathrm{d}\boldsymbol{r} = (\pi\sigma_G^2)^{3/2}\sum_{\boldsymbol{l}\in\boldsymbol{L}}\sum_{j=1}^{N}\sum_{j'=1}^{N}\exp\left(-\frac{|\boldsymbol{r}_j^p-\boldsymbol{r}_{j'}^q-\boldsymbol{l}-\boldsymbol{d}|^2}{4\sigma_G^2}\right). \tag{3.5}$$

We note that eq. (3.5) is invariant to permutations. If we then assume that $\sigma_G \ll r_{\text{sep}}$, where r_{sep} is the minimum atomic separation, then for displacement vector $\boldsymbol{d}$, permutation matrix $\boldsymbol{P}$ and lattice vectors $\boldsymbol{L}$ that minimise the RMSD we can approximate the integral above as

$$\Omega^{pq}(\boldsymbol{d}) \approx (\pi\sigma_G^2)^{3/2}\sum_{j=1}^{N}\exp\left(-\frac{|\boldsymbol{r}_j^p-\sum_{j'=1}^{N}\boldsymbol{P}_{jj'}(\boldsymbol{r}_{j'}^q-\boldsymbol{l}_{j'}-\boldsymbol{d})|^2}{4\sigma_G^2}\right). \tag{3.6}$$

If we also assume that $|\boldsymbol{r}_j^p - \sum_{j'=1}^N \boldsymbol{P}_{jj'}(\boldsymbol{r}_{j'}^q - \boldsymbol{l}_{j'} - \boldsymbol{d})| \ll \sigma_G$ for all j, so the structures are relatively similar, then if $\boldsymbol{d}_\mathrm{m} = \arg\max \Omega^{pq}(\boldsymbol{d})$

$$\begin{aligned}\Omega^{pq}(\boldsymbol{d}_\mathrm{m}) &\approx (\pi\sigma_G^2)^{3/2} \sum_{j=1}^N \left(1 - \frac{|\boldsymbol{r}_j^p - \sum_{j'=1}^N \boldsymbol{P}_{jj'}(\boldsymbol{r}_{j'}^q - \boldsymbol{l}_{j'} - \boldsymbol{d}_\mathrm{m})|^2}{4\sigma_G^2}\right) \\ &= N(\pi\sigma_G^2)^{3/2} - \frac{\pi^{3/2}\sigma_G^{-1/2}\sqrt{N}}{4}\,\mathrm{RMSD}(p,q). \end{aligned} \tag{3.7}$$

Hence the global maximum of $\Omega^{pq}(\boldsymbol{d})$ corresponds to the displacement that gives the minimum RMSD if the structures can be aligned sufficiently closely. Once the displacement is known, the corresponding optimal permutation matrix can be calculated using the Hungarian algorithm [47] or shortest augmenting path algorithm [39].

The choice of σ_G is important for determining the accuracy of this method, If σ_G is set too small then the approximations only hold true for very closely aligned systems, while if σ_G is too large then the value of the integral is no longer determined by the nearest neighbours at optimal alignment. In practice we found that setting σ_G to be around $1/3$ of the equilibrium pair separation produced good results over the widest range of structures.

We can see this value as being a compromise between having too small a kernel width, which would mean that the algorithm would only work for very similar structures, due to the limited overlap, versus too large a kernel width, which would make the overall density too homogeneous.

3.2 Global optimisation of the overlap integral

To identify the global maximum of $\Omega^{pq}(\boldsymbol{d})$ efficiently we use Parseval's theorem, which states that for functions with Fourier series

$$\rho_p(\boldsymbol{r}) = \sum_{\boldsymbol{k}\in\boldsymbol{K}} c_{\boldsymbol{k}}^p e^{i\boldsymbol{k}\cdot\boldsymbol{r}}, \qquad \rho_q(\boldsymbol{r}-\boldsymbol{d}) = \sum_{\boldsymbol{k}\in\boldsymbol{K}} c_{\boldsymbol{k}}^q e^{i\boldsymbol{k}\cdot\boldsymbol{r}} e^{-i\boldsymbol{k}\cdot\boldsymbol{d}}, \tag{3.8}$$

where $\boldsymbol{K}$ is the set of allowed wavevectors, so if $\boldsymbol{k} \in \boldsymbol{K}$, then $\boldsymbol{k} = 2\pi\boldsymbol{n}/L$, for $\boldsymbol{n} \in \mathbb{Z}^3$, that

$$\Omega^{pq}(\boldsymbol{d}) = \iiint_0^L \rho_p(\boldsymbol{r})\rho_q(\boldsymbol{r}-\boldsymbol{d})\mathrm{d}\boldsymbol{r} = \frac{1}{L^3}\sum_{\boldsymbol{k}\in\boldsymbol{K}} c_{\boldsymbol{k}}^p c_{\boldsymbol{k}}^{q\,*} e^{i\boldsymbol{k}\cdot\boldsymbol{d}}, \tag{3.9}$$

where z^* indicates the complex conjugate of z. The Fourier series coefficients can easily be calculated by treating the structure as a sum of delta functions at the atomic coordinates convolved with a Gaussian function of width σ_G

$$c^p_{\boldsymbol{k}} = e^{-\frac{1}{2}|\boldsymbol{k}|^2\sigma_G^2} \sum_{j=1}^{N} e^{-i\boldsymbol{k}\cdot\boldsymbol{r}^p_j}, \tag{3.10}$$

$$c^q_{\boldsymbol{k}} = e^{-\frac{1}{2}|\boldsymbol{k}|^2\sigma_G^2} \sum_{j=1}^{N} e^{-i\boldsymbol{k}\cdot\boldsymbol{r}^q_j} \tag{3.11}$$

For brevity we define the structure factors,

$$d^p_{\boldsymbol{k}} = \sum_{j=1}^{N} e^{-i\boldsymbol{k}\cdot\boldsymbol{r}^p_j}, \quad d^q_{\boldsymbol{k}} = \sum_{j=1}^{N} e^{-i\boldsymbol{k}\cdot\boldsymbol{r}^q_j}. \tag{3.12}$$

Here we note that the magnitude of the Fourier coefficients decays exponentially, so we can specify a cutoff wavevector, $|\boldsymbol{k}_{\max}| \gg 1/\sigma_G$ above which we do not need to calculate them. The value of the cutoff will determine the numerical accuracy of the calculation. We find

$$\Omega^{pq}(\boldsymbol{d}) \approx \frac{1}{L^3} \sum_{\boldsymbol{k}\in\boldsymbol{K}}^{|\boldsymbol{k}|<|\boldsymbol{k}_{\max}|} e^{-|\boldsymbol{k}|^2\sigma_G^2} d^p_{\boldsymbol{k}} d^{q\,*}_{\boldsymbol{k}} e^{i\boldsymbol{k}\cdot\boldsymbol{d}}. \tag{3.13}$$

This expression is simply the Fourier series representation of $\Omega^{pq}(\boldsymbol{d})$, so in order to calculate the maximum value of $\Omega^{pq}(\boldsymbol{d})$ we can perform the fast inverse Fourier transform (FFT) on the coefficients to calculate an array of values of $\Omega^{pq}(\boldsymbol{d})$. The value of $\boldsymbol{d}$ that maximises $\Omega^{pq}(\boldsymbol{d})$ can be found by fitting a quadratic or Gaussian to the points close to the maximum value of the array in the three axes, or by a local maximisation of eq. (3.13).

3.2.1 Width of kernel

From eq. (3.13) we see that the convolution of a second Gaussian kernel of width σ_{G_1} with $\Omega^{pq}(\boldsymbol{d})$ simply corresponds to the selection of a larger width $\sigma_{G_2} = \sqrt{\sigma_G^2 + 2\sigma_{G_1}^2}$ for the original Gaussian kernel.

3.2.2 Algorithmic complexity

The efficiency is primarily determined by the number of $\boldsymbol{k}$ values that need to be computed for the Fourier series representation of $\Omega^{pq}(\boldsymbol{d})$ to converge. The number of $\boldsymbol{k}$ values is proportional to the ratio L/σ_G, while $\sigma_G \propto r_{\text{sep}}$, the minimum atomic separation. If we assume that the atomic density is approximately uniform then $r_{\text{sep}} \propto (L/N)^{1/3}$, so the total number of $\boldsymbol{k}$ values will be proportional to $(L/\sigma_G)^3 \propto N$. Hence calculating the Fourier coefficients will be $O(N^2)$. The FFT will be $O(N \log N)$, so the total complexity of finding the optimal displacement will be $O(N^2)$. Solving the assignment problem to find the correct permutation is also $O(N^2)$, so the overall complexity of the alignment is still $O(N^2)$.

Much of the computational cost is associated with the calculation of the Fourier coefficients in eqs. (3.10) and (3.11). When aligning a large database of structures these coefficients can be precalculated, providing a significant performance improvement.

3.2.3 Limitations

This algorithm will fail to find the global RMSD when the difference between the structures is dominated by pairs of atoms that are a large distance apart, as the contribution of these pairs to the overlap integral is small and so will not be optimised. In this case it is possible that the RMSD is not a particularly useful measure of similarity, and so other methods for comparing and aligning structures may be more relevant.

3.3 Minimising RMSD for clusters

We can perform a very similar analysis for isolated clusters of atoms. For structures p and q, with atomic coordinates $\boldsymbol{R}^p$ and $\boldsymbol{R}^q$, and centroids already shifted to the origin, we seek

$$\text{RMSD}(p,q) = \frac{1}{\sqrt{N}} \min_{\alpha,\beta,\gamma,\boldsymbol{P}} |\boldsymbol{R}^p - \boldsymbol{P}\boldsymbol{R}^q\boldsymbol{M}(\alpha,\beta,\gamma)^\top|, \tag{3.14}$$

where $\boldsymbol{M}$ is the block diagonal coordinate rotation matrix containing N copies of $\boldsymbol{m}$,

$$\boldsymbol{m}(\alpha,\beta,\gamma) = \begin{bmatrix} \cos\gamma & \sin\gamma & 0 \\ -\sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \begin{bmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{3.15}$$

$\boldsymbol{m}(\alpha,\beta,\gamma)$ is a rotation matrix parametrised by the Euler angles, α, β and γ, representing three successive rotations around the z, y and then the z axis. We can redefine the overlap integral in eq. (2.8) for rotations:

$$\begin{aligned} \Omega^{pq}(\alpha,\beta,\gamma) &= \iiint_{\infty}^{-\infty} \rho_p(\boldsymbol{r})\rho_q(\boldsymbol{m}(\alpha,\beta,\gamma)\boldsymbol{r})\,\mathrm{d}\boldsymbol{r} \\ &= (\pi\sigma_G^2)^{3/2} \sum_{j=1}^{N}\sum_{j'=1}^{N} \exp\left(-\frac{|\boldsymbol{r}_j^p - \boldsymbol{m}(\alpha,\beta,\gamma)\boldsymbol{r}_{j'}^q|^2}{\sigma_G^2}\right), \end{aligned} \tag{3.16}$$

and similarly, as in eq. (3.7), for systems where every pair of aligned atoms is separated by much less than σ_G and if $(\alpha_\mathrm{m},\beta_\mathrm{m},\gamma_\mathrm{m}) = \arg\max \Omega^{pq}(\alpha,\beta,\gamma)$ then

$$\Omega^{pq}(\alpha_\mathrm{m},\beta_\mathrm{m},\gamma_\mathrm{m}) \approx N(\pi\sigma_G^2)^{3/2} - \pi^{3/2}\sigma_G^{-1/2}\sqrt{N}\,\mathrm{RMSD}(p,q). \tag{3.17}$$

To evaluate $\Omega^{pq}(\alpha,\beta,\gamma)$ efficiently we follow the method developed to calculate SOAP similarity kernels by Bartók et al. [67] and De et al. [41] based on expanding Gaussian functions by a modified form of the Rayleigh expansion [111],

$$\exp\left(-\frac{|\boldsymbol{r}-\boldsymbol{r}_j|^2}{2\sigma_G^2}\right) = 4\pi\exp\left(-\frac{r^2+r_j^2}{2\sigma_G^2}\right)\sum_{l=0}^{\infty}\sum_{m=-l}^{l} i_l\left(\frac{rr_j}{\sigma_G^2}\right) Y_l^m(\hat{\boldsymbol{r}})Y_l^m(\hat{\boldsymbol{r}}_j)^*, \tag{3.18}$$

where $r = |\boldsymbol{r}|$ and $\hat{\boldsymbol{r}} = \boldsymbol{r}/r$. $Y_l^m(\hat{\boldsymbol{r}})$ is the value of the spherical harmonic with degree l and order m evaluated at a point on the unit sphere, $\hat{\boldsymbol{r}}$, $i_l(r)$ are modified spherical Bessel functions of the first kind. Using this relationship we can express the two densities as

$$\begin{aligned} &\rho(\boldsymbol{m}(\alpha,\beta,\gamma)^T\boldsymbol{r})_q = \\ &\quad\sum_j\sum_{l=0}^{\infty}\sum_{m,m'=-l}^{l} 4\pi\exp\left(-\frac{r^2+r_j^{q2}}{2\sigma_G^2}\right) i_l\left(\frac{rr_j^q}{\sigma_G^2}\right) D^l_{mm'}(\alpha,\beta,\gamma)Y_l^m(\hat{\boldsymbol{r}})Y_l^m(\hat{\boldsymbol{r}}_j^q)^* \end{aligned} \tag{3.19}$$

$$\rho(\boldsymbol{r})_p = \sum_j \sum_{l=0}^{\infty} \sum_{m=-l}^{l} 4\pi \exp\left(-\frac{r^2 + r_j^{p2}}{2\sigma_G^2}\right) i_l\left(\frac{r r_j^p}{\sigma_G^2}\right) Y_l^m(\hat{\boldsymbol{r}}) Y_l^m(\hat{\boldsymbol{r}}_j^p)^*, \tag{3.20}$$

where $D^l_{mm'}(\alpha,\beta,\gamma)$ are the coefficients of the Wigner-D matrix that transforms the coefficients of the spherical harmonics by a rotation of α, β, γ, with $\sum_{m'} D^l_{mm'} Y_l^{m'}(\hat{\boldsymbol{r}}) = Y_l^m(\boldsymbol{m}\hat{\boldsymbol{r}})$. For brevity we have dropped the arguments, so $D^l_{mm'} \equiv D^l_{mm'}(\alpha,\beta,\gamma)$ and $\boldsymbol{m} \equiv \boldsymbol{m}(\alpha,\beta,\gamma)$, and abbreviated the sums, as $\sum_{l,m}\{\} \equiv \sum_{l=0}^{\infty}\sum_{m=-l}^{l}\{\}$. Substituting eqs. (3.19) and (3.20) into eq. (3.16)

$$\begin{aligned}\Omega^{pq}(\alpha,\beta,\gamma) = &\sum_{j,j'}\sum_{l,m}\sum_{l',m',m''} (4\pi)^2 Y_l^m(\hat{\boldsymbol{r}}_j^p) D^{l'}_{m'm''} Y_{l'}^{m''}(\hat{\boldsymbol{r}}_{j'}^q)^* \exp\left(-\frac{r_j^{p2} + r_j^{q2}}{2\sigma_G^2}\right)\times \\ &\int_0^\infty \exp\left(-\frac{r^2}{\sigma_G^2}\right) i_l\left(\frac{r r_j^p}{\sigma_G^2}\right) i_l\left(\frac{r r_{j'}^q}{\sigma_G^2}\right) r^2 \,\mathrm{d}r \int Y_l^m(\hat{\boldsymbol{r}})^* Y_{l'}^{m''}(\hat{\boldsymbol{r}}) \,\mathrm{d}\hat{\boldsymbol{r}}.\end{aligned} \tag{3.21}$$

The integrals can be evaluated analytically,

$$\int_0^\infty \exp\left(-\frac{r^2}{\sigma_G^2}\right) i_l\left(\frac{r r_j^p}{\sigma_G^2}\right) i_l\left(\frac{r r_{j'}^q}{\sigma_G^2}\right) r^2 \,\mathrm{d}r = \frac{\sqrt{\pi}\sigma_G^3}{4} i_l\left(\frac{r_j^p r_{j'}^q}{2\sigma_G^2}\right) \exp\left(\frac{r_j^{p2} + r_{j'}^{q\,2}}{4\sigma_G^2}\right), \tag{3.22}$$

$$\int Y_l^m(\hat{\boldsymbol{r}})^* Y_{l'}^{m''}(\hat{\boldsymbol{r}}) \,\mathrm{d}\hat{\boldsymbol{r}} = \delta_{ll'}\delta_{mm''}. \tag{3.23}$$

Hence

$$\Omega^{pq}(\alpha,\beta,\gamma) = \sum_{j,j'}\sum_{l,m,m'} 4\pi^{5/2}\sigma_G^3 \, Y_l^m(\hat{\boldsymbol{r}}_j^p) Y_l^{m'}(\hat{\boldsymbol{r}}_{j'}^q)^* \exp\left(-\frac{r_j^{p2} + r_{j'}^{q\,2}}{4\sigma_G^2}\right) i_l\left(\frac{r_j^p r_{j'}^q}{2\sigma_G^2}\right) D^l_{mm'}. \tag{3.24}$$

Now we can calculate the Fourier coefficients of the overlap integral as

$$\Omega^{pq}(\alpha,\beta,\gamma) = \sum_{l=0}^{l_{\max}} \sum_{m,m'=-l}^{l} I^l_{mm'} D^l_{mm'}(\alpha,\beta,\gamma), \tag{3.25}$$

where

$$I^l_{mm'} = \sum_{j,j'} 4\pi^{5/2}\sigma_G^3 \, Y_l^m(\hat{\boldsymbol{r}}_j^p) Y_l^{m'}(\hat{\boldsymbol{r}}_{j'}^q)^* \exp\left(-\frac{r_j^{p2} + r_{j'}^{q\,2}}{4\sigma_G^2}\right) i_l\left(\frac{r_j^p r_{j'}^q}{2\sigma_G^2}\right). \tag{3.26}$$

To evaluate the integral numerically we truncate the the sum at a maximum angular momentum degree, $l_{\max}$. To find the maximum value of Ω^{pq} we can use SOFT to perform a $SO(3)$ Fourier synthesis on $I^l_{mm'}$, which can be achieved in $O(\Delta_\theta^3 \log^2 \Delta_\theta)$ operations, where $2\Delta_\theta = l_{\max}$ is the angular resolution or bandwidth of SOFT [58]. Most implementations of SOFT (including ours) have $O(\Delta_\theta^4)$ computational complexity.

Calculating the full sum over all j and j' is computationally expensive, but the number of terms required can be reduced by omitting contributions where $|r^p_j - r^q_{j'}| \gg \sigma_G$. Assuming a uniform density of points, this observation means the number of terms in the sum will reduce from N^2 to $N^{5/3}$. Hence, calculating the Fourier coefficients requires $O(N^{5/3} l^3_{\max})$ operations.

The result given by this Fourier synthesis can be refined by performing a local minimisation of eq. (3.25), as the gradients of $D^l_{mm'}(\alpha, \beta, \gamma)$ can be calculated analytically or by fitting a set of Gaussian peaks to the output data, and the location of these peaks can be used as an initial starting point for the PERMDIST algorithm (see section 2.3.2).

3.3.1 Harmonic basis

The calculation of the cross terms in eq. (3.22) makes the overlap method more expensive, and as a result, it is harder to evaluate .To improve efficiency we can project eqs. (3.19) and (3.20) onto an orthogonal radial basis, which we can generate from the isotropic three-dimensional quantum harmonic oscillator (referred to here as the harmonic basis). Expressing eq. (3.19) in the harmonic basis we obtain,

$$\rho_p(\boldsymbol{r}) = \sum_{n,l,m} c^p_{nlm} N_{nl} r^l \exp\left(-\frac{r^2}{2r_0^2}\right) L_n^{l+1/2}\left(\frac{r^2}{r_0^2}\right) Y_l^m(\hat{\boldsymbol{r}}) = \sum_{n,l,m} c^p_{nlm} g_{nl}(r) Y_l^m(\hat{\boldsymbol{r}}), \tag{3.27}$$

where $L^m_n(r)$, are generalised Laguerre polynomials and

$$N_{nl} = \sqrt{\frac{2n!}{r_0^{2l+3}\Gamma(3/2+n+l)}} \tag{3.28}$$

is the normalisation constant, such that

$$\iiint g_{nl}(r) Y_l^m(\hat{\boldsymbol{r}})^* g_{n'l'}(r) Y_{l'}^{m'}(\hat{\boldsymbol{r}}) \,\mathrm{d}\boldsymbol{r} = \delta_{nn'}\delta_{ll'}\delta_{mm'}. \tag{3.29}$$

The coefficients of eq. (3.27) can be obtained using eqs. (3.18) and (3.29),

$$\begin{aligned} c^p_{nlm} &= \int \rho_p(\boldsymbol{r}) g_{nl}(r) Y_l^m(\hat{\boldsymbol{r}}) \, \mathrm{d}\boldsymbol{r} \\ &= 4\pi \sum_{j=1}^{N} Y_l^m(\hat{\boldsymbol{r}}_j^p)^* \int_0^\infty g_{nl}(r) \exp\left(-\frac{r^2 + r_j^{p2}}{2\sigma_G^2}\right) i_l\left(\frac{r r_j^p}{\sigma_G^2}\right) r^2 \, \mathrm{d}r \\ &= \sum_{j=1}^{N} Y_l^m(\hat{\boldsymbol{r}}_j^p)^* d_{nl}(r_j^p). \end{aligned} \tag{3.30}$$

For $n = 0$ we have the following analytic result

$$d_{0,l} = 4\sigma_G^3 \sqrt{\frac{\pi^3}{r_j^3 \Gamma\left(l + \frac{3}{2}\right)}} \left(\frac{r_0 r_j}{r_0^2 + \sigma_G^2}\right)^{l+\frac{3}{2}} \exp\left(-\frac{r_j^2}{2\left(r_0^2 + \sigma_G^2\right)}\right). \tag{3.31}$$

To evaluate eq. (3.30) for larger values of n we can use the following recurrence relations

$$n L_n^{l+1/2}(x) = (n + l + 1/2) L_{n-1}^{l+1/2}(x) - x L_{n-1}^{l+3/2}, \tag{3.32}$$

$$L_{n-1}^{l+3/2}(x) = L_{n-1}^{l+5/2}(x) - L_{n-2}^{l+5/2}(x), \tag{3.33}$$

$$i_l(x) = \frac{2l+3}{x} i_{l+1}(x) + i_{l+2}(x). \tag{3.34}$$

Hence we obtain a recurrence relation for the integral $d_{n,l}(r_j)$ (where for brevity we drop the argument of $d_{n,l}(r_j)$, so $d_{n,l}(r_j) \equiv d_{n,l}$ and noting that $d_{-1,l} = 0$)

$$\begin{aligned} 0 = &-\sqrt{\frac{n-1}{n}} \, d_{n-2,l+2} - \sqrt{\frac{n+l+1/2}{n}} \, d_{n-1,l} + \frac{(2l+3)\sigma_G^2}{r_j r_0 \sqrt{n}} d_{n-1,l+1} + \\ &\sqrt{\frac{n+l+3/2}{n}} \, d_{n-1,l+2} + d_{n,l}. \end{aligned} \tag{3.35}$$

Unfortunately, evaluating the forward recurrence of eq. (3.35) is numerically unstable for large n and l, and attempting to stabilise the recursion results in ill-conditioned matrices. This problem limits the ratio of maximum extent of the structure to the width of the kernel, $\max(r_j) < 10\sigma_G$. Within this regime Fourier coefficients of the

overlap integral can be calculated as,

$$\begin{aligned}\Omega^{pq}(\alpha,\beta,\gamma) &= \sum_{n,l,m}\sum_{n',l',m'}\sum_{m''}\iiint (c^{p}_{n,l,m})^{*} g_{nl}(r) Y_l^m(\hat{\boldsymbol{r}})^{*}\, c^{p}_{n',l',m'} g_{n'l'}(r) D^{l'}_{m''m'} Y_{l'}^{m''}(\hat{\boldsymbol{r}})\mathrm{d}\boldsymbol{r} \\ &= \sum_{n}(c^{p}_{n,l,m})^{*} c^{q}_{n,l,m'} D^{l}_{mm'}. \end{aligned} \tag{3.36}$$

This result can be used to perform an alignment by following the method in section 3.3. When aligning a large database of structures the algorithm can be made more efficient by precalculating the harmonic basis coefficients. This precalculation will come at a slight cost of accuracy in eq. (3.36), as only a fixed number of radial basis functions can be considered, whereas the numerical calculation for eq. (3.26) is exact.

Computational complexity Calculating eq. (3.36) requires specifying a cutoff angular momentum order, as discussed in section 3.3, and a cutoff harmonic basis order, such that $n \leq n_{\text{max}}$ and $n_{\text{max}} \propto N^{2/3}$. Hence the total complexity associated with calculating the harmonic basis coefficients will be approximately $O(N^{5/3} l^3_{\text{max}})$.

3.3.2 Spherical Fourier transforms

Generalising the Fourier transform to spherical coordinates gives an alternative method to obtain the SO(3) Fourier coefficients. For a function $f(\boldsymbol{r})$, the Fourier transform and Fourier synthesis can be defined

$$F(\boldsymbol{k}) = \mathcal{F}[f(\boldsymbol{r})]_{\boldsymbol{k}} = \iiint f(\boldsymbol{r}) \exp(-i\boldsymbol{k}\cdot\boldsymbol{r})\,\mathrm{d}\boldsymbol{r}, \tag{3.37}$$

$$f(\boldsymbol{r}) = \mathcal{F}^{-1}[f(\boldsymbol{k})]_{\boldsymbol{r}} = \frac{1}{(2\pi)^3}\iiint F(\boldsymbol{k}) \exp(i\boldsymbol{k}\cdot\boldsymbol{r})\,\mathrm{d}\boldsymbol{k}. \tag{3.38}$$

This approach can be generalised to spherical coordinates by expressing the exponential in spherical harmonics,

$$\exp(i\boldsymbol{k}\cdot\boldsymbol{r}) = 4\pi\sum_{l,m} i^l j_l(kr) Y_l^m(\hat{\boldsymbol{r}}) Y_l^m(\hat{\boldsymbol{k}})^{*}, \tag{3.39}$$

where $j_l(kr)$ are spherical Bessel functions of the first kind. We can use Parseval's theorem to evaluate the overlap integral of the two densities,

$$\begin{aligned}\Omega^{pq}(\alpha,\beta,\gamma) &= \iiint_{\infty}^{-\infty} \rho_p(\boldsymbol{r})\rho_q(\boldsymbol{mr})^* \mathrm{d}\boldsymbol{r} \\ &= \frac{1}{(2\pi)^3} \iiint_{\infty}^{-\infty} \mathcal{F}\left[\rho_p(\boldsymbol{r})\right]_{\boldsymbol{k}} \mathcal{F}\left[\rho_q(\boldsymbol{mr})\right]^*_{\boldsymbol{k}} \mathrm{d}\boldsymbol{k}. \end{aligned} \tag{3.40}$$

Using the convolution theorem we can calculate the Fourier transforms,

$$\begin{aligned} C^p(\boldsymbol{k}) &= \mathcal{F}\left[\rho_p(\boldsymbol{r})\right]_{\boldsymbol{k}} \\ &= (2\pi\sigma_G^2)^{3/2} \exp\left(-\frac{k^2\sigma_G^2}{2}\right) \sum_j \exp(-i\boldsymbol{k}\cdot\boldsymbol{r}_j^p) \\ &= (2\pi\sigma_G^2)^{3/2} \exp\left(-\frac{k^2\sigma_G^2}{2}\right) 4\pi \sum_j \sum_{l,m} (-i)^l j_l(\kappa r_j^p) Y_l^m(\hat{\boldsymbol{r}}_j^p)^* Y_l^m(\hat{\boldsymbol{k}}), \end{aligned} \tag{3.41}$$

$$\begin{aligned} C^q(\boldsymbol{k}) &= \mathcal{F}\left[\rho_q(\boldsymbol{mr})\right]_{\boldsymbol{k}} \\ &= (2\pi\sigma_G^2)^{3/2} \exp\left(-\frac{k^2\sigma_G^2}{2}\right) 4\pi \sum_j \sum_{l,m} \sum_{m'} (-i)^l j_l(kr_j^q) D^l_{m,m'} Y_l^{m'}(\hat{\boldsymbol{r}}_j^q)^* Y_l^m(\hat{\boldsymbol{k}}). \end{aligned} \tag{3.42}$$

Instead of evaluating eq. (3.40) as an integral we can evaluate it as a sum by considering the discrete spherical Fourier transform, where we truncate the integral up to a cut-off radius, r_{cut}, and use the orthogonality relation

$$\int_0^{r_{\text{cut}}} j_l\left(\frac{\kappa_{l,n}}{r_{\text{cut}}} r\right) j_l\left(\frac{\kappa_{l,n'}}{r_{\text{cut}}} r\right) r^2 \,\mathrm{d}r = \frac{\pi\, r_{\text{cut}}^3}{4\kappa_{l,n}} j_{l+1}(\kappa_{l,n})^2 \delta_{n'n}, \tag{3.43}$$

where $\kappa_{l,n}$ is the nth root of j_l, so $j_l(\kappa_{l,n}) = 0$, to transform eq. (3.38) into a sum (the spherical analogue of a discrete Hankel transform [112]),

$$f(\boldsymbol{r}) = \sum_{l,m} \sum_{n=1}^{\infty} F_{l,n}^{m} \sqrt{\frac{2\kappa_{l,n}}{r_{\text{cut}}^3 j_{l+1}(\kappa_{l,n})^2}}\, 4\pi i^l j_l\left(\frac{\kappa_{l,n}}{r_{\text{cut}}} r\right) Y_l^m(\hat{\boldsymbol{r}}), \tag{3.44}$$

$$F_{l,n}^{m} = \frac{1}{(2\pi)^3} \sqrt{\frac{2\kappa_{l,n}}{r_{\text{cut}}^3 j_{l+1}(\kappa_{l,n})^2}} \iiint_{|\boldsymbol{r}|<r_{\text{cut}}} f(\boldsymbol{r}) 4\pi(-i)^l j_l\left(\frac{\kappa_{l,n}}{r_{\text{cut}}} r\right) Y_l^m(\hat{\boldsymbol{r}})^* \,\mathrm{d}\boldsymbol{r}. \tag{3.45}$$

If we assume that $\rho_p(\boldsymbol{r}) = \rho_q(\boldsymbol{r}) = 0$ when $|\boldsymbol{r}| \geq r_{\text{cut}}$, then by inspection of eqs. (3.41) and (3.42) we can express the density functions as,

$$\rho_p(\boldsymbol{r}) = \sum_{l,m} \sum_{n=1}^{\infty} C_{l,n}^{p,m} \sqrt{\frac{2\kappa_{l,n}}{r_{\text{cut}}^3 j_{l+1}(\kappa_{l,n})^2}}\, 4\pi i^l j_l\left(\frac{\kappa_{l,n}}{r_{\text{cut}}} r\right) Y_l^m(\hat{\boldsymbol{r}}), \tag{3.46}$$

$$C_{l,n}^{p,m} = (2\sigma_G^2)^{3/2} \sum_{j=1}^{N} \sqrt{\frac{2\kappa_{l,n}}{r_{\text{cut}}^3 j_{l+1}(\kappa_{l,n})^2}} \exp\left(-\frac{\kappa_{l,n}^2 \sigma_G^2}{2r_{\text{cut}}^2}\right) j_l\left(\frac{\kappa_{l,n}}{r_{\text{cut}}} r_j^p\right) Y_l^m(\hat{\boldsymbol{r}}_j^p)^*, \tag{3.47}$$

$$\rho_q(\boldsymbol{mr}) = \sum_{l,m} \sum_{m'} \sum_{n=1}^{\infty} D_{m,m'}^{l} C_{l,n}^{p,m} \sqrt{\frac{2\kappa_{l,n}}{r_{\text{cut}}^3 j_{l+1}(\kappa_{l,n})^2}}\, 4\pi i^l j_l\left(\frac{\kappa_{l,n}}{r_{\text{cut}}} r\right) Y_l^{m'}(\hat{\boldsymbol{r}}), \tag{3.48}$$

$$C_{l,n}^{p,m} = (2\sigma_G^2)^{3/2} \sum_{j=1}^{N} \sqrt{\frac{2\kappa_{l,n}}{r_{\text{cut}}^3 j_{l+1}(\kappa_{l,n})^2}} \exp\left(-\frac{\kappa_{l,n}^2 \sigma_G^2}{2r_{\text{cut}}^2}\right) j_l\left(\frac{\kappa_{l,n}}{r_{\text{cut}}} r_j^q\right) Y_l^m(\hat{\boldsymbol{r}}_j^q)^*. \tag{3.49}$$

We can obtain an expression for the overlap integral as,

$$\Omega^{pq}(\alpha, \beta, \gamma) = \sum_{l,m,m'} D_{m,m'}^{l} \frac{1}{(2\pi)^3} \sum_{n}^{\infty} C_{l,n}^{p,m*} C_{l,n}^{q,m'}. \tag{3.50}$$

To evaluate the overlap integral we need to specify a cut-off order, n_{cut}^l, such that

$$\kappa_{l,n_{\text{cut}}^l} \gg \frac{r_{\text{cut}}}{\sigma_G}, \tag{3.51}$$

as the zeros of the spherical Bessel function are approximately uniformly distributed, $\kappa_{l,n_{\text{cut}}^l} \propto N^{1/3}$. The SO(3) Fourier coefficients can be calculated,

$$I_{l,m,m'} = \frac{1}{(2\pi)^3} \sum_{n}^{n_{\text{cut}}^l} C_{l,n}^{p,m*} C_{l,n}^{q,m'}, \tag{3.52}$$

which will require $O(l_{\text{max}}^3 N^{1/3})$ operations, while calculating the spherical Fourier coefficients will require $O(l_{\text{max}}^2 N^{4/3})$ operations.

3.4 Including multiple species

The methods as described above for both periodic and isolated systems assume that there is only one type of atomic species present; it is straightforward to extend these methods to apply to systems with multiple different types of atomic species. In the multiple species case, the Fourier or Harmonic coefficients for each species can be calculated independently. The Fourier coefficients for the species overlap integral can be calculated separately in turn. The coefficients for the total overlap integral are equal to the sum of the species overlap Fourier coefficients, so the Fourier transform only needs to be applied once for any given alignment involving multiple species. In the case of the method described in section 3.3 the scheme is modified by only summing over the indices of identical atomic species in eq. (3.26).

Note that unlike for a calculation of a SOAP-like coefficient cross-terms between different species do not need to be considered. The SOAP coefficient is calculated by averaging the overlap integral over all rotations, so if cross-terms are not included then the SOAP coefficient would not capture any relative rotation between different species. Whereas the above calculation calculates the overlap integral for a specific rotation which will capture relative rotations between different sets of species.

4 Branch and Bound Alignment

Here we describe the branch and bound algorithm, named Go-PERMDIST, developed to deterministically calculate the minimal RMSD between two structures. Its performance is compared against other approaches in chapter 5.

This algorithm is based on the branch and bound scheme developed by Li and Hartley [52] and Yang et al. [53] to minimize the nearest neighbour distance. It works by searching for the optimal rotation/displacement, as opposed to the optimal permutation. The search space can be mapped onto a finite three-dimensional volume. The branch and bound algorithm then finds the optimal rotation/displacement by splitting the search space into a set of cubes, and then calculates a *lower bound* for the RMSD within each cube. If the lower bound of any cube is higher than the current *upper bound* of the solution then the algorithm stops searching in that cube, otherwise the algorithm splits the cube into smaller sub-cubes and repeats the algorithm for these separate *branches*. It stops once the difference between the upper bound and lowest lower bound is less than some specified tolerance. Alternatively, because the optimal solution is normally found before it is proven to be optimal, the algorithm can be run for a set number of iterations to shorten the expected run time, at the expense of the guarantee of optimality.

4.1 Deterministic calculation of RMSD

To apply a branch and bound algorithm we need to parameterise the domain over which we are searching for a solution and define bounding functions that allow us to prune the search space. For isolated clusters Go-PERMDIST follows Li and Hartley [52] and uses the angle-axis representation of rotations, where all possible rotations can be described as a point within a $\mathbb{R}^3$ sphere of radius π, and search within the minimal $[-\pi, \pi]^3$ bounding cube enclosing this sphere. Search regions that are entirely outside this sphere can be discarded to reduce the search space by

around a factor of two. For periodic systems the search space exactly corresponds to displacements that fit within a single crystal supercell.

When searching for permutation-inversion isomers a second search domain of the same size can be defined, corresponding to the set of rotations on the inverted structure. For periodic systems our search space can be defined simply within the supercell. If we wish to find the global RMSD over all the point group symmetries of the supercell, then we can add extra search domains corresponding to translations within the unit cell for each point group operation. Additionally, if we are interested in finding the closest structure from a set of configurations to a target structure, we could treat each one as a separate search domain and search over all of them.

This version of branch and bound algorithm recursively explores the solution domain by breaking each cubic search region into eight smaller cubes, and estimating a lower and upper bound for the RMSD within each cube. If a cube is found to have a lower bound higher than the best found RMSD then the algorithm stops searching in that region of the domain, progressively eliminating areas of the search space. The process terminates once it finds a region where the upper bound is within a given tolerance of the lowest lower bound. The performance depends on our ability to accurately find lower and upper bounds for the search region. The functions that are used to bound the RMSD for clusters and periodic systems are defined below.

4.1.1 Bounding RMSD for clusters

For a given rotation matrix, $\boldsymbol{m}$, with corresponding angle-axis rotation, $\boldsymbol{v}$ and search region box width θ_{B}, an upper bound of the RMSD can be found by solving the permutational assignment problem between the target structure and rotated structure. Finding the lower bound of the RMSD requires locating a lower bound of the distance between all points in the structure within the search region. We can find a lower bound using the law of cosines, where points $\boldsymbol{r}_j^p$ and $\boldsymbol{m} \cdot \boldsymbol{r}_{j'}^q$ are separated by distance $d_{jj'}$ where,

$$d_{jj'}^2 = r_j^{p\,2} + r_{j'}^{q\,2} - 2r_j^p r_{j'}^q \cos\phi_{j,j'} \tag{4.1}$$

and $\phi_{j,j'}$ is the angle between $\boldsymbol{r}_j^p$ and $\boldsymbol{m} \cdot \boldsymbol{r}_{j'}^q$. We can calculate the lower bound of the distance between the points, $\underline{d}_{jj'}$ within the search region as,

$$\underline{d}_{jj'}^2 = \min_{|\theta| \leq \theta_{\mathrm{B}'}} r_j^{p\,2} + r_{j'}^{q\,2} - 2r_j^p r_{j'}^q \cos\left(\phi_{j,j'} + \theta\right). \tag{4.2}$$

where $\theta_{B'}$ is the maximum angle by which points can be rotated relative to the rotation $\boldsymbol{m}$ in the search box (see section 4.1.1). First we calculate the upper bound for $\cos(\phi_{j,j'} + \theta)$,

$$\begin{aligned}\overline{\cos}(\phi_{j,j'}) &= \max_{|\theta| \leq \theta_{B'}} \cos(\phi_{j,j'} + \theta) \\ &= \begin{cases} 1, & \text{when } |\phi_{j,j'}| \leq \theta_{B'} \\ \cos(\phi_{j,j'}) \cos(\theta_{B'}), + |\sin(\phi_{j,j'})| \sin(\theta_{B'}), & \text{when } |\phi_{j,j'}| > \theta_{B'}. \end{cases}\end{aligned} \tag{4.3}$$

We can now calculate a lower bound for the distance between the two points,

$$\underline{d}^2_{jj'} = r_j^{p\,2} + r_{j'}^{q\,2} - 2 r_j^p r_{j'}^q \overline{\cos}(\phi_{j,j'}). \tag{4.4}$$

This pairwise lower bound between all the points in both structures can be used by an assignment problem or nearest neighbour search algorithm to produce a lower bound for the RMSD in the bounding cube.

Composing Angle-Axis Rotations

To bound the pairwise distance we need to place a bound on the maximum angle by which a point could be displaced within the search box. The approach presented here differs from that used by Li and Hartley [52], and Yang et al. [53]. They bound the maximum angle by which a point can differ after two rotations using the inequality that the angular distance between two rotations is less than or equal to the Euclidean distance between the vectors in the angle-axis representation. Here we consider how angle-axis rotations are composed to bound the maximum angle.

For an angle-axis rotation $\boldsymbol{v} + \boldsymbol{e}$ we want to find the magnitude of the rotation vector $\boldsymbol{e}'$ such that rotation by vector $\boldsymbol{v}$ then $\boldsymbol{e}'$ is equivalent to rotation by vector $\boldsymbol{v} + \boldsymbol{e}$. By considering angle-axis rotations as arcs of a great circle their composition can be viewed as equivalent to vector addition of these arcs on the surface of a unit sphere (see fig. 4.1). The angle-axis rotation vector $\boldsymbol{v}$ has equivalent great circle arc AB, vector $\boldsymbol{v} + \boldsymbol{e}$ corresponds to arc AC, vector $\boldsymbol{e}'$ is equivalent to arc BC, and the composition of angle-axis rotations $\boldsymbol{v}$ followed by $\boldsymbol{e}'$ is $\boldsymbol{v} + \boldsymbol{e}$. The starting points of the arcs are arbitrary, so we have chosen A to correspond to the starting point of the rotations $\boldsymbol{v}$ and $\boldsymbol{v} + \boldsymbol{e}$, the intersection of their great circles.

To bound the distance between two points for the set of all possible rotations in a given search box, we find a bound on the arc $BC = |\boldsymbol{e}'|$ for the same search box,

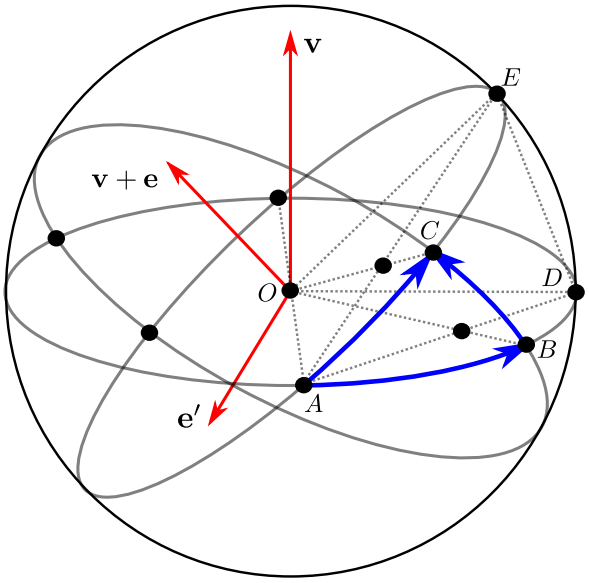


Fig. 4.1 Diagram indicating how angle-axis rotations can be composed. The rotation corresponding to the angle axis vector $\boldsymbol{v} + \boldsymbol{e}$ is equivalent to the composition of the rotation around $\boldsymbol{v}$ then $\boldsymbol{e}'$. Alternatively, considering rotations as arcs of great circles, the arc AC is equal to the arc BC added to the arc AB.

using the spherical law of cosines,

$$\cos BC = \cos AB \cos AC + \cos \angle DOE \sin AB \sin AC. \tag{4.5}$$

$\angle DOE$ is the angle between $\boldsymbol{v} + \boldsymbol{e}$ and $\boldsymbol{v}$, $AB = |\boldsymbol{v}|$ and $AC = |\boldsymbol{v} + \boldsymbol{e}|$. So we can bound these values, for a search box centred on vector $\boldsymbol{v}$ with rotation $\theta_1 = |\boldsymbol{v}|$ and box width θ_B:

$$\cos \angle DOE \geq \frac{\theta_1}{\sqrt{\theta_1^2 + 3\theta_B^2/4}} = \cos \bar{\theta}_2, \tag{4.6}$$

$$AB = \theta_1 = |\boldsymbol{v}|, \tag{4.7}$$

$$\theta_1 - \frac{\theta_\mathrm{B}}{2} \leq AC \leq \theta_1 + \frac{\theta_\mathrm{B}}{2}. \tag{4.8}$$

From this result we can find a bound for the maximum angle, $\theta_{\mathrm{B}'}$, a point can be rotated within the search box,

$$\begin{aligned}\cos\theta_{\mathrm{B}'} &= \min\left[\cos AB\right] \\ &= \min\left[\cos\frac{\theta_B}{2},\right. \\ &\qquad \left.\left(\cos(\theta_1)^2+\cos\bar{\theta}_2\sin(\theta_1)^2\right)\cos\frac{\theta_B}{2}-(1-\cos\bar{\theta}_2)\left|\sin\theta_1\cos\theta_1\sin\frac{\theta_B}{2}\right|\right].\end{aligned} \tag{4.9}$$

This result can then be used in eq. (4.3) to obtain a lower bound for the distance between two coordinates inside the search box.

4.1.2 Bounding RMSD for periodic systems

For periodic systems we can follow a similar procedure, where for a given displacement, $\boldsymbol{d}$, with bounding box width d_{B}, the upper bound of the RMSD can be found by solving the assignment problem between the target and translated structure. To find the lower bound of the RMSD we need the lower bound of the distance between all the points in the structure, so if we employ the notation in section 4.1.1, we can define the distance between points as

$$d_{jj'} = \min_{\boldsymbol{l}\in\boldsymbol{L}} |\boldsymbol{r}_j^p - \boldsymbol{r}_{j'}^q - \boldsymbol{d} + \boldsymbol{l}| = |\boldsymbol{r}_j^p - \boldsymbol{r}_{j'}^q - \boldsymbol{d} + \boldsymbol{l}_{jj'}|, \tag{4.10}$$

where $\boldsymbol{l}_{jj'}$ is the lattice vector that minimises the distance between the points. The lower bound of the distance between the points is

$$\underline{d}_{jj'} = \begin{cases} 0, & d_{jj'} \leq \sqrt{3}d_{\mathrm{B}}/2, \\ d_{jj'} - \sqrt{3}d_{\mathrm{B}}/2, & d_{jj'} > \sqrt{3}d_{\mathrm{B}}/2. \end{cases} \tag{4.11}$$

This result can be used to calculate a lower bound for the RMSD using the assignment algorithm.

We can improve this lower bound by splitting the cube into six identical square pyramids. Consider one of these pyramids (for example the top one in fig. 4.2), with triangular face normals, $\boldsymbol{v}_1$, $\boldsymbol{v}_2$, $\boldsymbol{v}_3$, and $\boldsymbol{v}_4$. The closest distance of pairs of particles for which $F_i(\boldsymbol{r}_j^p, \boldsymbol{r}_{j'}^q) = (\boldsymbol{r}_j^p - \boldsymbol{r}_{j'}^q - \boldsymbol{d} + \boldsymbol{l}) \cdot \boldsymbol{v}_i \geq 0$, for $i = 1, 2, 3, 4$, will only ever be

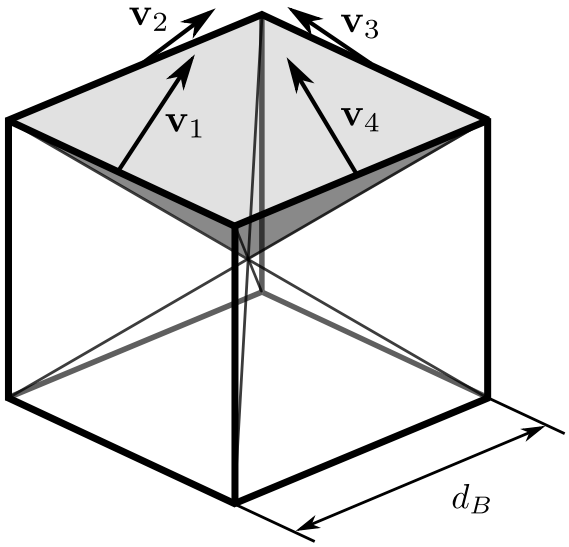


Fig. 4.2 A diagram showing how the search cube can be split into six identical pyramids. The faces of the top pyramid have been shaded.

$d_{jj'}$, so we can define a new pair wise distance lower bound for the pyramid as

$$\underline{d}_{jj'} = \begin{cases} d_{jj'}, & \text{if } F_i(\boldsymbol{r}_j^p, \boldsymbol{r}_{j'}^q) > 0 \text{ for } i = 1,2,3,4, \\ & \text{else:} \\ 0, & d_{jj'} \leq \sqrt{3} d_{\mathrm{B}}/2, \\ d_{jj'} - \sqrt{3} d_{\mathrm{B}}/2, & d_{jj'} > \sqrt{3} d_{\mathrm{B}}/2. \end{cases} . \tag{4.12}$$

The lower bound for the box then can be found by calculating the lower bound for each pyramid and taking the minimum value.

4.1.3 Approximating bounds

It requires $O(N^2)$ operations to solve the assignment problem, whereas using a k-dimensional binary search tree it is possible to find the set of nearest neighbours between two point clouds in $O(N \log N)$ operations [113]. We are only interested in the exact value of the upper bound if it is lower than any other upper bound found, so instead of solving the assignment problem for each search region we can perform an initial nearest neighbour search to give a lower bound for the upper bound search region, and then perform the same calculation as in eqs. (4.2) and (4.4) to produce a lower bound for the RMSD. If the calculation of the nearest neighbour distance

is found to give an upper bound less than the lowest found upper bound, then the more expensive calculation of the 'proper' upper bound using the assignment problem can be performed. Both the bounds calculated using the nearest neighbours approach will be at most equal to the bounds calculated using the assignment method.

Algorithm 3 Go-PERMDIST

Input: $\boldsymbol{R}^p$, $\boldsymbol{R}^q$, $\mathcal{C}_G$, (*optional* $\mathcal{C}_G^*$) ▷ Structures to align and search region(s)
Output: $\overline{E}, \boldsymbol{v}_{\text{best}}$ ▷ RMSD and transformation vector
add $\{\mathcal{C}_G^*\}$ to $\mathcal{Q}$
if Testing for Symmetries **then**
  add $\mathcal{C}_G^*$ to $\mathcal{Q}$
end if
$\overline{E} = +\infty$ ▷ Estimate of RMSD of p and q
loop
  get search cube $\mathcal{C}_t$ with lowest lower bound $\underline{E}(t)$ from $\mathcal{Q}$
  if $\overline{E} - \underline{E}(t) < \epsilon_a + \epsilon_r \underline{E}(t)$ **then**
    quit loop ▷ Stop algorithm once desired precision achieved.
  end if
  for 8 sub-cubes $\mathcal{C}_{t^*}$ of $\mathcal{C}_t$ **do**
    compute $\lfloor \overline{E} \rfloor(t)$
    if $\lfloor \overline{E} \rfloor(t) < \overline{E}$ **then** ▷ If estimate of upper bound less than current best upper bound, calculate upper bound using assignment algorithm.
      compute $\underline{E}(t)$
      if $\underline{E}(t^*) < \overline{E}$ **then**
        compute $\overline{E}(t^*)$
        if $\overline{E}(t^*) < (1 + \epsilon_r)\overline{E}$ **then**
          use PERMDIST algorithm to refine $\overline{E}(t^*)$ and $\boldsymbol{v}_{t^*}$ to $\overline{E}^*$ and $\boldsymbol{v}^*$
          $\overline{E}, \boldsymbol{v}_{\text{best}} = \overline{E}^*, \boldsymbol{v}^*$ ▷ Update best estimate of RMSD
        end if
        add $\mathcal{C}_{t^*}$ to $\mathcal{Q}$
      end if
    else
      compute $\lfloor \underline{E} \rfloor(t)$
      if $\lfloor \underline{E} \rfloor(t) < \overline{E}$ **then**
        add $\mathcal{C}_{t^*}$ to $\mathcal{Q}$
      end if
    end if
  end for
end loop

4.2 Branch and Bound algorithm

The branch and bound algorithm to find the global minimum RMSD, within a certain absolute tolerance, ϵ_a and relative tolerance, ϵ_r, uses a *best-first-search*, where the regions in search space with the smallest lower bounds are explored first. To describe the algorithm we first define some terms. We seek the globally optimum transformation vector $\boldsymbol{v}_G$ contained within a cubic search domain $\mathcal{C}_G$ of width w_G. For a periodic system $\boldsymbol{v}_G$ corresponds to the displacement vector and $w_G = L$, while for a cluster $\boldsymbol{v}_G$ corresponds to the rotation vector, $w_G = 2\pi$, and $\{\mathcal{C}_G^*\}$ are the set of search regions corresponding to point group symmetries of the system (e.g. inversion).

For a search domain, $\mathcal{C}_t$, centred at $\boldsymbol{v}_t$ and width w_t, we define the upper and lower bound of the RMSD to be $\overline{E}(t)$ and $\underline{E}(t)$. We define the upper and lower bound calculated by the nearest-neighbours approach as $\lfloor\overline{E}\rfloor(t)$ and $\lfloor\underline{E}\rfloor(t)$. We store the set of search regions in a priority queue, $\mathcal{Q}$, where the search region with the lowest lower bound, $\underline{E}(t_{\text{low}})$, is returned first. When an alignment is found with a lower bound within a certain tolerance of the current best found solution, then this alignment is refined by applying a single run of the iterative PERMDIST algorithm as described in section 2.3.2. A detailed description is given in algorithm 3.

4.2.1 Asymptotic behaviour

As the size of the search regions decreases the difference between the upper and lower bounds also decreases. For clusters we can see that with regions of angular size, θ_B, where $\theta_B \ll 1$,

$$d_{j,j'}^2 - \overline{d}_{j,j'}^2 \propto \theta_B r_j^p r_{j'}^q. \tag{4.13}$$

So the difference between the lower and upper bound will be proportional to θ_B. For periodic systems with regions of size, d_B, the difference between the lower and upper bound will be proportional d_B when $d_B \ll L/N^{1/3}$. The width of the search region is therefore proportional to the uncertainty in the lower bound. This result holds both when calculating the bounds using the assignment problem or the nearest-neighbours algorithm, so as the width of the search region decreases the difference between the bounds will decrease uniformly. This decrease guarantees that the global RMSD is found because the lowest upper bound calculated will always correspond to a possible RMSD alignment between the structures.

5 Comparison of Alignment Methods

The performance of various alignment algorithms was assessed by comparing the lowest RMSD values and associated computational cost for a test set of structures. Timing benchmarks correspond to a single CPU core on a workstation with an Intel 3.3 GHz i7 Haswell processor. We primarily benchmarked the new algorithms against the methods in the Cambridge Energy Landscapes software package, as they perform significantly better than alternative algorithms, as discussed in in section 5.3.

The FASTOVERLAP algorithm requires us to choose the width of the Gaussian kernels, σ_G. Our investigations have shown that setting σ_G equal to $1/3$ of the interatomic separation generally gives good performance. For cluster alignment the angular momentum cutoff, l_{max}, needs to be set as well, which defines the angular resolution, $\Delta_\theta = \pi/l_{\text{max}}$ of the SOFT to find the global maximum kernel correlation. For most purposes setting $l_{\text{max}} = 15$ worked well, though for large systems, the algorithm may display improved performance if the angular momentum cutoff is higher.

5.1 Periodic systems

Three different algorithms for aligning periodic systems in OPTIM [63], corresponding to keywords PERMDIST, ATOMMATCHFULL and ATOMMATCHDIST, were tested against the FASTOVERLAP kernel correlation/Gaussian overlap schemes. The algorithms were tested on amorphous local minima for a binary Lennard-Jones liquid containing 204 atoms of type A and 52 atoms of type B, with a density of $1.2\,\sigma_{AA}^{-3}$. The energies were calculated using the usual Lennard-Jones pair potential with the Stoddard–Ford quadratic cutoff [114]. The interaction parameters used were

$\epsilon_{\mathrm{AA}} = 1.0$, $\epsilon_{\mathrm{AB}} = 1.5$, $\epsilon_{\mathrm{BB}} = 0.5$, $\sigma_{\mathrm{AA}} = 1.0$, $\sigma_{\mathrm{AB}} = 0.8$ and $\sigma_{\mathrm{BB}} = 0.88$, corresponding to a popular model glass former [115].

For the FASTOVERLAP algorithm we set the kernel width to $1/3$ of the average interatomic spacing, $\sigma_G = \sigma_{\mathrm{AA}}/(3\sqrt[3]{1.2}) = 0.314\,\sigma_{\mathrm{AA}}$, and the cutoff wavevector order to 6.

5.1.1 Data generation

Two data sets of 100 distinct minima were generated using the Python Energy Landscape Explorer (PELE) [116] and used to compare the algorithms. The first set was created by a basin-hopping [12–14] global optimisation run from a random starting point. The second data set was generated by taking random steps away from one particular minimum, using a local geometry optimisation after each step to locate new minima. The steps were performed by assigning every atom a uniform random displacement along each axis of up to $0.3\,\sigma_{\mathrm{AA}}$, no permutations of atomic identity were made. The minimum RMSD for every pair of minima in each data set was calculated using several different alignment algorithms to compare them. Because these schemes are not necessarily symmetric in their arguments, all of the 10,000 possible pairs of minima were used.

The above procedure for generating the minima tends to produce pairs of structures that are already reasonably well aligned. A naive calculation of the RMSD without any form of alignment often produces an RMSD very close to the optimal value. Hence each minimum was also scrambled by applying a random global translation and permutation.

5.1.2 Performance on scrambled data

A graphical comparison of the performance for the scrambled data sets is shown in fig. 5.1. The lower the RMSD, the better the alignment. The fourth column shows the best RMSD located, so if FASTOVERLAP always calculated the lowest RMSD it would give a straight line for that column. The percentage of RMSDs found by each method within a certain tolerance of the best RMSD is shown in fig. 5.2.

For RMSD $< 0.6\,\sigma_{AA}$ the FASTOVERLAP method always found an alignment quite close to the best RMSD. However, for a small number ($\sim 1\%$) of more distant pairs of minima it fails. For these structures the RMSD is dominated by atoms that are separated by $> 1\,\sigma_{AA}$, so the approximations made in the derivation are expected to fail and optimising the overlap no longer corresponds to optimising the RMSD.

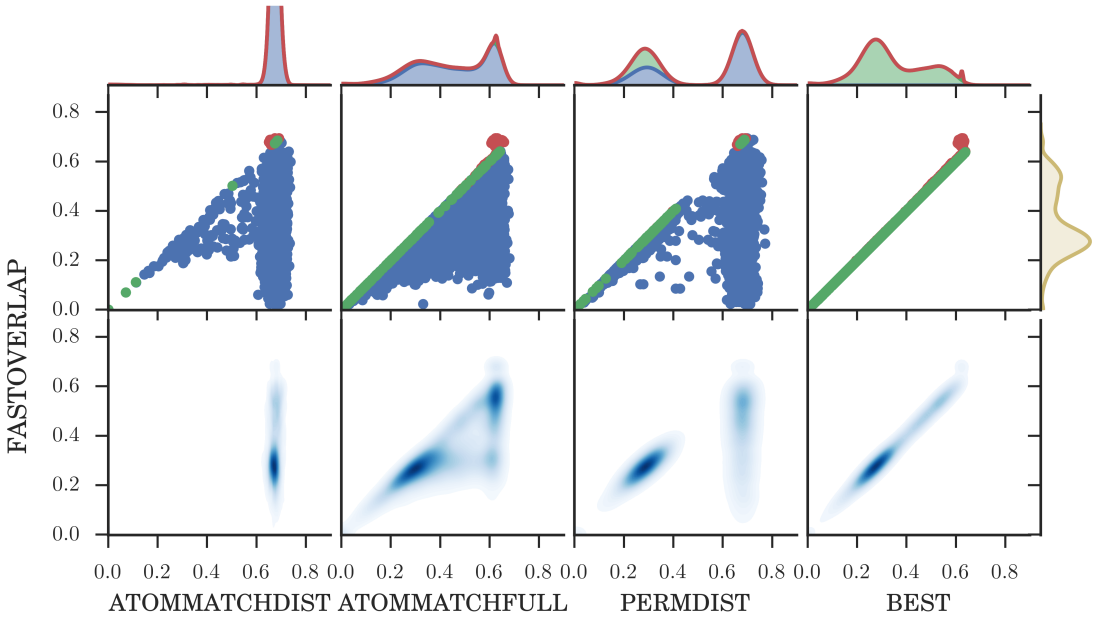


Fig. 5.1 Comparison of the $\mathrm{RMSD}/\sigma_{AA}$ calculated by FASTOVERLAP against the RMSD found by various different alignment algorithms for scrambled amorphous binary Lennard-Jones structures. BEST gives the lowest RMSD found by any means. The top row shows a scatter plot of the RMSD found by FASTOVERLAP against the RMSD found by the methods listed on the bottom; red, green and blue points indicate whether the FASTOVERLAP method found a higher, equal or lower RMSD. The bottom row shows the density distribution of the scatter plots. Above the scatter plots the marginal distribution of the RMSD found by the methods listed below are shown. On the right next to the scatter graphs the marginal distribution of the RMSD found by FASTOVERLAP is shown. All the marginal distributions are on the same scale.

These results show that when aligning reasonably close minima, the FASTOVERLAP method is very reliable and is significantly better than the other methods.

All the other methods show significantly worse performance than FASTOVERLAP, except for the more distant pairs of minima, often failing to identify relatively close minima reliably. ATOMMATCHDIST does not identify the lowest RMSD for the vast majority of minima. PERMDIST is slightly better at identifying relatively close minima than ATOMMATCHFULL, but failed to find the global minimum for nearly all the pairs separated by intermediate distances, where ATOMMATCHFULL performs slightly better.

The bimodal distribution of RMSD found is due to the two different methods for generating minima. The dataset that produced minima by stepping from the

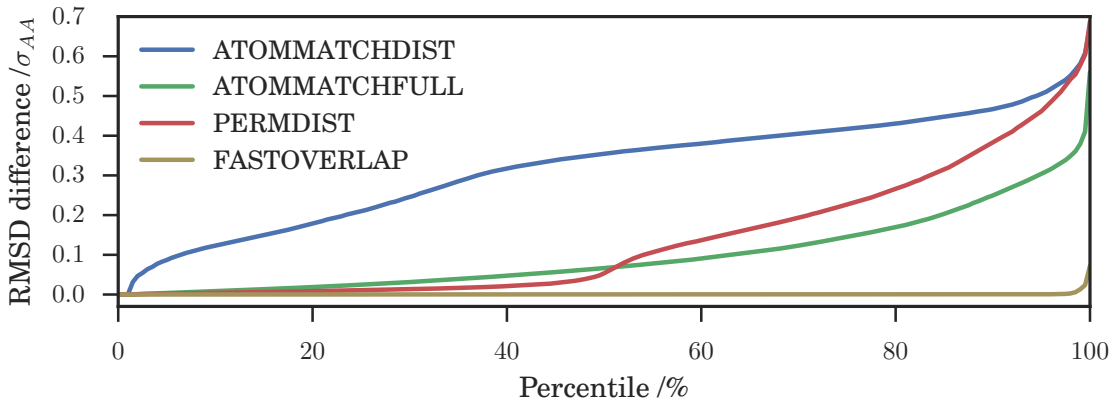


Fig. 5.2 Graph comparing the accuracy of the different methods for aligning the scrambled binary Lennard-Jones structures, plotting the difference between the calculated RMSD and the optimal value against the percentage of alignments with a smaller difference.

same minimum repeatedly tended to generate very similar structures, while the basin-hopping run tended to produce more diverse structures.

We also note that for this system the RMSD is peaked around $0.7\,\sigma_{\mathrm{AA}}$, as this is around the maximum the RMSD of the system can be after solving the assignment problem. For a bad alignment the atomic separations will be approximately evenly distributed between 0 and 1, resulting in an RMSD of around $0.7\,\sigma_{\mathrm{AA}}$. When one of the alignment algorithms fails to find the correct translational alignment then it will return an RMSD of around $0.7\,\sigma_{\mathrm{AA}}$, so worse methods will have larger peaks at $0.7\,\sigma_{\mathrm{AA}}$ due to having more failed alignments.

5.1.3 Computational complexity

To measure the computational complexity the time to perform the alignment was calculated for supercells of increasing size. The potential used in this case corresponds to a single atomic species with pairwise Lennard-Jones interactions and fixed number density $1.05\,\sigma_{\mathrm{AA}}$. System sizes ranging from 128 to 16384 atoms were tested. The average times taken for the different sized systems are shown in fig. 5.3; we observe an asymptotic approach to $O(N^2)$ scaling, as suggested by the analysis in section 3.2.2. Calculating all 10,000 alignments with the FASTOVERLAP algorithm required 90 s.

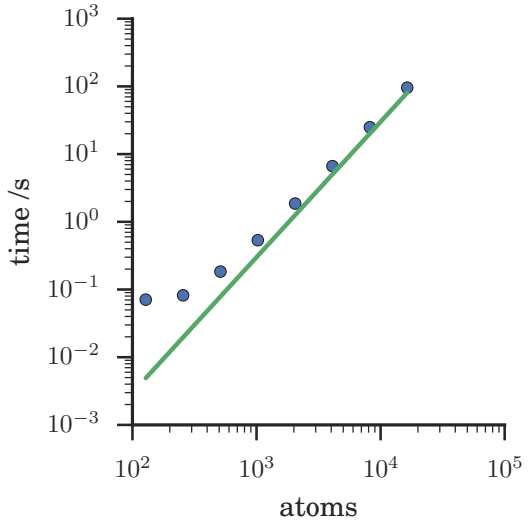


Fig. 5.3 Average time required to calculate the RMSD using the FASTOVERLAP algorithm for periodic binary Lennard-Jones minima at a number density of 1.05. The green line shows an N^2 relationship.

5.1.4 Go-PERMDIST

We found that Go-PERMDIST requires a significantly larger runtime than FASTOVERLAP to achieve comparable performance. The run time for FASTOVERLAP was approximately equivalent to around five steps of the Go-PERMDIST algorithm, which normally needed 100–1000 steps to find the optimal alignment.

5.2 Clusters

5.2.1 FASTOVERLAP

For aligning clusters the algorithm corresponding to the keyword PERMINVOPT in GMIN [62] was compared to the FASTOVERLAP algorithm for clusters. PERMINVOPT is the same algorithm as PERMDIST, but it also tests alignment for inverted structures. The maximum number of iterations in the PERMDIST algorithm was varied from 300 to 3000 to evaluate the effect of this parameter on the alignment. The algorithms were compared for Lennard-Jones (LJ) clusters of 38 atoms, LJ_{38}, using a database of 1000 distinct minima generated in a discrete path sampling study [15, 16]. Minimum RMSD values were calculated for all pairs.

For the FASTOVERLAP algorithm the kernel width was set to approximately $1/3$ of the average interatomic spacing, $\sigma_G = 0.3\,\sigma_{\mathrm{LJ}}$. The cutoff angular momentum degree was set to 15.

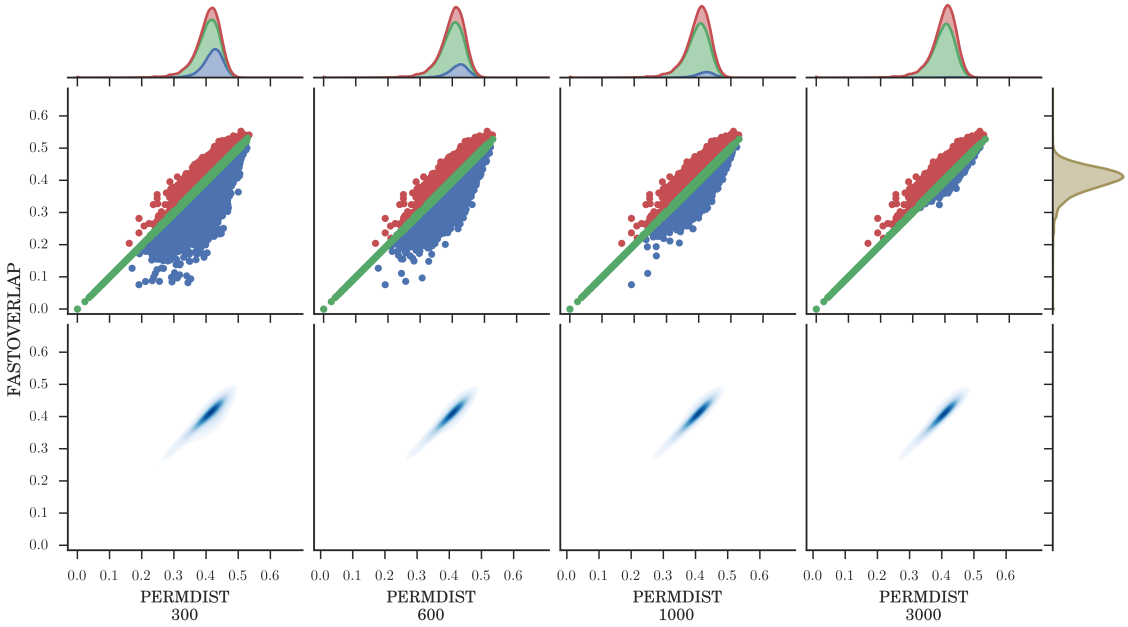


Fig. 5.4 Comparison of the RMSD calculated by FASTOVERLAP against the RMSD found by PERMDIST for clusters of 38 Lennard-Jones atoms as a function of the number of PERMDIST iterations. The top row shows a scatter plot of the RMSD found by FASTOVERLAP against the RMSD found by PERMDIST on the bottom; red, green and blue points indicate whether the FASTOVERLAP method found a higher, equal or lower RMSD. The bottom row shows the density distribution of the scatter plots. Above the scatter plots the marginal distribution of the RMSD is illustrated. On the right, next to the scatter graphs, the marginal distribution of the RMSD found by FASTOVERLAP is shown. All the marginal distributions are on the same scale.

Performance

A comparison of the performance of PERMDIST and FASTOVERLAP is shown in fig. 5.4. The percentage of RMSDs found by each method within a certain tolerance of the best RMSD is shown in fig. 5.5.

FASTOVERLAP finds the optimal RMSD for about 71% of the pairs of minima tested, and always finds the optimal RMSD for pairs separated by less than $0.15\,\sigma$. After 600 iterations PERMDIST has nearly identical performance to FASTOVERLAP, with FASTOVERLAP performing slightly better for closer pairs of structures. After 1000 iterations PERMDIST is better, the same or worse than FASTOVERLAP for 26%, 67% or 7% of the pairs of minima, and after 3000 iterations these figures change to 28%, 71% or 0.5% of the pairs.

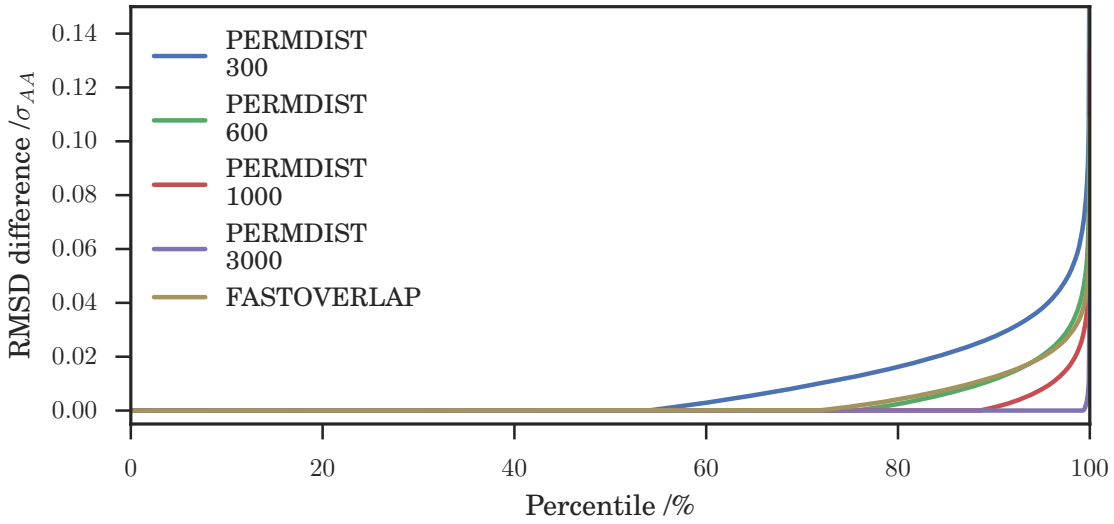
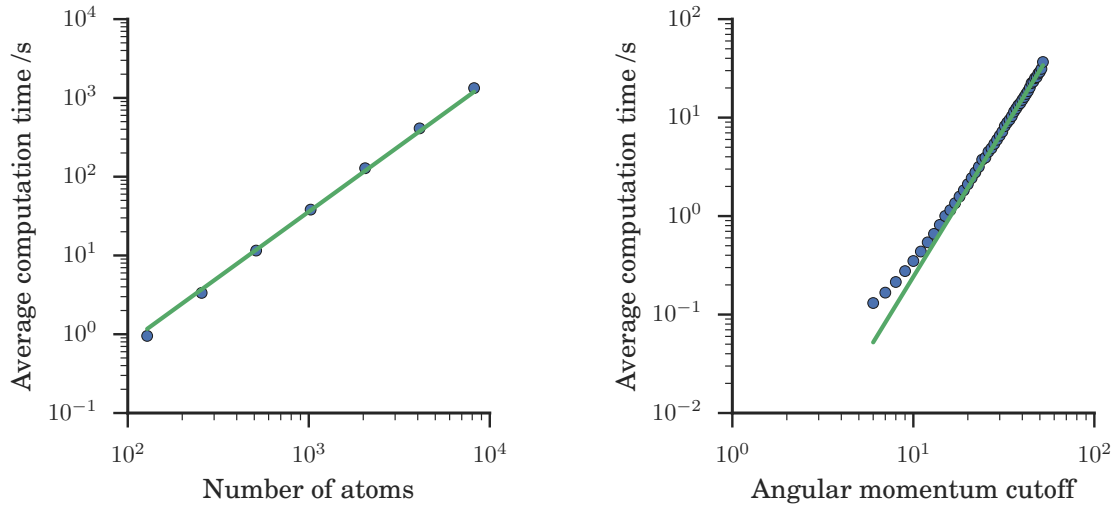


Fig. 5.5 Graph comparing accuracy of the different methods for aligning LJ_{38} clusters, showing the percentage of alignments that achieved within a certain RMSD of the best found RMSD for each respective algorithm.

The FASTOVERLAP algorithm failed to find the optimal RMSD for a few moderately close pairs of minima with 'non-cooperative' alignments [117], where the difference in structure is dominated by a small number of atoms moving a relatively long distance. For these alignments choosing a larger kernel width generally resulted in finding the optimal alignment. For more distant minima FASTOVERLAP tended to fail because numerous atoms needed to be displaced a long way in the optimal alignment, so the assumptions made in the derivation do not hold (see section 3.3).

Computational complexity

A test set of random LJ minima was used to analyse the computational scaling of the algorithm with system size ranging from 128 to 8192 atoms. The results are for a kernel width of $\sigma_G = 0.3\sigma$ and angular momentum cutoff $l_{\text{max}} = 15$. The timings of the calculations shown in fig. 5.6b confirm the expected $O(N^{5/3})$ scaling for fixed angular momentum cutoff, deduced in section 3.3. The scaling with respect to the angular momentum cutoff is shown in fig. 5.6a; the $O(l_{\text{max}}^3)$ behaviour suggests that the computational complexity is dominated by the $O(N^{5/3}l_{\text{max}}^3)$ calculation of the SO(3) Fourier coefficients, rather than the $O(l_{\text{max}}^4)$ cost of performing the inverse SO(3) Fourier transform (see section 3.3).



(a) Average time required to align different sized structures using the FASTOVERLAP algorithm, with a fixed angular momentum cutoff of $l_{\max} = 15$, for random Lennard-Jones clusters. The green line shows an $N^{5/3}$ relationship.

(b) Average time required to align finite structures corresponding to random minima for Lennard-Jones clusters of 128 atoms using the FASTOVERLAP algorithm, for a range of angular momentum cutoffs. The green line shows an $l_{\max}^3$ relationship.

Fig. 5.6 Computational complexity of FASTOVERLAP for finite clusters

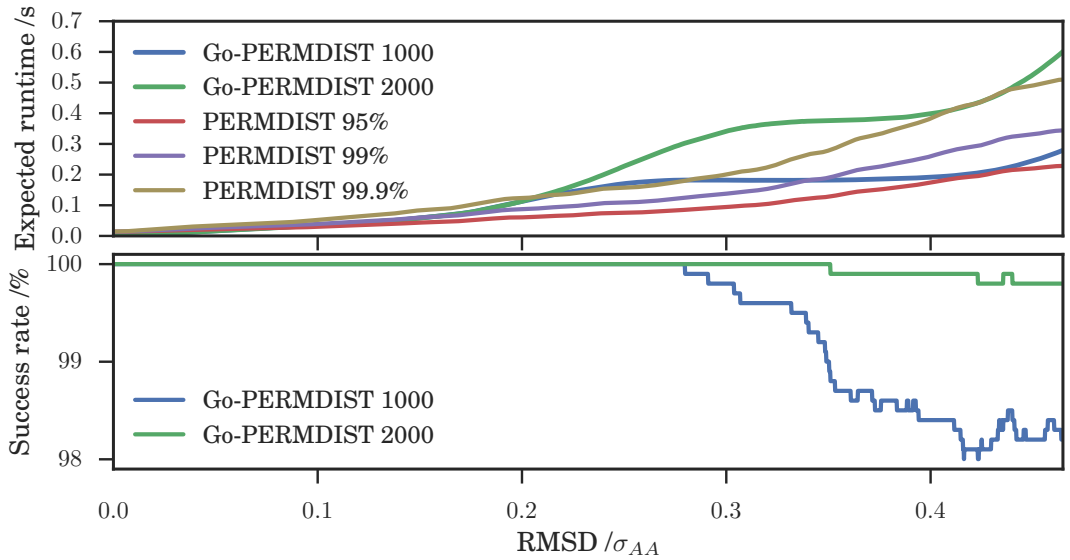


Fig. 5.7 A comparison of the performance for Go-PERMDIST and PERMDIST over a range of alignments of LJ_{38} clusters. The top graph shows the expected runtime of Go-PERMDIST limited to 1000 or 2000 iterations, and the expected runtime of the PERMDIST algorithm to achieve a given success rate to find the global minimum RMSD. The bottom graph shows the rolling-average success rate of the limited iteration Go-PERMDIST algorithm.

5.2.2 Go-PERMDIST

Although the Go-PERMDIST algorithm can calculate the minimal RMSD deterministically, for structures that are relatively distant this calculation can take an extremely long time. However if speed is critical, the Go-PERMDIST algorithm can be run with a maximum number of iterations to ensure that it terminates faster. This behaviour is offset by a slight loss in reliability.

For a pair of structures, if the random starting orientations are selected uniformly (which is not an immediately straightforward task [118]), there will be a fixed probability, p_{find}, that PERMDIST will find the global minimum RMSD for a random starting orientation, which will be proportional to the size of the basin of attraction. The probability that the correct global minimum RMSD has been found after n random orientations will be

$$p_{\text{found}}(n) = 1 - (1 - p_{\text{find}})^n \tag{5.1}$$

and the number of random starting orientations to achieve a certain success rate, f_{success} will be

$$n_{\text{rate}}(f_{\text{success}}) = \frac{\ln(1 - f_{\text{success}})}{\ln(1 - p_{\text{find}})}. \tag{5.2}$$

The maximum likelihood estimator of p_{find} is

$$\hat{p}_{\text{find}} = \frac{N_{\text{trials}}}{\sum_{j=1}^{N_{\text{trials}}} n_j} \tag{5.3}$$

for N_{trials} independent alignments, where PERMDIST finds the global minimum after n_j random starting orientations for alignment number j. For any given system size there is generally a fixed computation time for each iteration so we can use eqs. (5.2) and (5.3) to estimate the time that PERMDIST would need to run for to achieve a given level of accuracy. It was found that p_{find} was strongly correlated with the distance between the structures.

Data Generation

Three test sets were generated to compare the performance of PERMDIST and Go-PERMDIST.

$\mathbf{LJ_{38}}$ 6000 pairs of structures from the dataset used in section 5.2.1, with RMSD distributed from 0 to 0.6 σ.

Au_{55} 1000 pairs of structures from a dataset of the 100 lowest lying minima of 55 gold atoms, Au_{55}, modelled by the Gupta potential [119] and found by basin-hopping using GMIN [62], with RMSD distributed from 0 to 1.3 Å.

Au_{147} 1000 pairs of structures from a dataset of the 100 lowest lying minima of 147 gold atoms, Au_{147}, modelled by the Gupta potential [119] and found by basin-hopping using GMIN [62], with RMSD distributed from 0 to 1.5 Å.

Performance

Graphs comparing the performance of Go-PERMDIST and PERMDIST are shown in figs. 5.7 to 5.9. Each figure shows a comparison for the estimated runtime between Go-PERMDIST limited to 1000 or 2000 iterations and the expected runtime of the PERMDIST algorithm to reach a certain level of accuracy for a given RMSD (estimated using eqs. (5.2) and (5.3) and the pairs with the most similar RMSD).

The two algorithms show comparable performance, and both find higher RMSD alignments more difficult. For PERMDIST, the number of random orientations that needed to be tested increased approximately as RMSD^2 for $\text{RMSD} \gg 0$. The runtime for PERMDIST to achieve the same level of accuracy as Go-PERMDIST was generally higher or similar to the expected runtime of Go-PERMDIST.

Ensuring that rotations were sampled uniformly was important for many alignments with PERMDIST, especially for pairs of structures with distinct alignments but very similar RMSDs. For these structures the PERMDIST algorithm would often be attracted to the region around the alignment with a slightly higher RMSD, and so would take a disproportionately long time to find the best alignment.

5.3 Comparison to permutation optimisation schemes

We also tested the performance of our own implementations of other algorithms against the above methods, in particular the Monte Carlo permutation optimisation algorithm developed by Sadeghi et al. [40] and the branch and bound permutation optimisation algorithm developed by Hong et al. [61].

Both algorithms were found to scale exponentially with system size, which made the calculation of the minimal RMSD for systems with more than around 15 atoms much slower than FASTOVERLAP, PERMDIST and Go-PERMDIST. This behaviour is expected, as both algorithms optimise over an exponentially large

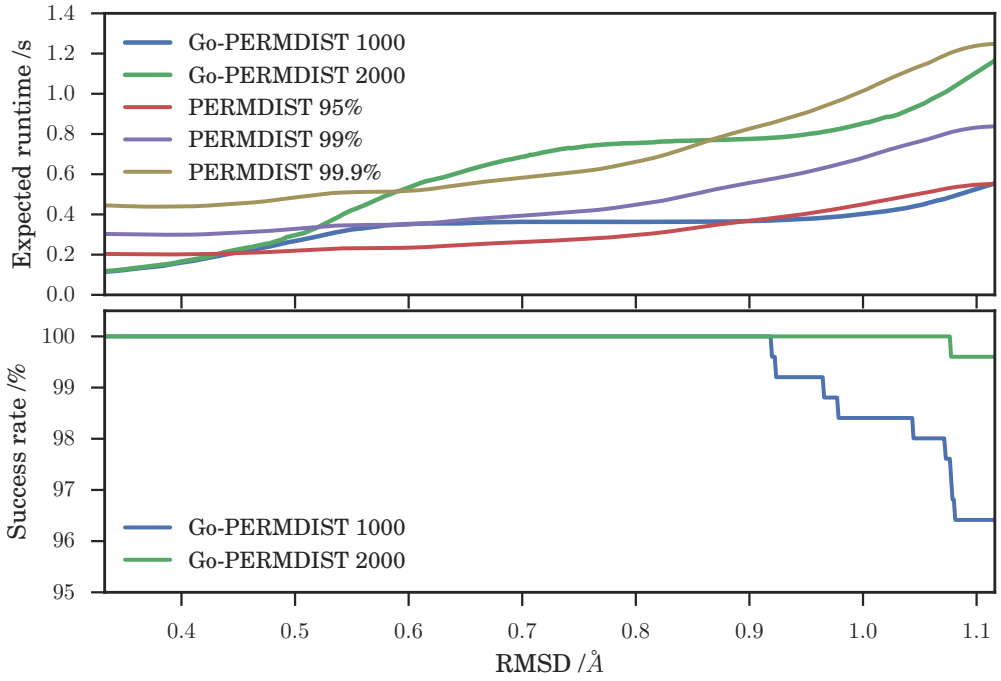


Fig. 5.8 A comparison of the performance for Go-PERMDIST and PERMDIST over a range of alignments of Au_{55} clusters. The top graph shows the expected runtime of Go-PERMDIST limited to 1000 or 2000 iterations, and the expected runtime of the PERMDIST algorithm to achieve a given success rate to find the global minimum RMSD. The bottom graph shows the rolling-average success rate of the limited iteration Go-PERMDIST algorithm.

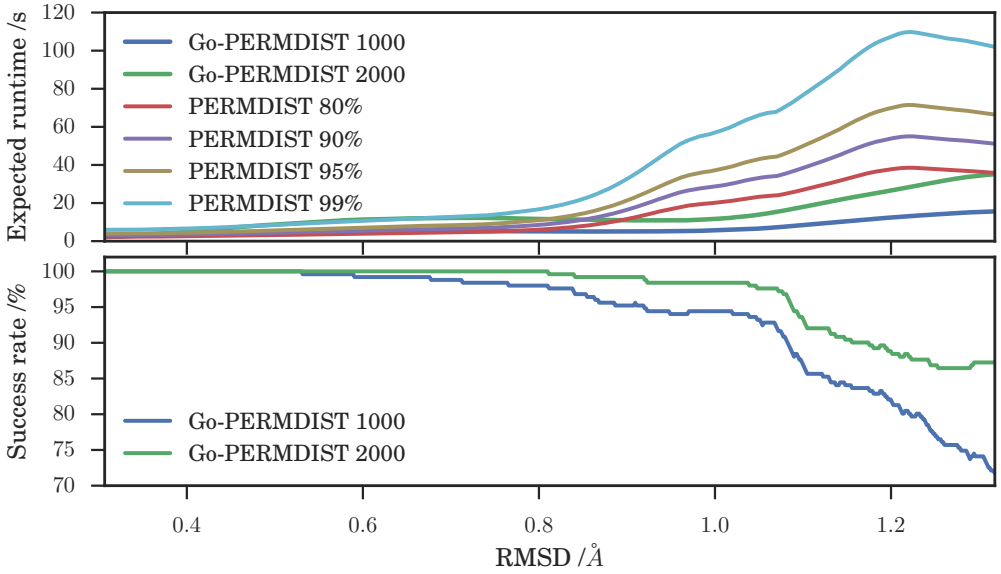


Fig. 5.9 A comparison of the performance for Go-PERMDIST and PERMDIST over a range of alignments of Au_{147} clusters. The top graph shows the expected runtime of Go-PERMDIST limited to 1000 or 2000 iterations, and the expected runtime of the PERMDIST algorithm to achieve a given success rate to find the global minimum RMSD. The bottom graph shows the rolling-average success rate of the limited iteration Go-PERMDIST algorithm.

space of permutations, whereas the FASTOVERLAP, PERMDIST and Go-PERMDIST algorithms are effectively 3D optimisation algorithms.

If the structures are initially relatively well aligned then the Monte Carlo permutation algorithm could occasionally find the optimal alignment relatively quickly. However, for systems with as few as 12 atoms it could take over 20,000 steps to find the optimal alignment, especially if the structures were not initially close. We implemented the algorithm in python using PELE [116].

The branch and bound permutation optimisation algorithm was relatively efficient when aligning permutational isomers (or close to permutational isomers) compared to the other methods tested, because it is then easier to discard branches with the wrong permutation, so only a relatively small number of permutations need to be tested. However, the number of permutations required increases exponentially with RMSD, so the algorithm showed poor performance in general. We implemented the algorithm in python, and to improve the performance we used the shortest-augmenting path algorithm to calculate the upper bounds of the branches.

Code

The python and Fortran code that was used to perform the FASTOVERLAP and Go-PERMDIST calculations can be found at:

https://github.com/matthewghgriffiths/fastoverlap

6 Nested Basin-Sampling

In this chapter we report a new method for the evaluation of integrals over an energy landscape exhibiting broken ergodicity, *nested basin-sampling* (NBS). We apply this approach to LJ_{31}. The results of this simulation are analysed and the heat capacity calculated is benchmarked against results generated by parallel tempering (PT), basin-sampling parallel tempering (BSPT), and standard nested sampling (NS) simulations.

We also introduce the No Galilean U-Turn Sampler (NoGUTS), a new sampling scheme based on the No U-Turn Sampler (NUTS) to work with the Galilean Monte Carlo scheme to aid the efficient generation of new live points and a simple stepsize adjustment scheme for nested sampling to ensure effective selection of stepsize during the NBS simulation.

6.1 Introduction

Standard nested sampling must be run with a minimum number of live points when simulating systems exhibiting broken ergodicity with nested sampling so that at there are a sufficient number of points in each basin once they become disconnected, to ensure uniform sampling across the disconnected basins. Choosing the correct number of live points to simulate poses a challenge, as it is not obvious *a priori* what will be sufficient, and simulating a large number of live points is computational expensive and tricky to parallelise.

NBS tackles this problem by performing NS simulations with a single live point, with each new live point being spawned by a random walk originating from the previous live point. These simulations will be called nested optimisations (NOpts) as each simulation is guaranteed to finish in a minimum. This approach means that a given NOpt will never jump out the basin it is currently in, so the NOpts that are in

the same basin can be combined together to provide an estimate of the configuration volume of that specific basin. We describe NOpts in more detail in section 6.2.

Instead of inferring the volumes of disconnected basins by the number of live points present in a given basin, the volume of a given basin can be estimated from the fraction of NOpts that fall into it and the statistics of the aggregated NOpts associated with the basin. So to ensure sufficient sampling across disconnected regions enough NOpts must be done to ensure sufficient statistics to estimate the probability of a NOpt landing in a specific basin and there are enough NOpts in each basin such that their aggregated statistics are sufficiently accurate. This approach has the advantage of being highly parallelisable, and does not require the number of live points to be chosen before beginning the simulation. The computational details for inferring the basin volumes and integrals over the potential energy surface (PES) are discussed in section 6.3.

In this superposition based approach basins are considered and sampled separately. In contrast SENS uses the harmonic approximation to the potential to seed low energy replicas into the simulation [94]. Similar superposition-based approaches are used by MULTINEST and POLYCHORD, though in these codes it is assumed that it is possible to cluster the configuration space into either disconnected regions or a set of hyperellipsoids, which is not always straightforward for an arbitrary PES and/or a high-dimensional system.

The behaviour of regions in configuration space becoming mutually inaccessible can be visualised using a disconnectivity graph (DG) [17] (see also fig. 1.2), shown

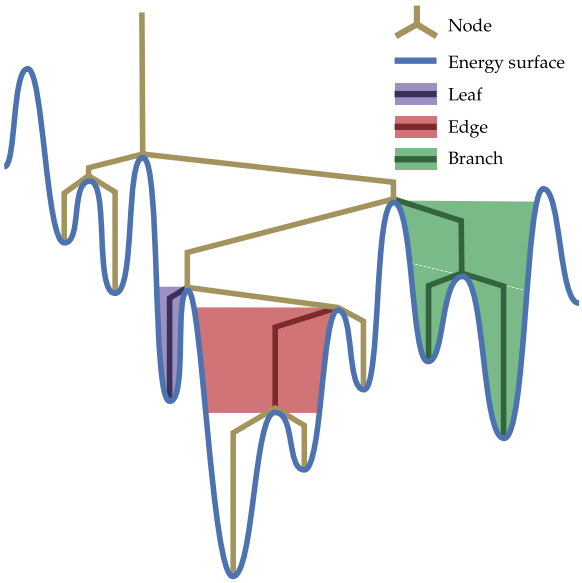


Fig. 6.1 Classification scheme for a disconnectivity graph. Two points in space are considered to be connected if it is possible to move from the highest energy point to the lowest energy point without exceeding the energy of the highest. A disconnectivity graph describes the topology of the connectivity of an energy landscape. Nodes on a disconnectivity graph indicate the energy threshold at which different *child* volumes in the energy landscape become connected. A *branch* volume of a node corresponds to a region that becomes connected to other branches above the node.

in fig. 6.1. Any random walk constrained to stay below the energy of a given *node* will only be able to visit the volume of space associated with whichever *branch* the random walk begins in. An *edge* of the disconnectivity graph represents a region in space where all pairs of points in the region are connected by a barrierless path, where the energy does not exceed the higher value of the pair. We will refer to different regions or volumes of configuration space as leaves, edges or branches. In NBS it is these regions that are considered.

The DG we employ for NBS differs from standard DGs, where nodes correspond to the minimum energy transition state that connects two branches descending from that node. In NBS the nodes correspond to the energy level at which MC walks to generate new replicas in the two regions do not cross the barrier between them, which happens once the probability of the MC walk moving between the regions becomes too small. These barriers will be termed *lazy* barriers to differentiate them from potential energy barriers, because the MC walk has not been run for 'long enough' to cross them. For brevity, we will henceforth refer to lazy basins simply as basins. An approach to self-consistently infer this DG from just the NOpt results without needing a PES specific similarity metric is described in section 6.7.2.

NBS presents some advantages over standard nested sampling:

- the NOpt simulations are embarrassingly parallel,
- it is easy to perform additional simulations to increase the accuracy of the simulation,
- by modelling the volumes of the basins independently it is possible to enhance the accuracy at low energies by using the harmonic superposition approximation to calculate the configuration volume at low energies, in a similar manner to basin-sampling.

Unfortunately, these advantages come at the cost of creating more stringent requirements for the generation of new live points. Ensuring that the new live points are sufficiently decorrelated from the previous point becomes more important as the number of live points in the simulation decreases.

To alleviate these issues we introduce NoGUTS in section 6.5 to facilitate more efficient generation of new replicas within a basin. In addition the selection of an appropriate stepsize during the course of the simulation is important to ensure that each new live point is generated efficiently, however naively changing the stepsize can induce bias [120, 121].

In section 6.6 we describe a scheme to enable selection of an appropriate stepsize during the course of a NOpt, which mostly nullifies the sampling bias that can occur when the stepsize is adjusted during a MC simulation. A logistic model relating the acceptance rate to the cut-off energy and stepsize is used to choose an efficient stepsize, with a delay to reduce history dependence.

6.2 Nested optimisation

The NBS DG can be sampled by *nested optimisation* (NOpt), defined as NS with only a single live point. Performing NS with a single live point means that new replicas are always spawned by MC walks of length $N_{\mathrm{MC}}^{\mathrm{opt}}$ starting from the location of the last live point, which means that the NS will never jump across a lazy barrier. As the nested optimisation run continues it will therefore descend the NBS DG, sampling all the edges connected from the starting edge, to the minimum that it finishes in (see fig. 6.1).

A single NOpt run will not provide good statistics about the configuration volume of the edges it samples, but as more nested optimisation runs are completed and merged (see section 2.4.4) a more accurate picture of the configuration volume of the edges can be built. Furthermore, at any given node, the relative volumes of each of the child edges of the node can be estimated by analysing the statistics of the number of NOpt runs that fall into each edge, which is discussed in detail in sections 6.3.3 and 6.3.4.

Each nested optimisation run is completely independent, so these calculations are embarrassingly parallel to perform.

6.2.1 Stopping criterion

Many nested sampling algorithms use the statistics of the live points to determine a stopping criterion for the simulation, commonly when the energy difference between the highest and lowest energy replica reduces below some energy tolerance. In NBS, this approach would not work, as there is only ever one live point. Instead the statistics of the dead points can be considered. For example, the difference in energy, $V_{\mathrm{tol}}^{\mathrm{opt}}$, between the current live point and the one sampled $N_{\mathrm{stop}}^{\mathrm{opt}}$ iterations ago, will be approximately equal to the energy difference in a $2^{N_{\mathrm{stop}}^{\mathrm{opt}}}$ live point standard NS simulation at the same energy cut-off, which makes this comparison an effective termination criterion for an NOpt.

6.3 Nested basin-sampling calculations

Supposing we have performed a set of nested optimisation runs for a PES, and we know the path that every run took during the simulation. Using these results we can proceed with a calculation similar to that described in section 2.4.4 to calculate the global properties of the PES, as in eq. (2.22).

6.3.1 Notation

An edge volume, $\Phi^{\beta_\delta}_{\beta_{\delta+1}}$, can be defined with respect to the NBS DG by its parent node, β_δ, and child node, $\beta_{\delta+1}$; where δ is the depth of the node on the NBS DG, and β indexes the siblings of the node. The *branch volume* associated with a node, β_δ, can be defined as $\Phi_{\beta_\delta} = \Phi^{\beta_{\delta-1}}_{\beta_\delta} + \sum_{\beta_{\delta+1}} (\Phi_{\beta_{\delta+1}})$, see figs. 6.1 and 6.2. For a edge connecting to node β_δ, the jth point of the aggregated runs has energy V^{β_δ} and $n^{\beta_\delta}_j$ live points present. $M^{\beta_\delta}_{\beta_{\delta+1}}$ runs fall from branch β_δ into $\beta_{\delta+1}$.

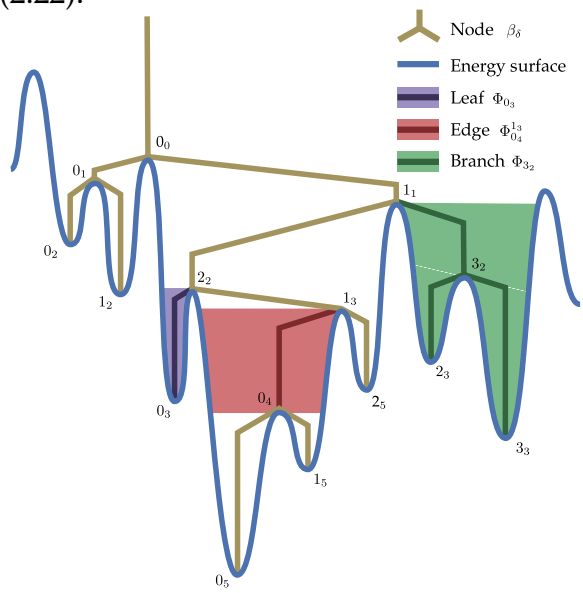


Fig. 6.2 The notation scheme for a NBS disconnectivity graph.

The integral of $f(V(\boldsymbol{x}))$ over Φ_0 will be equal to the sum of the integrals over all the edges of the disconnectivity graph,

$$\mathcal{I}_{\Phi_0}[f] = \sum_{\delta} \sum_{\beta_\delta} \sum_{\beta_{\delta+1}} \mathcal{I}_{\Phi^{\beta_\delta}_{\beta_{\delta+1}}}[f]. \tag{6.1}$$

In section 6.3.3 we show how to compute the first and second moments of $\mathcal{I}_{\Phi_0}[f]$, following a similar, if a little more involved, process to that described in section 2.4.4. In section 6.3.4 we generalise this approach to estimate the configuration volume from the volumes of child leaves, which can be calculated using the harmonic approximation, as explained in section 6.4.2. In section 6.3.5 we show how these estimate can be combined to create a more accurate overall estimate.

6.3.2 Estimating basin configuration volumes

If the NBS DG is known and there is a set of NOpt runs, the configuration volume of the edges can be estimated by two complementary methods.

1. The configuration volumes of each edge of the NBS DG can be calculated using NS, with the additional slight complication that the relative configuration volumes of edges connected to any given node is modelled by the Dirichlet distribution (see section 2.5.2). Where we assume that the process by which the NOpts fall into different edges can be modelled as a multinomial process with the multinomial probabilities probability proportional to the volume of the child edge at the energy at which the edges become disconnected during a NOpt.

2. Because the configuration volume has been separated into different regions it is also possible to estimate the configuration volume near a minimum basin (leaf) using the harmonic superposition approximation, eq. (2.19). The relative volumes of *higher* energy levels can then also be estimated from the aggregated NOpt runs, calculating the configuration volumes *bottom-up*. This calculation enhances the relative basin size estimates at low energy.

The mathematical details for computing these top-down and bottom-up estimates of the configuration volume are described in section 6.3, given a known NBS DG and aggregated NS results for each edge.

6.3.3 Top-down calculations

First we consider the edge volume, $\Phi_{\beta_\delta}^{\beta_{\delta-1}}$, which we can express in terms of the parent and child branch volumes,

$$\Phi_{\beta_{\delta+1}} = p_{\beta_{\delta+1}}^{\beta_\delta} \left(\Phi_{\beta_\delta} - \Phi_{\beta_\delta}^{\beta_{\delta-1}} \right), \tag{6.2}$$

where $p_{\beta_{\delta+1}}^{\beta_\delta}$ is the *branch probability* of a NS run going from branch Φ_{β_δ} to $\Phi_{\beta_{\delta+1}}$. The branch probability will be Dirichlet distributed over the indexes, $\beta_{\delta+1}$,

$$p_{\beta_{\delta+1}}^{\beta_\delta} \sim \mathrm{Dir}(M_{\beta_{\delta+1}}^{\beta_\delta}), \tag{6.3}$$

and we can calculate the moments of the branch probabilities,

$$\mathbb{E}_{\mathrm{B}}\left[p_{\beta_{\delta+1}}^{\beta_\delta}\right] = \frac{M_{\beta_{\delta+1}}^{\beta_\delta}}{M^{\beta_\delta}}, \tag{6.4}$$

$$\mathbb{E}_{\mathrm{D}}\left[{p_{\beta_{\delta+1}}^{\beta_\delta}}^2\right] = \frac{M_{\beta_{\delta+1}}^{\beta_\delta}(M_{\beta_{\delta+1}}^{\beta_\delta}+1)}{M^{\beta_\delta}(M^{\beta_\delta}+1)}, \tag{6.5}$$

$$\mathbb{E}_{\mathrm{D}}\left[p_{\beta_{\delta+1}}^{\beta_\delta} p_{\beta'_{\delta+1}}^{\beta_\delta}\right] = \frac{M_{\beta_{\delta+1}}^{\beta_\delta} M_{\beta'_{\delta+1}}^{\beta_\delta}}{M^{\beta_\delta}(M^{\beta_\delta}+1)}, \tag{6.6}$$

where $\mathbb{E}_{\mathrm{D}}$ indicates that this is the expectation of the top down volume. The *edge ratio* $X_{\beta_\delta}^{\beta_{\delta-1}} = \left(\Phi_{\beta_\delta} - \Phi_{\beta_\delta}^{\beta_{\delta-1}}\right)/\Phi_{\beta_\delta}$ can be calculated using NS, as

$$X_{\beta_\delta}^{\beta_{\delta-1}} = \prod_{j=1}^{N_{\mathrm{NS}}^{\beta_\delta}} t_j^{\beta_\delta}, \tag{6.7}$$

which will have moments

$$\mathbb{E}_{\mathrm{D}}\left[X_{\beta_\delta}^{\beta_{\delta-1}}\right] = \prod_{j=1}^{N_{\mathrm{NS}}^{\beta_\delta}} \frac{n_j^{\beta_\delta}}{n_j^{\beta_\delta}+1}, \tag{6.8}$$

$$\mathbb{E}_{\mathrm{D}}\left[{X_{\beta_\delta}^{\beta_{\delta-1}}}^2\right] = \prod_{j=1}^{N_{\mathrm{NS}}^{\beta_\delta}} \frac{n_j^{\beta_\delta}}{n_j^{\beta_\delta}+2}. \tag{6.9}$$

Hence we can define the edge volume in terms of the edge ratios and branch probabilities,

$$\Phi_{\beta_{\delta'+1}}^{\beta_{\delta'}} = \Phi_0\, p_{\beta_{\delta'+1}}^{\beta_{\delta'}} (1 - X_{\beta_{\delta'+1}}^{\beta'_\delta}) \prod_{\delta=0}^{\delta'-1} p_{\beta_{\delta+1}}^{\beta_\delta} X_{\beta_{\delta+1}}^{\beta_\delta}. \tag{6.10}$$

The different edge ratios will be uncorrelated, so calculating the moments of $\Phi_{\beta_{\delta'+1}}^{\beta_{\delta'}}$ can be done by substituting the appropriate moments into eq. (6.9).

As $X_{\beta_{\delta+1}}^{\beta_\delta}$ and $\mathcal{I}_{\boldsymbol{\Phi}_{\beta_{\delta+1}}^{\beta_\delta}}[f]$ both depend on $t_j^{\beta_{\delta+1}}$, $X_{\beta_{\delta+1}}^{\beta_\delta}$ will be correlated with $\mathcal{I}_{\boldsymbol{\Phi}_{\beta_{\delta+1}}^{\beta_\delta}}[f]$, so we need to calculate the moments of the product $X_{\beta_{\delta+1}}^{\beta_\delta}\mathcal{I}_{\boldsymbol{\Phi}_{\beta_{\delta+1}}^{\beta_\delta}}[f]$ in order to make

an unbiased estimate of $\bar{g}$,

$$\mathbb{E}_{\mathrm{D}}\left[X_{\beta_{\delta+1}}^{\beta_\delta}\mathcal{I}_{\Phi_{\beta_{\delta+1}}^{\beta_\delta}}[f]\right] = \mathbb{E}_{\mathrm{D}}\left[\Phi_{\beta_{\delta+1}}^{\beta_\delta}\right]\left(\prod_{l=1}^{N_{\mathrm{NS}}^{\beta_{\delta+1}}}\frac{n_l^{\beta_{\delta+1}}}{n_l^{\beta_{\delta+1}}+1}\right)\sum_{j}^{N_{\mathrm{NS}}^{\beta_{\delta+1}}}\left(\frac{g_j^{\beta_{\delta+1}}}{n_j^{\beta_{\delta+1}}}\prod_{k=1}^{j}\frac{n_k^{\beta_{\delta+1}}+1}{n_k^{\beta_{\delta+1}}+2}\right), \tag{6.11}$$

$$\mathbb{E}_{\mathrm{D}}\left[X_{\beta_{\delta+1}}^{\beta_\delta}{}^2\mathcal{I}_{\Phi_{\beta_{\delta+1}}^{\beta_\delta}}[f]\right] = \mathbb{E}_{\mathrm{D}}\left[\Phi_{\beta_{\delta+1}}^{\beta_\delta}\right]\left(\prod_{l=1}^{N_{\mathrm{NS}}^{\beta_{\delta+1}}}\frac{n_l^{\beta_{\delta+1}}}{n_l^{\beta_{\delta+1}}+2}\right)\sum_{j}^{N_{\mathrm{NS}}^{\beta_{\delta+1}}}\left(\frac{g_j^{\beta_{\delta+1}}}{n_j^{\beta_{\delta+1}}}\prod_{k=1}^{j}\frac{n_k^{\beta_{\delta+1}}+2}{n_k^{\beta_{\delta+1}}+3}\right), \tag{6.12}$$

$$\begin{aligned}\mathbb{E}_{\mathrm{D}}\left[X_{\beta_{\delta+1}}^{\beta_\delta}\mathcal{I}_{\Phi_{\beta_{\delta+1}}^{\beta_\delta}}[f^2]\right] &= \mathbb{E}_{\mathrm{D}}\left[\Phi_{\beta_{\delta+1}}^{\beta_\delta}{}^2\right]\left(\prod_{l=1}^{N_{\mathrm{NS}}^{\beta_{\delta+1}}}\frac{n_l^{\beta_{\delta+1}}}{n_l^{\beta_{\delta+1}}+1}\right)\\ &\times\sum_{l=1}^{N_{\mathrm{NS}}^{\beta_{\delta+1}}}\left[\frac{2g_l^{\beta_{\delta+1}}}{n_l^{\beta_{\delta+1}}+1}\left(\prod_{k=1}^{j}\frac{n_k^{\beta_{\delta+1}}+1}{n_k^{\beta_{\delta+1}}+2}\right)\sum_{j=1}^{l}\left(\frac{g_j^{\beta_{\delta+1}}}{n_j^{\beta_{\delta+1}}+2}\prod_{k=1}^{j}\frac{n_k^{\beta_{\delta+1}}+2}{n_k^{\beta_{\delta+1}}+3}\right)\right],\end{aligned} \tag{6.13}$$

$$\begin{aligned}\mathbb{E}_{\mathrm{D}}\left[X_{\beta_{\delta+1}}^{\beta_\delta}{}^2\mathcal{I}_{\Phi_{\beta_{\delta+1}}^{\beta_\delta}}[f^2]\right] &= \mathbb{E}_{\mathrm{D}}\left[\Phi_{\beta_{\delta+1}}^{\beta_\delta}{}^2\right]\left(\prod_{l=1}^{N_{\mathrm{NS}}^{\beta_{\delta+1}}}\frac{n_l^{\beta_{\delta+1}}}{n_l^{\beta_{\delta+1}}+2}\right)\\ &\times\sum_{l=1}^{N_{\mathrm{NS}}^{\beta_{\delta+1}}}\left[\frac{2g_l^{\beta_{\delta+1}}}{n_l^{\beta_{\delta+1}}+2}\prod_{k=1}^{j}\frac{n_k^{\beta_{\delta+1}}+2}{n_k^{\beta_{\delta+1}}+3}\sum_{j=1}^{l}\left(\frac{g_j^{\beta_{\delta+1}}}{n_j^{\beta_{\delta+1}}+3}\prod_{k=1}^{j}\frac{n_k^{\beta_{\delta+1}}+3}{n_k^{\beta_{\delta+1}}+4}\right)\right].\end{aligned} \tag{6.14}$$

To simplify this calculation, we define the *branch integral*,

$$\mathcal{I}_{\Phi_{\beta_\delta}}[f] = \sum_{\delta'=\delta}^{\arg\max_\delta \beta_\delta}\mathcal{I}_{\Phi_{\beta_\delta}^{\beta_{\delta-1}}}[f], \tag{6.15}$$

over the configuration space, so we can define the branch integral in terms of the edge integral and daughter branch integral,

$$\mathcal{I}_{\Phi_{\beta_\delta}}[f] = \mathcal{I}_{\Phi_{\beta_\delta}^{\beta_{\delta-1}}}[f] + \sum_{\beta'}\mathcal{I}_{\Phi_{\beta'_{\delta+1}}}[f]. \tag{6.16}$$

Hence the first moment can be calculated as

$$\mathbb{E}_{\mathrm{D}}\left[\mathcal{I}_{\Phi_{\beta_\delta}}[f]\right] = \mathbb{E}_{\mathrm{D}}\left[\mathcal{I}_{\Phi^{\beta_{\delta-1}}_{\beta'_\delta}}[f]\right] + \sum_{\beta'} \mathbb{E}_{\mathrm{D}}\left[\mathcal{I}_{\Phi_{\beta'_{\delta+1}}}[f]\right], \tag{6.17}$$

and the second moment will be

$$\mathbb{E}_{\mathrm{D}}\left[\mathcal{I}_{\Phi_{\beta_\delta}}[f]^2\right] = \mathbb{E}_{\mathrm{D}}\left[\mathcal{I}_{\Phi^{\beta_{\delta-1}}_{\beta_\delta}}[f]^2\right] + \sum_{\beta'} \mathbb{E}_{\mathrm{D}}\left[\mathcal{I}_{\Phi^{\beta_{\delta-1}}_{\beta'_\delta}}[f]\mathcal{I}_{\Phi_{\beta'_{\delta+1}}}[f]\right] + \sum_{\beta',\beta''} \mathbb{E}_{\mathrm{D}}\left[\mathcal{I}_{\Phi^{\beta_\delta}_{\beta'_{\delta+1}}}[f]\mathcal{I}_{\Phi^{\beta_\delta}_{\beta''_{\delta+1}}}[f]\right], \tag{6.18}$$

and the moments can be calculated as

$$\mathbb{E}_{\mathrm{D}}\left[\mathcal{I}_{\Phi^{\beta_{\delta-1}}_{\beta_\delta}}[f^2]\right] = \mathbb{E}_{\mathrm{D}}\left[p^{\beta_{\delta-1}}_{\beta_\delta}\right] \times \left(\mathbb{E}_{\mathrm{D}}\left[\mathcal{I}_{\Phi^{\beta_{\delta-1}}_{\beta_{\delta+1}}}[f^2]\right] - 2\mathbb{E}_{\mathrm{D}}\left[X^{\beta_{\delta-1}}_{\beta_\delta}\mathcal{I}_{\Phi^{\beta_{\delta-1}}_{\beta_\delta}}[f^2]\right] + \mathbb{E}_{\mathrm{D}}\left[\left(X^{\beta_{\delta-1}}_{\beta_\delta}\mathcal{I}_{\Phi^{\beta_{\delta-1}}_{\beta_\delta}}[f^2]\right)\right]\right), \tag{6.19}$$

$$\mathbb{E}_{\mathrm{D}}\left[\mathcal{I}_{\Phi^{\beta_{\delta-1}}_{\beta_\delta}}[f]\mathcal{I}_{\Phi_{\beta'_{\delta+1}}}[f]\right] = \frac{\mathbb{E}_{\mathrm{D}}\left[{\Phi_{\beta_{\delta-1}}}^2\right]}{\mathbb{E}_{\mathrm{D}}\left[\Phi_{\beta_\delta}\right]\mathbb{E}_{\mathrm{D}}\left[\Phi^{\beta_\delta-1}_{\beta_\delta}\right]} \times \left(\mathbb{E}_{\mathrm{D}}\left[X^{\beta_{\delta-1}}_{\beta_\delta}\mathcal{I}_{\Phi^{\beta_{\delta-1}}_{\beta_\delta}}[f]\right] - \mathbb{E}_{\mathrm{D}}\left[{X^{\beta_{\delta-1}}_{\beta_\delta}}^2\mathcal{I}_{\Phi^{\beta_{\delta-1}}_{\beta_\delta}}[f]\right]\right)\mathbb{E}_{\mathrm{D}}\left[\mathcal{I}_{\Phi_{\beta'_{\delta+1}}}[f]\right], \tag{6.20}$$

$$\mathbb{E}_{\mathrm{D}}\left[\mathcal{I}_{\Phi_{\beta'_{\delta+1}}}[f]\mathcal{I}_{\Phi_{\beta''_{\delta+1}}}[f]\right] = \frac{\mathbb{E}_{\mathrm{D}}\left[{\Phi_{\beta_\delta}}^2\right]}{\mathbb{E}_{\mathrm{D}}\left[\Phi_{\beta_\delta}\right]^2}\frac{\mathbb{E}_{\mathrm{D}}\left[p^{\beta_\delta}_{\beta'_{\delta+1}}p^{\beta_\delta}_{\beta''_{\delta+1}}\right]}{\mathbb{E}_{\mathrm{D}}\left[p^{\beta_\delta}_{\beta'_{\delta+1}}\right]\mathbb{E}_{\mathrm{D}}\left[p^{\beta_\delta}_{\beta''_{\delta+1}}\right]}\mathbb{E}_{\mathrm{D}}\left[\mathcal{I}_{\Phi_{\beta'_{\delta+1}}}[f]\right]\mathbb{E}_{\mathrm{D}}\left[\mathcal{I}_{\Phi_{\beta''_{\delta+1}}}[f]\right], \tag{6.21}$$

where the branch volume moments are

$$\mathbb{E}_{\mathrm{D}}\left[\Phi_{\beta_\delta}\right] = \mathbb{E}_{\mathrm{D}}\left[\Phi_0\right]\left(\prod_{\delta'=0}^{\delta}\mathbb{E}_{\mathrm{D}}\left[p^{\beta_{\delta'-1}}_{\beta_{\delta'}}\right]\mathbb{E}_{\mathrm{D}}\left[X^{\beta_{\delta'-1}}_{\beta_{\delta'}}\right]\right), \tag{6.22}$$

$$\mathbb{E}_{\mathrm{D}}\left[\Phi^2_{\beta_\delta}\right] = \mathbb{E}_{\mathrm{D}}\left[\Phi^2_0\right]\left(\prod_{\delta'=0}^{\delta}\mathbb{E}_{\mathrm{D}}\left[{p^{\beta_{\delta'-1}}_{\beta_{\delta'}}}^2\right]\mathbb{E}_{\mathrm{D}}\left[{X^{\beta_{\delta'-1}}_{\beta_{\delta'}}}^2\right]\right). \tag{6.23}$$

6.3.4 Bottom-up calculations

The volume of an edge, $\Phi^{\beta_\delta}_{\beta_{\delta+1}}$, can be expressed in terms of the volume of the branch it connects to and the configuration volume ratios,

$$\Phi^{\beta_\delta}_{\beta_{\delta+1}}(V_j^{\beta_{\delta+1}}) = \Phi_{\beta_{\delta+1}} \prod_{k=j}^{N_{\mathrm{NS}}^{\beta_{\delta+1}}} \frac{1}{t_k^{\beta_{\delta+1}}}, \tag{6.24}$$

and the volume of a branch can also be expressed in terms of its child edges and nodes,

$$\Phi_{\beta_\delta} = \sum_{\beta_{\delta+1}} \left(\Phi^{\beta_\delta}_{\beta_{\delta+1}} + \Phi_{\beta_{\delta+1}} \right). \tag{6.25}$$

The moments for the edge volumes can be calculated for the bottom up calculation,

$$\mathbb{E}_{\mathrm{D}} \left[\Phi^{\beta_\delta}_{\beta_{\delta+1}}(V_j^{\beta_{\delta+1}}) \right] = \mathbb{E}_{\mathrm{D}} \left[\Phi_{\beta_{\delta+1}} \right] \left(\prod_{k=j}^{N_{\mathrm{NS}}^{\beta_{\delta+1}}} \frac{n_k^{\beta_{\delta+1}}}{n_k^{\beta_{\delta+1}} - 1} \right), \tag{6.26}$$

$$\mathbb{E}_{\mathrm{D}} \left[\Phi^{\beta_\delta}_{\beta_{\delta+1}}(V_j^{\beta_{\delta+1}})^2 \right] = \mathbb{E}_{\mathrm{D}} \left[{\Phi_{\beta_{\delta+1}}}^2 \right] \left(\prod_{k=j}^{N_{\mathrm{NS}}^{\beta_{\delta+1}}} \left(\frac{n_k^{\beta_{\delta+1}}}{n_k^{\beta_{\delta+1}} - 2} \right) - 1 \right). \tag{6.27}$$

Hence the branch moments are

$$\mathbb{E}_{\mathrm{D}} \left[\Phi_{\beta_\delta} \right] = \sum_{\beta_{\delta+1}} \left(\mathbb{E}_{\mathrm{D}} \left[\Phi_{\beta_{\delta+1}} \right] \prod_{k=1}^{N_{\mathrm{NS}}^{\beta_{\delta+1}}} \left(\frac{n_k^{\beta_{\delta+1}}}{n_k^{\beta_{\delta+1}} - 1} \right) \right), \tag{6.28}$$

$$\mathbb{E}_{\mathrm{D}} \left[\Phi^2_{\beta_\delta} \right] = \sum_{\beta_{\delta+1}} \left(\mathbb{E}_{\mathrm{D}} \left[\Phi^2_{\beta_{\delta+1}} \right] \prod_{k=1}^{N_{\mathrm{NS}}^{\beta_{\delta+1}}} \left(\frac{n_k^{\beta_{\delta+1}}}{n_k^{\beta_{\delta+1}} - 2} \right) \right). \tag{6.29}$$

The integral eq. (2.22) can then be evaluated using equations eqs. (6.26) and (6.28). However, it is more useful to interpolate between the bottom-up and top-down calculations, as this enables the HSA to significantly enhance the accuracy of the density of states obtained by NBS, which we will discuss next.

6.3.5 Interpolating between the top-down and bottom-up calculations

The second moments calculated for the top-down and bottom-up procedures can be used to obtain a weighted sum of the two results that naturally incorporates the associated uncertainty with either calculation to produce the best overall estimate of the basin volumes. As the configuration volumes for both procedures are calculated by the product of a set of independently distributed variables, the overall configuration volume was calculated by a weighted sum of logarithms,

$$\begin{aligned}\mathbb{E}_{\mathrm{I}}\left[\ln\left(\Phi_{\beta_{\delta+1}}^{\beta_\delta}(V_j^{\beta_{\delta+1}})\right)\right] \\ \approx \frac{\left(\frac{\ln\left(\mathbb{E}_{\mathrm{D}}\left[\Phi_{\beta_{\delta+1}}^{\beta_\delta}(V_j^{\beta_{\delta+1}})\right]\right)}{w_{\mathrm{D}}(V_j^{\beta_{\delta+1}})}+\frac{\ln\left(\mathbb{E}_{\mathrm{U}}\left[\Phi_{\beta_{\delta+1}}^{\beta_\delta}(V_j^{\beta_{\delta+1}})\right]\right)}{w_{\mathrm{U}}(V_j^{\beta_{\delta+1}})}\right)}{\left(\frac{1}{w_{\mathrm{D}}(V_j^{\beta_{\delta+1}})}+\frac{1}{w_{\mathrm{U}}(V_j^{\beta_{\delta+1}})}\right)},\end{aligned} \tag{6.30}$$

where $\mathbb{E}_{\mathrm{D/U}}\left[\Phi_{\beta_{\delta+1}}^{\beta_\delta}(V_j^{\beta_{\delta+1}})\right]$ is the expected configuration volume [calculated top down (D) or bottom up (U)] of all the basins connected to $\beta_{\delta+1}$ up to an energy of $V_j^{\beta_{\delta+1}}$, and $\mathbb{E}_{\mathrm{I}}$ is the expectation value of the interpolation.

The chosen weighting was the logarithmic ratio of the second moment to the square of the first moment,

$$w_{\mathrm{U/D}}(V_j^{\beta_{\delta+1}}) = \ln\left(\frac{\mathbb{E}_{\mathrm{U/D}}\left[\Phi_{\beta_{\delta+1}}^{\beta_\delta}(V_j^{\beta_{\delta+1}})^2\right]}{\mathbb{E}_{\mathrm{U/D}}\left[\Phi_{\beta_{\delta+1}}^{\beta_\delta}(V_j^{\beta_{\delta+1}})\right]^2}\right), \tag{6.31}$$

which is an effective approximation to the uncertainty in the logarithmic volume. Though as the logarithmic volume can be expressed as the sum of a set of independent variables a direct calculation is relatively straightforward to perform.

Before the above calculation can be performed, the relative scale factor between the bottom-up and top down calculations needs to be determined. In theory this scale factor could be estimated by calculating the exact configuration volume for each harmonic basin, and then the volume available to the minimum within its constraints associated within its rotational and translational degrees of freedom. However, in this implementation of NBS it was found that instead, the relative factor could be determined by minimising the logarithmic difference between the

top-down and bottom-up basin volumes for each of the leaves, at the harmonic energy, as calculated in section 6.4.2, scaled by the sum of the weights, as calculated in eq. (6.31).

This scheme unfortunately does not smoothly connect the configuration volumes above and below a node, so to ensure a smooth interpolation, the value of $\mathbb{E}_{\mathrm{U}}\left[\Phi_{\beta_\delta}\right]$ can be adjusted slightly to ensure that

$$\mathbb{E}_{\mathrm{I}}\left[\Phi_{\beta_\delta}^{\beta_{\delta-1}}(V_{N_{\mathrm{NS}}}^{\beta_\delta})\right] = \sum_{\beta_{\delta+1}} \mathbb{E}_{\mathrm{I}}\left[\Phi_{\beta_{\delta+1}}^{\beta_\delta}(V_{N_{\mathrm{NS}}}^{\beta_\delta})\right]. \tag{6.32}$$

With the above caveats the trapezium rule can then be used to evaluate eq. (2.22) using eq. (6.30) on each edge and then summing the results.

6.4 Determining the disconnectivity graph

To calculate the configuration volume, as described in section 6.3, the NBS DG first needs to be known. While it may be possible to adapt the procedure used by Pártay et al. [110] to generate landscape charts, this approach to detecting disconnecting regions would require the configurations generated by the NS to be saved, which significantly increases the storage demands of the method and requires an appropriate similarity metric specific to the problem at hand.

An alternative approach was developed for this work, where different basins are merged together at the energy level above which the configuration volume estimated by NS for each basin looks identical, which avoids storing the configurations of the dead points and the specification of a problem specific metric.

This approach does not guarantee that the DG generated will accurately represent the true NBS DG as it only merges basins when the configuration volumes appear identical. However, the overall density of states produced should not be affected by the merge, ensuring that the method produces self-consistent results. The mergers primarily serve to decrease the *uncertainty* of the configuration volume estimates.

6.4.1 Comparing basin volumes

The configuration volume of two different basins can be compared by Bayesian model comparison, as described in section 2.5.3, where the hypothesis that the basin volumes are the same can be compared against the probability that they are different. Suppose there are two different basins, β_δ, β'_δ, connected to the same parent basin,

$\beta_{\delta-1}$. The basins should merge at the energy above which the density of states appear identical for both basins. To find this energy threshold the configuration volume ratio at two different energy levels, V_j, V_{j+1}, was modelled as a beta-distributed variable [see eq. (2.34)],

$$\frac{\Phi_{\beta_\delta}^{\beta_{\delta-1}}(V_j)}{\Phi_{\beta_\delta}^{\beta_{\delta-1}}(V_{j+1})} = t_{\beta_\delta}^{\beta_{\delta-1}}(V_j) \sim \mathcal{B}\left(a_{\beta_\delta}^{\beta_{\delta-1}}(V_j), b_{\beta_\delta}^{\beta_{\delta-1}}(V_j)\right), \tag{6.33}$$

whose parameters can be estimated from the first and second moments of the configuration volume using eqs. (2.38) and (2.39).

We can interpret $a_{\beta_\delta}^{\beta_{\delta-1}}(V_j)$ and $b_{\beta_\delta}^{\beta_{\delta-1}}(V_j)$ as binomial pseudocounts of uniformly sampled points in $\Phi_{\beta_\delta}^{\beta_{\delta-1}}$ with an energy cut-off of V_j, where $a_{\beta_\delta}^{\beta_{\delta-1}}(V_j)$ are the number of points observed to have energy greater than V_{j+1} and $b_{\beta_\delta}^{\beta_{\delta-1}}(V_j)$ less than. It is possible to generate true count data from the individual results from the NOpts runs in this region, but the statistical properties of these count results would not be as good.

The *maximum a posteriori* (*map*) estimate of the merge energy,

$$V_{\text{merge}}^{\beta_\delta=\beta'_\delta} = \arg\max_{V_j} \prod_{V_{j'}<=V_j} \Pr\left(t_{\beta_\delta}^{\beta_{\delta-1}}(V_{j'}) = t_{\beta'_\delta}^{\beta_{\delta-1}}(V_{j'})\right) \prod_{V_{j''}>V_j} \Pr\left(t_{\beta_\delta}^{\beta_{\delta-1}}(V_{j''}) \neq t_{\beta'_\delta}^{\beta_{\delta-1}}(V_{j'})\right), \tag{6.34}$$

of two branches, Φ_{β_δ} and $\Phi_{\beta'_\delta}$ can be obtained by modelling the fitted beta parameters of $t_{\beta_\delta}^{\beta_{\delta-1}}(V_j)$ and $t_{\beta'_\delta}^{\beta_{\delta-1}}(V_j)$ from eq. (6.33) as binomial pseudocounts and using eqs. (2.47) and (2.48). This approach allows us to self-consistently merge different edges on the NBS DG, as the edges only merge when their densities of states look the same. Using eq. (2.50) it is also possible to consider merging multiple edges simultaneously.

The above procedures for merging the different basins require an ordered list of energy levels to compare all the separate basins. This list was generated by choosing energy levels evenly spaced in the logarithm of the configuration volume of the aggregated runs for all the separate basins, so the configuration volume ratio would be approximately constant as the energy levels decreased. In this work a volume ratio of 0.5 was chosen, so V_j corresponds to the expected energy of a new live point generated with energy cut-off V_{j+1}.

6.4.2 Determining the harmonic energy range

To start the bottom-up procedure we must calculate the energy range over which a minimum, μ, can be treated as harmonic. Here a similar procedure was performed as described above. A set of energy levels, $V_j < V_{j+1}$, evenly spaced by the logarithm of the *harmonic configuration volume*, $t_{\text{harm}} = (V_j - V_\mu^Q)^{\kappa/2} \big/ (V_{j+1} - V_\mu^Q)^{\kappa/2}$. The probability that the NBS volume ratio and harmonic volume ratio are the same can be calculated,

$$\Pr\left(t_{\beta_\delta}^{\beta_{\delta-1}}(V_j) = t_{\text{harm}}\right) = \text{Bin}(a(V_j)|a(V_j) + b(V_j), t_{\text{harm}}), \tag{6.35}$$

and from eq. (2.48) the probability they are different,

$$\Pr\left(t_{\beta_\delta}^{\beta_{\delta-1}}(V_j) \neq t_{\text{harm}}\right) = \binom{a(V_j) + b(V_j)}{a(V_j)} B\left(a(V_j) + \alpha_{\text{prior}}, b(V_j) + \alpha_{\text{prior}}\right) \tag{6.36}$$

where we have dropped $a_{\beta_\delta}^{\beta_{\delta-1}}(V_j)$ from $a(V_j)$ and $b(V_j)$ for brevity, and α_{prior} is an appropriate prior for the beta distribution. The *map* estimate of the harmonic energy level,

$$V_\mu^{\text{harm}} = \arg\max_{V_j} \prod_{V_{j'} \leq V_j} \Pr\left(t_{\beta_\delta}^{\beta_{\delta-1}}(V_{j'}) = t_{\text{harm}}\right) \prod_{V_{j''} > V_j} \Pr\left(t_{\beta_\delta}^{\beta_{\delta-1}}(V_{j''}) \neq t_{\text{harm}}\right), \tag{6.37}$$

can then be found.

6.4.3 Local sampling close to a minimum

The chance of a random nested optimisation run finishing in a specific minimum will be extremely small for most of the minima, which means that the density of states will be rather uncertain before the basin has merged with other basins, as estimated by the NS at energies close to the minimum. To decrease this uncertainty the local basin of a minima can be sampled by performing traditional NS, except that all the new replicas are generated by random walks beginning at the minimum itself. This process may only work up to a certain energy level, as the random walks to generate new live points may cease to be ergodic. The *map* estimate can be calculated as in eq. (6.34), except that inequality signs are swapped, as we expect the density of states to diverge as the energy *increases*. We will refer to this process as local NS, to indicate that it samples only the section of the disconnectivity tree local to the minimum.

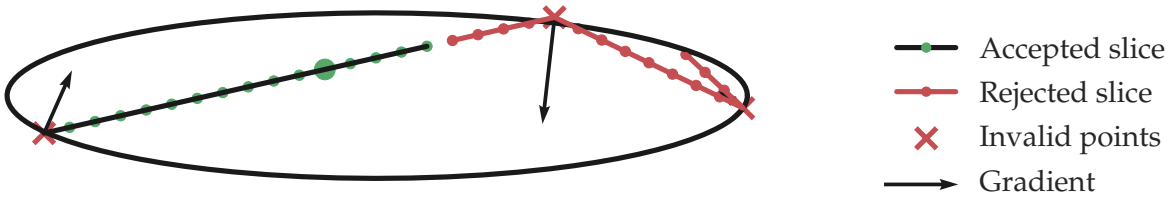


Fig. 6.3 A graphic showing how NoGUTS generates slices. The large green point shows the starting point. The red slice represents a proposed slice to double the length of the current slice. However, because the red slice has a u-turn present, the proposal to extend the slice is rejected and the algorithm quits.

6.5 The No Galilean U-Turn Sampler

Here we describe the No Galilean U-Turn Sampler (NoGUTS), which is a modification of the No U-Turn Sampler (NUTS) [91, 93] (see section 2.4.1) to work with Galilean sampling (see section 2.4.4) [104, 105]. As NoGUTS can be viewed as a form of multivariate reflective slice sampling [122] with an automatic stopping criterion, and as such we will refer to the trajectories generated by the algorithm as *slices*.

Several modifications have been made to the NUTS algorithm to make it work for the problem at hand.

- The leapfrog integration step has been replaced with the Galilean sampling equivalent step, which is described in algorithm 6.
- Each point, $\boldsymbol{R}'$, on a slice has a forward $\boldsymbol{p}'_+$ and backwards velocity $\boldsymbol{p}'_-$ associated with it, to account for reflections to the velocity at invalid points. This modification primarily affects the Galilean step, as described in algorithm 6, but also introduces some additional bookkeeping in the BuildTree and NoGUTS algorithms as described in algorithms 4 and 5, compared to NUTS.
- The ability to include constraints in the simulation has been incorporated, described in algorithm 7. Constraints can be most straightforwardly incorporated by rejecting any slice that violates the constraints with a simple test function TestConstraint($\boldsymbol{R}'$), which returns true if $\boldsymbol{R}'$ satisfies the constraints, and false otherwise. However, by introducing a 'constraint potential', $V_{\text{constraint}}(\boldsymbol{R})$, which is 0 for all configurations that satisfy the constraint and positive for invalid configurations, the NoGUTS simulation can reflect off the constraint boundaries in addition to the energy cut-off boundaries. In the case that the configuration encountered violates both the potential cut-off and the constraint, the algorithm as described reflects off the sum of the normalised gradient and

constraint gradient at that point. The NoGUTS simulation could also reflect off just the gradient or the constraint gradient and maintain detailed balance. It is straightforward to construct continuous constraint potentials for hard sphere constraints, by summing the excess radius of all the points that exceed the radius of the hard sphere.

- In NUTS the points to include in the trajectory are determined by a 1-D slice sampling process, however in the case of sampling from a hard constraint, all points that are valid can be straightforwardly included in the slice generated.
- The stopping criterion also needs to be slightly modified to account for the multivalued velocities, so for the positive direction the positive velocity must be used and vice versa for the negative direction.
- For atomic simulations rigid body motions of the atoms are spurious, so before the stopping criterion is calculated, any net linear and angular momentum of the velocity vectors can be subtracted without violating detailed balance.

6.5.1 Overview

Galilean sampling can be a very efficient method for exploring a hard constraint space as it allows long-range directed moves away from the starting point. The choice of simulation length for Galilean sampling is extremely important, as the reflective nature of the movement can cause the replica to start moving back to its starting point, significantly reducing the efficiency.

NoGUTS (and NUTS) enable the simulation to detect when this process occurs and stop, whilst *maintaining detailed balance*.

The algorithm works by recursively doubling a slice of points, or equivalently building a binary tree, until it reaches a stopping criterion, as shown in fig. 6.3. For each iteration the algorithm selects forwards or backwards directions randomly and then attempts to build a slice of equal length in that direction. If at any point of building the new slice the algorithm detects that the stopping criterion would have been satisfied then NoGUTS stops and rejects the new slice. It rejects this new slice because the probability of moving from it to the current slice would be zero, as a NoGUTS simulation starting from the new slice would terminate before adding the current slice. If the new slice is successfully added to the current slice then the stopping criterion can be tested again on the new combined slice, and then the process can be repeated. This procedure ensures that each valid point on the slice

Algorithm 4 The NoGUTS algorithm

Function: NoGUTS
Input: $\boldsymbol{R}$, δ, V, V_{cut}, j_{max}
Output: $\boldsymbol{R}, V', n_{\text{accept}}, n_{\text{reject}}$ ▷ The new live point generated by NoGUTS
$j_{\text{depth}} = 0$
$n_{\text{accept}} = 0, n_{\text{reject}} = 0$
$s_{\text{valid}} = \text{True}$
$\boldsymbol{R}^+ = \boldsymbol{R}^+ = \boldsymbol{R}$
$\boldsymbol{p}_+^+ = \boldsymbol{p}_+^- = \boldsymbol{p}_-^+ = \boldsymbol{p}_-^- \sim \mathcal{N}(0, \boldsymbol{I})$ ▷ Initialise random velocity
while s_{valid} AND $j_{\text{depth}} < j_{\text{max}}$ **do**
  if unif$(0, 1) < 0.5$ **then** ▷ Make proposal to double slice
    $\_, \_, \_, \boldsymbol{R}^-, \boldsymbol{p}_+^-, \boldsymbol{p}_-^-, \boldsymbol{R}', \boldsymbol{p}_+', \boldsymbol{p}_-', V', n'_{\text{accept}}, n'_{\text{reject}}, s'_{\text{valid}}$
      $=$ BuildTree$(\boldsymbol{R}^-, \boldsymbol{p}_+^-, \boldsymbol{p}_-^-, V_{\text{cut}}, -1, \delta, j_{\text{depth}})$ ▷ See algorithm 5
  else
    $\boldsymbol{R}^+, \boldsymbol{p}_+^+, \boldsymbol{p}_-^+, \_, \_, \_, \boldsymbol{R}', \boldsymbol{p}_+', \boldsymbol{p}_-', V', n'_{\text{accept}}, n'_{\text{reject}}, s'_{\text{valid}}$
      $=$ BuildTree$(\boldsymbol{R}^+, \boldsymbol{p}_+^+, \boldsymbol{p}_-^+, V_{\text{cut}}, 1, \delta, j_{\text{depth}})$
  end if
  if s'_{valid} AND unif$(0, 1) < n'_{\text{accept}}/n_{\text{accept}}$ **then** ▷ take point selected by new slice
    $\boldsymbol{R}_{\text{NoGUTS}} = \boldsymbol{R}'$
    $V_{\text{NoGUTS}} = V'$
  end if
  $n_{\text{accept}} = n_{\text{accept}} + n'_{\text{accept}}, n_{\text{reject}} = n_{\text{reject}} + n'_{\text{reject}}$
  $s_{\text{valid}} = s'_{\text{valid}}$ AND StopCriterion$(\boldsymbol{R}^+, \boldsymbol{p}_+^+, \boldsymbol{p}_-^+, \boldsymbol{R}^-, \boldsymbol{p}_+^-, \boldsymbol{p}_-^-)$
    ▷ Test whether proposal is valid or whether the new slice has performed a u-turn
  $j = j + 1$
end while

has an equal probability of generating an identical slice, preserving detailed balance. To ensure that the algorithm terminates in a reasonable time a maximum recursion depth, j_{depth}, can be specified.

The algorithm does not need to store all the valid points as the slice is generated. Instead it maintains for every sub-slice its associated selected point, which has been randomly chosen uniformly out of the valid points in that sub-slice. When joining two sub-slices the algorithm will randomly pick a selected point from one of the sub-slices with probability equal to the number of valid points in that sub-slice, ensuring that the selected point of the new slice has been selected uniformly from the union of valid points of for the pair [93].

This process of implicitly building the slice is performed by the recursive BuildTree function described in algorithm 5, which is called by the NoGUTS algorithm, de-

Algorithm 6 Galilean step

Function: GalileanStep
Input: $\boldsymbol{R}, \boldsymbol{p}_+, \boldsymbol{p}_-, V_{\text{cut}}, v_{\text{dir}}, \delta, V$ ▷ Start point, forwards velocity, backwards velocity, energy cut-off, direction, stepsize, potential
Output: $\boldsymbol{R}', \boldsymbol{p}'_+, \boldsymbol{p}'_-, V', n'$ ▷ New point, forwards velocity, backwards velocity, energy, and validity
if $v_{\text{dir}} = -1$ **then** ▷ Selecting appropriate velocity vector
  $\boldsymbol{p} = -\boldsymbol{p}_-$
else
  $\boldsymbol{p} = \boldsymbol{p}_+$
end if
$\boldsymbol{R}' = \boldsymbol{R} + \delta\boldsymbol{p}$
$V' = V(\boldsymbol{R}')$
$\boldsymbol{G}' = \nabla_{\boldsymbol{R}} V(\boldsymbol{R}')$
$r_{\text{energy}} = V' > V_{\text{cut}}$
if r_{energy} **then** ▷ New point not valid
  $n' = 0$ ▷ Reflect velocity
  $\boldsymbol{p}' = \boldsymbol{p} - 2\dfrac{\boldsymbol{G}' \cdot \boldsymbol{p}}{\boldsymbol{G}' \cdot \boldsymbol{G}'}\boldsymbol{G}'$
else ▷ New point is valid
  $n' = 1$
  $\boldsymbol{p}' = \boldsymbol{p}$
end if
if $v_{\text{dir}} = -1$ **then** ▷ Set new velocity vectors
  $\boldsymbol{p}'_+ = \boldsymbol{p}_-$
  $\boldsymbol{p}'_- = -\boldsymbol{p}'$
else
  $\boldsymbol{p}'_+ = \boldsymbol{p}'$
  $\boldsymbol{p}'_- = \boldsymbol{p}_+$
end if

Algorithm 7 Constrained Galilean step

Function: ConsGalileanStep
Input: $\boldsymbol{R}, \boldsymbol{p}_+, \boldsymbol{p}_-, V_{\text{cut}}, v_{\text{dir}}, \delta, V, V_{\text{constraint}}$
Output: $\boldsymbol{R}', \boldsymbol{p}'_+, \boldsymbol{p}'_-, V', n'$
if $v_{\text{dir}} = -1$ **then**
  $\boldsymbol{p} = -\boldsymbol{p}_-$
else
  $\boldsymbol{p} = \boldsymbol{p}_+$
end if
$\boldsymbol{R}' = \boldsymbol{R} + \delta\boldsymbol{p}$
$V' = V(\boldsymbol{R}')$
$\boldsymbol{G}' = \nabla_{\boldsymbol{R}} V(\boldsymbol{R}')$
$V'_{\text{constraint}} = V_{\text{constraint}}(\boldsymbol{R}')$
$\boldsymbol{G}'_{\text{constraint}} = \nabla_{\boldsymbol{R}} V_{\text{constraint}}(\boldsymbol{R}')$
$r_{\text{energy}} = V' > V_{\text{cut}}$
$r_{\text{constraint}} = V'_{\text{constraint}} > 0$
if r_{energy} OR $r_{\text{constraint}}$ **then**
  if r_{energy} AND NOT $r_{\text{constraint}}$ **then**
    $\boldsymbol{G}'' = \boldsymbol{G}'$
  else if NOT r_{energy} AND $r_{\text{constraint}}$ **then**
    $\boldsymbol{G}'' = \boldsymbol{G}'_{\text{constraint}}$
  else if r_{energy} AND $r_{\text{constraint}}$ **then** ▷ Reflect off both gradients
    $\boldsymbol{G}'' = \dfrac{\boldsymbol{G}'}{|G'|} + \dfrac{\boldsymbol{G}'_{\text{constraint}}}{|G'_{\text{constraint}}|}$
  end if
  $n' = 0$
  $\boldsymbol{p}' = \boldsymbol{p} - 2\dfrac{\boldsymbol{G}'' \cdot \boldsymbol{p}}{\boldsymbol{G}'' \cdot \boldsymbol{G}''}\boldsymbol{G}''$
else
  $n' = 1$
  $\boldsymbol{p}' = \boldsymbol{p}$
end if
if $v_{\text{dir}} = -1$ **then**
  $\boldsymbol{p}'_+ = \boldsymbol{p}_-$
  $\boldsymbol{p}'_- = -\boldsymbol{p}'$
else
  $\boldsymbol{p}'_+ = \boldsymbol{p}'$
  $\boldsymbol{p}'_- = \boldsymbol{p}_+$
end if

Algorithm 5 BuildTree

Function: BuildTree
Input: $\boldsymbol{R}, \boldsymbol{p}_+, \boldsymbol{p}_-, V_{\text{cut}}, v_{\text{dir}}, \delta, j_{\text{depth}}, V$
Optional: $V_{\text{constraint}}$, TestConstraints
Output: $\boldsymbol{R}^+, \boldsymbol{p}_+^+, \boldsymbol{p}_-^+, \boldsymbol{R}^-, \boldsymbol{p}_+^-, \boldsymbol{p}_-^-, \boldsymbol{R}', \boldsymbol{p}_+, \boldsymbol{p}'_-, V', n_{\text{accept}}, n_{\text{reject}}, s_{\text{valid}}$
if $j_{\text{depth}} = 0$ **then**
  if using constraint potential **then** ▷ See algorithm 7
    $\boldsymbol{R}', \boldsymbol{p}'_+, \boldsymbol{p}'_-, V', n' = \text{ConsGalileanStep}(\boldsymbol{R}, \boldsymbol{p}_+, \boldsymbol{p}_-, V_{\text{cut}}, v_{\text{dir}}, \delta, V, V_{\text{constraint}})$
  else ▷ See algorithm 6
    $\boldsymbol{R}', \boldsymbol{p}'_+, \boldsymbol{p}'_-, V', n' = \text{GalileanStep}(\boldsymbol{R}, \boldsymbol{p}_+, \boldsymbol{p}_-, V_{\text{cut}}, v_{\text{dir}}, \delta, V)$
  end if
  $\boldsymbol{R}^+, \boldsymbol{p}_+^+, \boldsymbol{p}_-^+ = \boldsymbol{R}', \boldsymbol{p}'_+, \boldsymbol{p}'_-$
  $\boldsymbol{R}^-, \boldsymbol{p}_+^-, \boldsymbol{p}_-^- = \boldsymbol{R}', \boldsymbol{p}'_+, \boldsymbol{p}'_-$
  $s_{\text{valid}} = \text{TestConstraints}(\boldsymbol{R}')$ ▷ Stop if constraint broken
else ▷ Recursively build binary tree
  $\boldsymbol{R}^+, \boldsymbol{p}_+^+, \boldsymbol{p}_-^+, \boldsymbol{R}^-, \boldsymbol{p}_+^-, \boldsymbol{p}_-^-, \boldsymbol{R}', \boldsymbol{p}'_+, \boldsymbol{p}'_-, V', n'_{\text{accept}}, n'_{\text{reject}}, s'_{\text{valid}}$
    $= \text{BuildTree}(\boldsymbol{R}, \boldsymbol{p}_+, \boldsymbol{p}_-, V_{\text{cut}}, v_{\text{dir}}, \delta, j_{\text{depth}} - 1)$
  if s'_{valid} **then**
    if $v_{\text{dir}} = -1$ **then**
      $\_, \_, \_, \boldsymbol{R}^-, \boldsymbol{p}_+^-, \boldsymbol{p}_-^-, \boldsymbol{R}'', \boldsymbol{p}''_+, \boldsymbol{p}''_-, V'', n''_{\text{accept}}, n''_{\text{reject}}, s''_{\text{valid}}$
        $= \text{BuildTree}(\boldsymbol{R}^-, \boldsymbol{p}_+^-, \boldsymbol{p}_-^-, V_{\text{cut}}, v_{\text{dir}}, \delta, j_{\text{depth}} - 1)$
    else
      $\boldsymbol{R}^+, \boldsymbol{p}_+^+, \boldsymbol{p}_-^+, \_, \_, \_, \boldsymbol{R}'', \boldsymbol{p}''_+, \boldsymbol{p}''_-, V'', n''_{\text{accept}}, n''_{\text{reject}}, s''_{\text{valid}}$
        $= \text{BuildTree}(\boldsymbol{R}^+, \boldsymbol{p}_+^+, \boldsymbol{p}_-^+, V_{\text{cut}}, v_{\text{dir}}, \delta, j_{\text{depth}} - 1)$
    end if
    $n_{\text{accept}} = n'_{\text{accept}} + n''_{\text{accept}}$
    $s_{\text{valid}} = s''_{\text{valid}}$ AND $\text{StopCriterion}(\boldsymbol{R}^+, \boldsymbol{p}_+^+, \boldsymbol{R}^-, \boldsymbol{p}_-^-)$
    if $\text{unif}(0, 1) < \frac{n''_{\text{accept}}}{n_{\text{accept}}}$ **then** ▷ Implicitly join slices together
      $\boldsymbol{R}', V', \boldsymbol{p}'_+, \boldsymbol{p}'_- = \boldsymbol{R}'', V'', \boldsymbol{p}''_+, \boldsymbol{p}''_-$
    end if
  end if
end if

scribed in algorithm 4, to progressively double a slice until the stopping criterion is reached. In these algorithms the function $\text{unif}(a, b)$ uniformly generates a random real number between a and b.

6.6 Adapting the stepsize

During the course of a NOpt run the energy cut-off and valid region of space will vary drastically, so the optimal stepsize to maintain an efficient acceptance rate will tend to decrease by several orders of magnitude during the course of the run. To cope with this variation we can define a simple logistic model for predicting the acceptance probability, p_{acc}, for a given stepsize, δ, at a given energy cut-off, V_{cut},

$$p_{\text{acc}}(\delta, V_{\text{cut}}) = \frac{1}{1 + \exp\left(m_{\text{acc}} V_{\text{cut}} + c_{\text{acc}}\right)\delta}. \tag{6.38}$$

Rearranging we find

$$m_{\text{acc}} V_{\text{cut}} + c_{\text{acc}} = \operatorname{logit} p_{\text{acc}} + \ln \delta, \tag{6.39}$$

where $\operatorname{logit} x = \ln x - \ln(1 - x)$. This suggests that the appropriate stepsize for a NoGUTS simulation can be chosen by performing an appropriate linear fit to the previous simulation results. Suppose during a simulation with cut-off energy V_t, and stepsize δ_t, that n_t^{acc} moves finish below V_t and n_t^{rej} finish above V_t. Then we can calculate the expected value,

$$\mathbb{E}\left[\operatorname{logit} p_{\text{acc}}(\delta_t, V_t)\right] = \psi_0(n_t^{\text{acc}}) - \psi_0(n_t^{\text{rej}}), \tag{6.40}$$

by modelling $p_{\text{acc}} \sim \mathcal{B}(n_{\text{accept}}, n_{\text{reject}})$, where $\psi_0(x) = d \ln(\Gamma(x))/\, dx$ is the digamma function. This model enables us to predict an appropriate step size for a given energy cut-off.

6.6.1 Avoiding non-Markovian dynamics

If the stepsize of a MCMC simulation is adjusted without due care the simulation may cease to be Markovian [120, 121]. However, during an NOpt the stepsize must be adjusted quite drastically, as the energy cut-off decreases to maintain an efficient acceptance rate.

One method to significantly reduce any sampling artifacts generated by adapting the stepsize is to introduce a delay, $N_{\text{delay}}^{\text{opt}}$, to incorporating the accept/reject statistics to the above model, so that the replica in NBS has time to completely move away from the regions used to determine the optimal stepsize.

Additionally, to avoid biasing this model with high energy points, a rolling window of length $N_{\text{window}}^{\text{opt}}$ can be applied, so that only the last $N_{\text{window}}^{\text{opt}}$ dead points

generated (and the associated lag introduced by the delay) are used to choose the stepsize for the NoGUTS simulation.

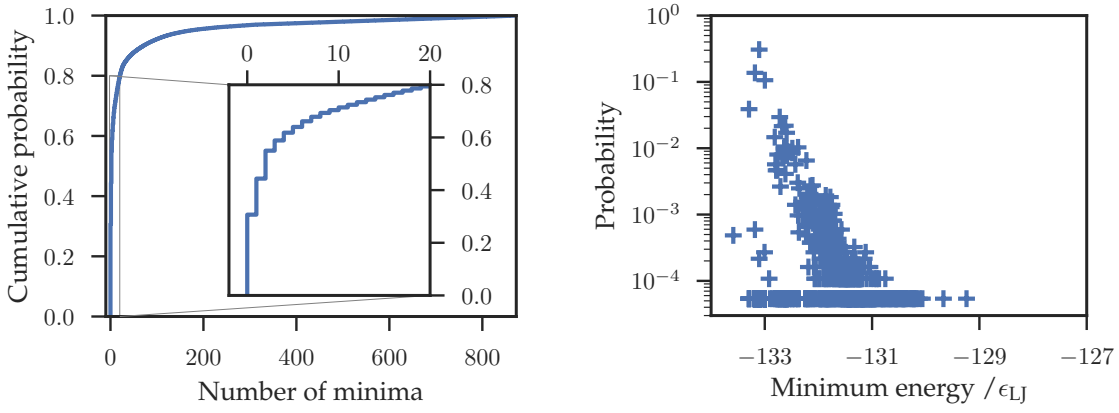
6.7 Results

LJ clusters have been extensively studied at a range of sizes [33, 87, 89, 94, 123], which makes them useful model systems for benchmarking various methods.

LJ_{31} has been extensively studied by a variety of different approaches, as it is the smallest LJ cluster to exhibit a complex heat capacity, with a solid-solid peak at low temperatures and a solid-liquid peak at higher temperatures. To accurately replicate both thermodynamic features requires effective sampling, both at the lowest energies to accurately replicate the solid-solid peak, and at higher energies to replicate the solid-liquid peak.

To avoid evaporation of the cluster the atoms were constrained to stay within a sphere of radius 2.5 σ_{LJ}, as in previous studies [89, 94]. 20,000 independent NOpts were performed. In addition, for each of the 4 lowest energy minima, local NS, as described in section 6.4.3, was performed with 1000 live points.

Each new live point was generated by 20 iterations of NoGUTS with a max tree depth of $j_{max} = 8$. Overall angular and linear momentum were removed from the



(a) Cumulative probability

(b) Probability versus energy

Fig. 6.4 In (a) the cumulative probability of a nested optimisation ending in a given range of the most likely minima is shown for LJ_{31}. The inset shows a magnification of the cumulative probabilities of the 50 most likely minima. In (b) the probability of landing in a specific minimum is plotted against the energy of that minimum in NS runs for LJ_{31}.

velocities before evaluating the NoGUTS stopping criterion. The target acceptance ratio was chosen to be $p_{\text{acc}} = 0.5$, the optimal stepsize was determined over a window of $N^{\text{opt}}_{\text{window}} = 100$ points, with a delay of $N^{\text{opt}}_{\text{delay}} = 20$ iterations to avoid non-Markovian behaviour. The simulation was stopped when the energy difference between the live point and the dead point from $N^{\text{opt}}_{\text{stop}} = 10$ iterations previously was less than $V^{\text{opt}}_{\text{tol}} = 0.1\,\epsilon_{\text{LJ}}$.

For comparison, a calculation of the heat capacity using standard NS with 20,000 live points was also performed. The live points were generated using NoGUTS with the same parameters as for the NBS calculation. This NS simulation was performed using the software developed by Martiniani et al. [94]. The simulation was stopped when the energy difference of the live points was less than 0.1ϵ.

The NBS simulation overall generated 11×10^6 live points using 3×10^{10} energy gradient calculations. The NS simulation generated 10×10^6 live points using 3.5×10^{10} energy gradient calculations.

The runs generated by the local NBS were found to be indistinguishable from the runs generated by standard NBS at all energies when performing the *maximum a posteriori* calculation of the merge energy.

6.7.1 Distribution of minima

It is interesting to analyse the distribution of minima generated by 20,000 NOpts, as shown in figs. 6.4a and 6.4b. Only 9 of the runs landed in the global minimum, whereas 30.6% of the NOpt runs landed in just a single minimum ($V_\mu = -133.1\,\epsilon_{\text{LJ}}$); 79.4% of the runs landed in just 20 of the minima; and during 20,000 minimisations the nested optimisations, only 873 distinct minima were found, whilst the actual number of minima for LJ_{31} has been estimated as approximately 10^{15}, excluding permutation-inversion isomers [89].

This structure is remarkably different from performing standard minimisations on LJ_{31}, and suggests there might be ways of associating most of the LJ_{31} PES of interest with a very small number of minima.

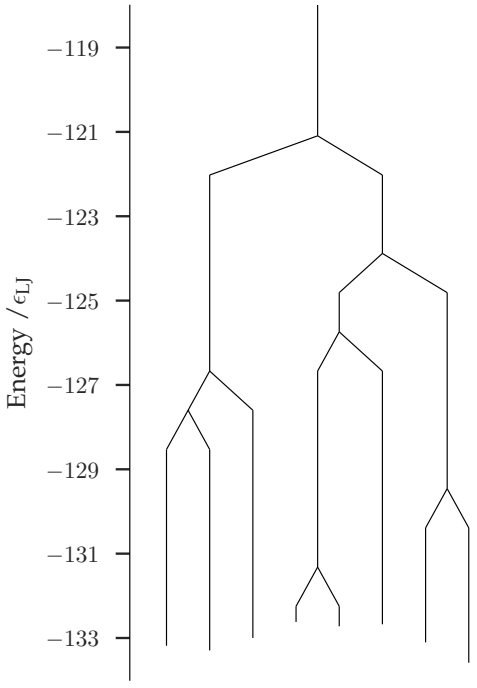


Fig. 6.5 The NBS disconnectivity graph for LJ_{31} with 20,000 NOpts.

It is not immediately obvious what drives this difference, but it seems that the leaves associated with most minima do not make a meaningful contribution to the overall configuration volume as opposed to the branches associated with the minima.

6.7.2 Disconnectivity graph

The NBS results were used to construct a NBS DG, which is shown in fig. 6.5. The maximum entropy prior, $\alpha_{\text{prior}} = 0.5$ was chosen when determining the merge energies.

6.7.3 Heat capacity

Using the DG illustrated in section 6.7.2 the heat capacity of LJ_{31} was calculated using NBS and is shown in fig. 6.6. For comparison results generated by the equivalent

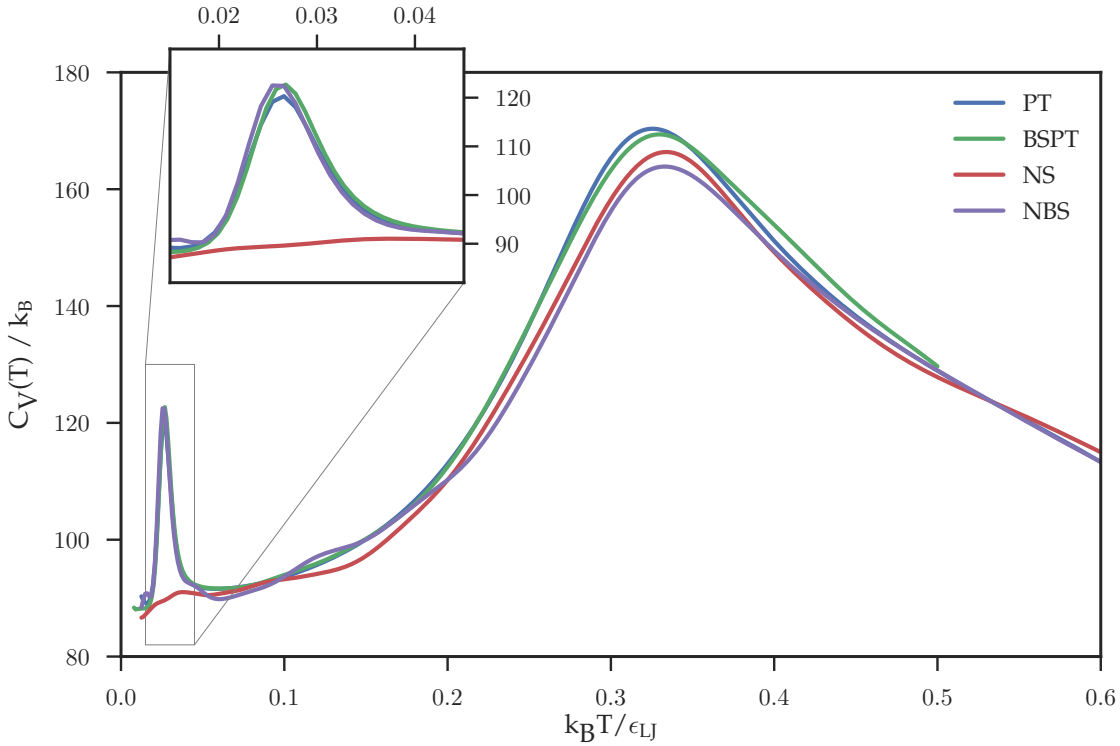


Fig. 6.6 The heat capacity of LJ_{31}, as calculated by NBS with 20,000 NOpts and 1,000 local NOpts on each of the 4 lowest energy minima. For comparison results from a standard NS simulation with 20,000 live points and a previous study using BSPT and PT [89] are shown. Each new live point for the NS and NBS simulation was generated using NoGUTS with $j_{\max} = 8$. Inset is a magnification of the low temperature solid-solid peak.

calculation for the standard NS simulation with 20,000 live points and a previous study using BSPT and PT are also illustrated.

Both the NBS and NS simulations show very similar high temperature solid-liquid heat capacity peaks, though both are slightly lower than the peaks as calculated by PT and BSPT. The NBS results closely match the low temperature solid-solid peak calculated by PT and BSPT. The standard NS simulation failed to find the lowest energy minimum and so fails to reproduce the lowest temperature peak. At higher temperatures the NBS and PT heat capacity curves match extremely well.

Code

The python and Fortran code that was used to perform these calculations can be found at:

https://github.com/matthewghgriffiths/nestedbasinsampling

7 Conclusions and Further Work

7.1 Alignment algorithms

In this work we have shown that it is possible to estimate the RMSD between structures by calculating the maximum kernel correlation, so long as the interatomic separation is relatively large compared to the kernel size. We then demonstrated that the FASTOVERLAP algorithm can find the maximum value of the overlap in periodic and isolated systems efficiently and deterministically using FFTs and SOFTs. Additionally, we have shown that it is possible to calculate the true RMSD of a system deterministically in a manner that scales polynomially with the number of atoms and RMSD, using the branch and bound algorithm, Go-PERMDIST. The correct RMSD is often obtained when a early exit condition is applied.

For periodic systems FASTOVERLAP performs particularly well, and scales favourably with both system and database size for multiple alignment tasks. The algorithm reliably identifies the optimal alignment for pairs of structures that are reasonably close together. For more distant configurations the performance degrades as the assumptions underlying the derivation of the algorithm break down. However, for these structures it is likely that finding the minimum RMSD is less critical in applications. For periodic systems the Go-PERMDIST algorithm is significantly slower than the FASTOVERLAP algorithm.

For isolated clusters of atoms the FASTOVERLAP algorithm performs less well, while the PERMDIST and Go-PERMDIST procedures are relatively effective and efficient. The Go-PERMDIST algorithm shows comparable or better performance than PERMDIST with random restarts when using the early exit condition.

7.1.1 Recommended usage of alignment methods

If run time is not critical we recommend using the Go-PERMDIST algorithm, which is guaranteed to find the best RMSD for both periodic systems and clusters given enough time. If the run time is important, then the FASTOVERLAP algorithm should be used for periodic systems and the Go-PERMDIST algorithm with an early exit condition could be used for clusters. For biomolecules it is likely that other methods, for example LPERMDIST [26], which take into consideration the local structure of the molecule by permuting local groups of atoms, will be more effective.

7.1.2 Further work

In certain applications we seek only the closest structures to a given target structure from a large database. In this situation it will be possible to adapt the Go-PERMDIST method to simultaneously align over the database and to quit once it has found the closest structures, instead of aligning the target structure with every member of the database.

When modelling the growth of clusters or mutations of proteins, it can be useful to align structures with different numbers of atoms. The Go-PERMDIST algorithm could be modified to perform this alignment, but the translational component would also have to be considered, in addition to the rotational alignment, as superimposing the centroids of the two structures no longer results in the optimal alignment. This step could be achieved by including translation alignment as in the Go-ICP method [53, 54].

It may also be possible to improve the performance of the PERMDIST algorithm using BH global optimisation [12–14] and taking smaller, non-random, rotational steps. Care would be required to ensure that the procedure does not get stuck in a local minimum.

When aligning very large databases of structures it can be prohibitively expensive to align every possible pair. The FASTOVERLAP method allows the RMSD between structures to be estimated quickly as an alternative metric. It may be possible in the future to develop diagnostic statistics that could be used to give an indication whether FASTOVERLAP has found the optimal alignment or determine whether there is a better kernel width for a particular pair of structures.

The maximal kernel overlap found by FASTOVERLAP may also be useful when only the similarity between two structures is needed, for example when comparing configurations with different numbers of atoms. It also may be possible to gen-

eralise the calculation of the SO(3) Fourier coefficients to allow optimisation over translations in addition to rotations.

The FASTOVERLAP method could also be applied as an alternative or in addition to SOAP when calculating the covariance between local atomic environments in the Gaussian approximation potentials framework [70, 71].

7.2 Nested basin-sampling

Many schemes that have been developed to calculate equilibrium thermodynamic properties require parallelisation to function efficiently [33, 87, 89, 94, 123]. In this work we present a new method, nested basin-sampling (NBS), which proceeds by performing a set of embarrassingly parallel nested optimisations (NOpts) whose results can be combined after the simulations end.

By splitting the configuration volume into separate regions, the calculation can provide a more detailed understanding of the structure of the energy landscape, and how the global thermodynamic properties are encoded. The harmonic approximation can be employed to enhance the accuracy at low temperatures, by separating the configuration volume into disconnected regions.

NBS was used to calculate the heat capacity of LJ_{31}, a benchmark system exhibiting broken ergodicity [89]. The heat capacity as calculated using NBS shows strong agreement with other methods, and compares favourably against NS. It was able to successfully resolve the low temperature solid-solid heat capacity peak, which standard NS missed, when performed with a comparable number of live points and energy gradient calculations.

The close agreement with the previous results suggest that the stepsize adjustment scheme, combined with NoGUTS are sufficient to ensure that the results generated by the NOpts are generating sufficiently unbiased samples from the PES of LJ_31.

7.2.1 Further work

There are several directions for future work.

- Due to the embarrassingly parallel nature of the NBS calculation, this approach can tackle much larger systems, where equilibrium is usually difficult to achieve.

- The method in its current form is still fairly inefficient compared to BSPT. However, there are many avenues that could be explored to increase its efficiency, particularly when combined with its local sampling scheme. It is likely that good results can still be achieved with a smaller number of NOpts, and some preliminary work suggests that reasonable heat capacity curves can be generated using just local sampling.
- It should be possible to enhance the results generated by local sampling using configurations generated by previous local sampling NOpts as starting points, instead of the minimum.
- It should also be possible to assign contributions to the heat capacity to specific parts of the NBS disconnectivity graph by extending the scheme that was recently applied to results obtained with BSPT [124].
- It should be possible to relate the NBS DG to the DG obtained by generating a transition state network.
- Since NBS partitions the PES into a set of separate regions it is possible to quantify which regions in configuration space have been poorly sampled, further prioritise sampling in those regions, and also provide better measures of convergence for the simulation.
- There are a variety of hyperparameters, p_{acc}, j_{depth}, $N_{\mathrm{delay}}^{\mathrm{opt}}$, $N_{\mathrm{MC}}^{\mathrm{opt}}$, $N_{\mathrm{stop}}^{\mathrm{opt}}$, $N_{\mathrm{window}}^{\mathrm{opt}}$, and $V_{\mathrm{tol}}^{\mathrm{opt}}$, that need to be chosen before beginning a NBS calculation. It is important to quantify how the choice of these hyperparameters affects the overall results and efficiency of the NBS simulation.
- The properties of the NoGUTS sampler could be explored in more detail, in particular how the target acceptance rate affects the overall efficiency of the method.

References

[1] Wales, D. J. *Energy Landscapes*; Cambridge University Press: Cambridge, 2003.

[2] Skilling, J. *Bayesian Anal.* **2006**, *1*, 833–860.

[3] Desautels, T.; Calvert, J.; Hoffman, J.; Jay, M.; Kerem, Y.; Shieh, L.; Shimabukuro, D.; Chettipally, U.; Feldman, M. D.; Barton, C.; Wales, D. J.; Das, R. *JMIR Med. Informatics* **2016**, *4*, e28.

[4] Desautels, T.; Das, R.; Calvert, J.; Trivedi, M.; Summers, C.; Wales, D. J.; Ercole, A. *BMJ Open* **2017**, *in press*.

[5] Das, R.; Wales, D. J. *R. Soc. Open Sci.* **2017**, *0*, 0.

[6] Ballard, A. J.; Das, R.; Martiniani, S.; Mehta, D.; Sagun, L.; Stevenson, J. D.; Wales, D. J. *Phys. Chem. Chem. Phys.* **2017**, *19*, 12585–12603.

[7] Frenkel, D.; Wales, D. J. *Nat. Mater.* **2011**, *10*, 410–411.

[8] Chakrabarti, D.; Fejer, S. N.; Wales, D. J. *Computational Nanoscience*; The Royal Society of Chemistry, 2011; pp 58–81.

[9] Stillinger, F. H.; Weber, T. A. *Phys. Rev. A* **1982**, *25*, 978–989.

[10] Stillinger, F. H.; Weber, T. A. *Science* **1984**, *225*, 983.

[11] Wales, D. J.; Doye, J. P. K. *J. Chem. Phys.* **2003**, *119*, 12409–12416.

[12] Li, Z.; Scheraga, H. A. *Proc. Natl. Acad. Sci. USA* **1987**, *84*, 6611.

[13] Li, Z.; Scheraga, H. A. *J. Mol. Struct.* **1988**, *179*, 333.

[14] Wales, D.; Doye, J. *J. Phys. Chem. A* **1998**, *101*, 5111–5116.

[15] Wales, D. J. *Mol. Phys.* **2002**, *100*, 3285.

[16] Wales, D. J. *Mol. Phys.* **2004**, *102*, 891.

[17] Wales, D. J. *Int. Rev. Phys. Chem.* **2006**, *25*, 237.

[18] Laidler, K. J. *Chemical Kinetics*; Harper & Row: New York, 1997.

[19] Dellago, C.; Bolhuis, P. G.; Chandler, D. *J. Chem. Phys.* **1998**, *108*, 9236.

[20] Rao, F.; Caflisch, A. *J. Mol. Biol.* **2004**, *342*, 299.

[21] Noé, F.; Krachtus, D.; Smith, J. C.; Fischer, S. *J. Chem. Theory Comput.* **2006**, *2*, 840.

[22] Noé, F.; Fischer, S. *Curr. Op. Struct. Biol.* **2008**, *18*, 154.

[23] Prada-Gracia, D.; Gómez-Gardenes, J.; Echenique, P.; Fernando, F. *PLoS Comput. Biol.* **2009**, *5*, e1000415.

[24] Wales, D. J. *Curr. Op. Struct. Biol.* **2010**, *20*, 3.

[25] Carr, J. M.; Trygubenko, S. A.; Wales, D. J. *J. Chem. Phys.* **2005**, *122*, 234903.

[26] Wales, D. J.; Carr, J. M. *J. Chem. Theory Comput.* **2012**, *8*, 5020.

[27] Bauer, M. S.; Strodel, B.; Fejer, S. N.; Koslover, E. F.; Wales, D. J. *J. Chem. Phys.* **2010**, *132*, 054101.

[28] Maxwell, J. C. *London, Edinburgh, Dublin Philos. Mag. J. Sci. 4th Ser.* **1870**, *40*, 421–426.

[29] Wales, D. J. *Mol. Phys.* **1993**, *78*, 151–171.

[30] Stillinger, F. H. *Science* **1995**, *267*, 1935.

[31] Strodel, B.; Wales, D. J. *Chem. Phys. Lett.* **2008**, *466*, 105–115.

[32] Sharapov, V. A.; Meluzzi, D.; Mandelshtam, V. A. *Phys. Rev. Lett.* **2007**, *98*, 105701.

[33] Sharapov, V. A.; Mandelshtam, V. A. *J. Phys. Chem. A* **2007**, *111*, 10284–10291.

[34] Verma, J.; Khedkar, V.; Coutinho, E. *Curr. Top. Med. Chem.* **2010**, *10*, 95.

[35] Ramakrishnan, R.; Dral, P. O.; Rupp, M.; von Lilienfeld, O. A. *J. Chem. Theory Comput.* **2015**, *11*, 2087–2096.

[36] Huo, H.; Rupp, M. *arXiv preprint arXiv:1704.06439* **2017**, Accessed: 2017-08-22.

[37] Rosenbrock, C. W.; Homer, E. R.; Csányi, G.; Hart, G. L. W. *arXiv preprint arXiv:1703.06236* **2017**, Accessed: 2017-08-22.

[38] Bartók, A. P.; Csányi, G. *Int. J. Quantum Chem.* **2015**, *115*, 1051–1057.

[39] Jonker, R.; Volgenant, A. *Computing* **1987**, *38*, 325.

[40] Sadeghi, A.; Ghasemi, S. A.; Schaefer, B.; Mohr, S.; Lill, M. A.; Goedecker, S. *J. Chem. Phys.* **2013**, *139*, 184118.

[41] De, S.; Bartók, A. P.; Csányi, G.; Ceriotti, M. *Phys. Chem. Chem. Phys.* **2016**, *18*, 13754.

[42] Salvi, J.; Matabosch, C.; Fofi, D.; Forest, J. *Image Vis. Comput.* **2007**, *25*, 578.

[43] Hill, D. L. G.; Hawkes, D. J.; Crossman, J. E.; Gleeson, M. J.; Cox, T. C. S.; Bracey, E. E. C. M. L.; Strong, A. J.; Graves, P. *Br. J. Radiol.* **1991**, *64*, 1030.

[44] Fitzgibbon, A. W. *Image Vis. Comput.* **2003**, *21*, 1145.

[45] Besl, P.; McKay, N. D. *IEEE Trans. Pattern Anal. Mach. Intell.* **1992**, *14*, 239.

[46] Hundt, R.; Schön, J. C.; Jansen, M. *J. Appl. Crystallogr.* **2006**, *39*, 6.

[47] Kuhn, H. W. *Nav. Res. Logist. Q.* **1955**, *2*, 83.

[48] Coutsias, E. A.; Seok, C.; Dill, K. A. *J. Comput. Chem.* **2004**, *25*, 1849.

[49] Kabsch, W. *Acta Crystallogr. Sect. A* **1978**, *34*, 827.

[50] Schaefer, B.; Goedecker, S. *J. Chem. Phys.* **2016**, *145*, 034101.

[51] Granger, S.; Pennec, X. *Eur. Conf. Comput. Vis. (ECCV 2002), Vol. 2353 LNCS* **2002**, 418.

[52] Li, H.; Hartley, R. The 3D-3D Registration Problem Revisited. Proc. IEEE Int. Conf. Comput. Vis. 2007; p 1.

[53] Yang, J.; Li, H.; Jia, Y. *Proc. IEEE Int. Conf. Comput. Vis.* **2013**, 1457.

[54] Yang, J.; Li, H.; Campbell, D.; Jia, Y. *IEEE Trans. Pattern Anal. Mach. Intell.* **2016**, *38*, 2241.

[55] Tsin, Y.; Kanade, T. In *Computer Vision - ECCV 2004: 8th European Conference on Computer Vision, Prague, Czech Republic, May 11-14, 2004. Proceedings, Part III*; Pajdla, T., Matas, J., Eds.; Springer Berlin Heidelberg: Berlin, Heidelberg, 2004; p 558.

[56] Horn, B. K. P. *Proc. IEEE* **1984**, *72*, 1671.

[57] Makadia, A.; Patterson, A. I.; Daniilidis, K. *2006 IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. - Vol. 1* **2006**, *1*, 1297.

[58] Kostelec, P. J.; Rockmore, D. N. *J. Fourier Anal. Appl.* **2008**, *14*, 145.

[59] Comin, M.; Guerra, C.; Dellaert, F. *J. Comput. Biol.* **2009**, *16*, 1577.

[60] Padhorny, D.; Kazennov, A.; Zerbe, B. S.; Porter, K. A.; Xia, B.; Mottarella, S. E.; Kholodov, Y.; Ritchie, D. W.; Vajda, S.; Kozakov, D. *Proc. Natl. Acad. Sci.* **2016**, *113*, E4286–E4293.

[61] Hong, E.-J.; Lee, K.-H.; Wenzel, W. RMSD computation for clusters of identical particles. Proc. 4th WSEAS Conf. Math. Biol. 2008; p 46.

[62] Wales, D. J. GMIN: A program for basin-hopping global optimisation, basin-sampling, and parallel tempering. http://www-wales.ch.cam.ac.uk/software.html, Accessed: 2017-08-22.

[63] Wales, D. J. OPTIM: A program for geometry optimisation and pathway calculations. http://www-wales.ch.cam.ac.uk/software.html, Accessed: 2017-08-22.

[64] de Souza, V. K.; Wales, D. J. *J. Stat. Mech.* **2016**, *2016*, 074001.

[65] Hundt, R. KPLOT: A Program for Plotting and Investigation of Crystal Structures. University of Bonn, Germany.

[66] Hundt, R.; Schön, J. C.; Neelamraju, S.; Zagorac, J.; Jansen, M. *J. Appl. Crystallogr.* **2013**, *46*, 587.

[67] Bartók, A. P.; Kondor, R.; Csányi, G. *Phys. Rev. B* **2013**, *87*, 184115.

[68] Ferré, G.; Maillet, J. B.; Stoltz, G. *J. Chem. Phys.* **2015**, *143*, 104114.

[69] Behler, J.; Parrinello, M. *Phys. Rev. Lett.* **2007**, *98*, 1.

[70] Bartók, A. P.; Payne, M. C.; Kondor, R.; Csányi, G. *Phys. Rev. Lett.* **2010**, *104*, 1.

[71] Behler, J. *J. Chem. Phys.* **2016**, *145*, 170901.

[72] Rupp, M.; Tkatchenko, A.; Müller, K.-R.; von Lilienfeld, O. A. *Phys. Rev. Lett.* **2012**, *108*, 058301.

[73] Zhu, L.; Amsler, M.; Fuhrer, T.; Schaefer, B.; Faraji, S.; Rostami, S.; Ghasemi, S. A.; Sadeghi, A.; Grauzinyte, M.; Wolverton, C.; Goedecker, S. *J. Chem. Phys.* **2016**, *144*, 034203.

[74] Ramírez, M.; Rogan, J.; Valdivia, J. A.; Varas, A.; Kiwi, M. *Zeitschrift fur Phys. Chemie* **2016**, *230*, 977.

[75] Steinhardt, P. J.; Nelson, D. R.; Ronchetti, M. *Phys. Rev. B* **1983**, *28*, 784.

[76] Kondor, R. *CoRR* **2007**, *abs/cs/0701127*.

[77] Gelbrich, T.; Threlfall, T. L.; Hursthouse, M. B. *CrystEngComm* **2012**, *14*, 5454.

[78] Behler, J. *J. Chem. Phys.* **2011**, *134*, 074106.

[79] Pietrucci, F.; Andreoni, W. *Phys. Rev. Lett.* **2011**, *107*, 1.

[80] Barker, J.; Bulin, J.; Hamaekers, J.; Mathias, S. *arXiv Prepr. arXiv1611.05126* **2016**,

[81] Valle, M.; Oganov, A. R. *Acta Crystallogr. Sect. A Found. Crystallogr.* **2010**, *66*, 507.

[82] Saberi Fathi, S. M.; White, D. T.; Tuszynski, J. A. *Proteins Struct. Funct. Bioinforma.* **2014**, *82*, 2756.

[83] Swendsen, R. H.; Wang, J.-S. *Phys. Rev. Lett.* **1986**, *57*, 2607–2609.

[84] Sugita, Y.; Okamoto, Y. *Chem. Phys. Lett.* **1999**, *314*, 141–151.

[85] Earl, D. J.; Deem, M. W. *Phys. Chem. Chem. Phys.* **2005**, *7*, 3910.

[86] Fukunishi, H.; Watanabe, O.; Takada, S. *J. Chem. Phys.* **2002**, *116*, 9058.

[87] Mandelshtam, V. A.; Frantsuzov, P. A.; Calvo, F. *J. Phys. Chem. A* **2006**, *110*, 5326–5332.

[88] Bogdan, T. V.; Wales, D. J.; Calvo, F. *J. Chem. Phys.* **2006**, *124*, 44102.

[89] Wales, D. J. *Chem. Phys. Lett.* **2013**, *584*, 1–9.

[90] Frenkel, D.; Smit, B. Understanding Molecular Simulation. 2002.

[91] Betancourt, M. *arXiv Prepr. arXiv1701.02434* **2017**,

[92] Duane, S.; Kennedy, A.; Pendleton, B. J.; Roweth, D. *Phys. Lett. B* **1987**, *195*, 216–222.

[93] Hoffman, M. D.; Gelman, A. *J. Mach. Learn. Res.* **2014**, *15*, 1593–1623.

[94] Martiniani, S.; Stevenson, J. D.; Wales, D. J.; Frenkel, D. *Phys. Rev. X* **2014**, *4*, 31034.

[95] Mezey, P. G. *Potential Energy Hypersurfaces*; Elsevier: Amsterdam, 1987.

[96] Wales, D. J. *Mol. Phys.* **1993**, *78*, 151.

[97] Wang, J.-s.; Swendsen, R. H. *J. Stat. Phys.* **2002**, *106*, 245.

[98] Kim, J.; Straub, J. E.; Keyes, T. *Phys. Rev. Lett.* **2006**, *97*, 1–4.

[99] Kim, J.; Keyes, T.; Straub, J. E. *J. Chem. Phys.* **2009**, *130*, 1–10.

[100] Kim, J.; Straub, J. E.; Keyes, T. *J. Phys. Chem. B* **2012**, *116*, 8646–8653.

[101] Higson, E.; Handley, W.; Hobson, M.; Lasenby, A. *Prepr.* **2017**,

[102] Handley, W. J.; Hobson, M. P.; Lasenby, A. N. *Mon. Not. R. Astron. Soc. Lett.* **2015**, *450*, L61–L65.

[103] Keeton, C. R. *Mon. Not. R. Astron. Soc.* **2011**, *414*, 1418–1426.

[104] Feroz, F.; Skilling, J. Exploring multi-modal distributions with nested sampling. AIP Conf. Proc. 2013; pp 106–113.

[105] Betancourt, M. *AIP Conf. Proc.* **2010**, *1305*, 165–172.

[106] Feroz, F.; Hobson, M. P.; Bridges, M. *Mon. Not. R. Astron. Soc.* **2009**, *398*, 1601–1614.

[107] Brewer, B. J.; Pártay, L. B.; Csányi, G. *Stat. Comput.* **2011**, *21*, 649–656.

[108] Feroz, F.; Hobson, M. P.; Cameron, E.; Pettitt, A. N. *arXiv Prepr. arXiv1306.2144* **2013**, 28.

[109] Chopin, N.; Robert, C. P. *Biometrika* **2010**, *97*, 741–755.

[110] Pártay, L. B.; Bartók, A. P.; Csányi, G. *J. Phys. Chem. B* **2010**, *114*, 10502–10512.

[111] Kaufmann, K.; Baumeister, W. *J. Phys. B At. Mol. Opt. Phys.* **1989**, *22*, 1.

[112] Baddour, N.; Chouinard, U. *J. Opt. Soc. Am. A* **2015**, *32*, 611.

[113] Freidman, J. H.; Bentley, J. L.; Finkel, R. A. *ACM Trans. Math. Softw.* **1977**, *3*, 209.

[114] Stoddard, S. D.; Ford, J. *Phys. Rev. A* **1973**, *8*, 1504.

[115] Kob, W.; Andersen, H. C. *Phys. Rev. E* **1995**, *51*, 4626.

[116] PELE: Python Energy Landscape Explorer, https://github.com/pele-python/pele. https://github.com/pele-python/pele, Accessed: 2017-08-22.

[117] Trygubenko, S. A.; Wales, D. J. *J. Chem. Phys.* **2004**, *121*, 6689.

[118] León, C. A.; Massé, J.-C.; Rivest, L.-P. *J. Multivar. Anal.* **2006**, *97*, 412.

[119] Rosato, V.; Guillope, M.; Legrand, B. *Phil. Mag. A* **1989**, *59*, 321.

[120] Miller, M. A.; Amon, L. M.; Reinhardt, W. P. *Chemical Physics Letters* **2000**, *331*, 278 – 284.

[121] Swendsen, R. H. *Phys. Procedia* **2011**, *15*, 81–86.

[122] Neal, R. M. *Ann. Statist.* **2003**, *31*, 705–767.

[123] Ballard, A. J.; Martiniani, S.; Stevenson, J. D.; Somani, S.; Wales, D. J. *Wiley Interdiscip. Rev. Comput. Mol. Sci.* **2015**, *5*, 273–289.

[124] Wales, D. J. *Phys. Rev. E* **2017**, *95*, 030105(R).

Publications

Some of the work presented in this thesis has been published in the following paper:

Griffiths M.; Niblett, S. P.; Wales, D. J. *J. Chem. Theor. Comput.* **2017**, *13*, 4914.