DEPARTMENT OF PHYSICS

University of Strathclyde Science

PH450 Report 2021-22

Performance Enhancements of the LAMMPS CG-DNA Package and Subsequent DNA Analysis of CTG Sequence Hairpin Effects

Submitted in partial fulfilment for the degree of MPhys Physics with Advanced Research

Lewis MacLeod Russell
Registration No.: 201817988

SUPA Department of Physics, University of Strathclyde, Glasgow G4 0NG, United Kingdom

Dr Oliver Henrich (Primary Supervisor)
(Dated: 4th April 2022)

Abstract

The content of this Report centres around improvements to the use and conversion of co-ordinate systems within the in-built DNA simulation package, "CG-DNA", for LAMMPS [1]. LAMMPS is an open-source software package focusing on providing a computationally diverse and efficient molecular dynamics (MD) toolkit. These are primarily designed towards academic and scientific researchers, and has in-built packages designed for a range of discipline areas, including: coarse-gained and mesoscopic systems, solid-state materials, and biomolecular and polymer systems. The "CG-DNA" package is one such example, and is primarily maintained by Dr. Henrich.

Upon release of the "CG-DNA" package, Dr. Henrich later discovered that areas within the code where disproportionately consuming large percentages of the total simulation times [2]. Motivated by the interest to rebalance these sections of code, thus reducing simulations time, the main focus of this Project and Report was determined.

After successful time reductions where achieved, the new source code was utilised for the study of specific DNA behaviours; the details of which will be clarified in the latter Sections of this Report.

Acknowledgements

Significant recognition is due to Dr. Oliver Henrich who has been a dedicated supervisor and mentor throughout the duration of this project. Whilst the work produced has been of my own doing, Dr. Henrich has provided exceptional guidance on all the useful insights that are typical of such computational based projects. From all the random, seemingly insignificant, command line arguments that suddenly rectify terminal errors. To providing concise physical and conceptual explanations of the coarse-gained ox-DNA model; allowing me to work in my own time at understanding this complex system into greater depth. Without this, my job would have been far more difficult and progress would have been on the slower side!

In addition, the online documentation for LAMMPS [3] is well-written to say the least. Due to its global and large-scale open-source natural, this documentation is crucial - without it, continued software development of the package would have a far higher barrier to entry, and less computational physicists would be inclined to embrace LAMMPS and its features. The documentation is thorough in its description of how to use the package for scientists interested in its features, as well as giving plenty of information to developers as to how the package functions at a software level. Plenty of pseudo-code is given [4], and an overview of the class inheritance structure of LAMMPS is also provided [5]. There is more than adequate detail, well-written here, that provided me a strong conceptual basis of LAMMPS's source code that then enabled me to analyse the source code directly and implement the physical improvements that where necessary.

Contents

List of Figures

I. Introduction

The emergence of coarse-grained modelling offers an intermediate balance between computational accuracy, reasonable simulation times, and modestly large system size and time scales [6]. As such, coarse-grained modelling has gained popularity within the molecular simulation study of biological systems [7, 8], where the complexity of such systems often rules out atomic-resolution simulation due to its computationally expensive price-tag.

Perhaps one of the greatest polymers of interest within this field is that of DNA - the genetic library of cellular information within nature, responsible for housing the genetic code by which all life is governed. However, this is a prime example of a molecule so large that it must be simplified for computational purposes if any realistic time and size scale is to be achieved.

A range of models exist for this purpose [6–9], however the *"oxDNA"* variant will be of focus within this literature [2, 9–12]. Specifically, this Project was aimed at analysing the source code of the CG-DNA package within the open-source molecular dynamics package *"LAMMPS"*, and ultimately intended to develop and implement computational efficiency improvements to the package, thus reducing simulation times for the end users. The CG-DNA add-on package for LAMMPS is one of only two currently available implementations of the oxDNA model, and hence any reduction to simulation time provides substantial benefit directly to the main user-base of the package.

The improvement that was developed significantly reduced the handling times of Quaternions within the CG-DNA Package for LAMMPS (herein, this will simply be referred to as "the Software"). Quaternions form a 4-Dimensional co-ordinate system that describes the orientation of an object, and is an extension of more typical 2-Dimensional complex numbers that those within STEM[1] are used to. However, force routines that can handle 4-component quaternion momenta and torques directly are not provided within LAMMPS [2], and so their conversion into sets of typical 3-Dimensional co-ordinates are required. Reducing these calculations was what directly equated to improved calculation times for the Software.

Upon developing the improvements, the new source code was used to investigate a DNA sequence with a length of 64-Base-Pairs. DNA typically forms a double-stranded helical structure, however topological defects have been discovered that show isolated hairpin-like off-shoots from the usual DNA sequence [13, 14]. This is where one of the two stands diverges from the sequence, doubles back on itself, and reconnects back to form the usual double-helix formation. Experimental results have proven that such hairpins can cause "slippage events" where a sequence of DNA base-pairs relocate from the hairpin into the main double-helical section of the strand. This has been linked to at least 17 degenerative genetic disorders within humans [13]. Using the Software, this effect was prevalently observed, confirming the validity of the package to replicate these findings [13, 14].

This Report is primarily divided into three sections: Theory, Methodology, and Results. Within the Theory section, a mathematical description of quaternions is given, followed by an overview of various DNA modelling packages, a breakdown of how the oxDNA model functions and what purpose it serves, and why this model has been developed for LAMMPS under the CG-DNA Package. The Methodology is arguably the largest portion of this Report, and explains how the original source code functioned, where its shortcomings lay, and how this was improved and rectified. Configuration of the Software as an end-user is also described as part of the DNA Hairpin investigative description. Results are provided towards the end of the Report, with the source code improvements being highlighted in the form of a side-by-side benchmark comparison, and the findings of the "slippage events" occurring in a simulated DNA Hairpin are presented.

Lastly, the Discussion and Conclusion section will highlight the outcome of this Project, determine its success, and consider its ramifications.

[1] STEM - "Science, Technology, Engineering, and Mathematics".

II. Theory

This Theory section intends to deliver background insight regarding the fundamentals of the Project. By the end of this section, it is hoped that the reader has a conceptual knowledge of both quaternions and the oxDNA model. This should help in understanding the Methodology section, without having to introduce too many new concepts at once and disrupting the flow of the text.

A. Quaternions and their Properties

Despite being most famously known for his creation of the "Hamiltonian", a fundamental energy description used within analytical mechanics, Sir William Rowan Hamilton was also responsible for the invention of "quaternions" [15].

Quaternions are not typically introduced within undergraduate courses, although their importance can not be understated. Their ability to describe both 3D orientation, without the drawbacks of other methods, has resulted in their dominant use within computing science. It is highly likely that the device you are using to read this Report is relying on quaternions somewhere within its operating code, whether that be to track phone orientation and movement, or to apply rotations to the graphics displayed on your screen.

Before exploring quaternions, lets first consider more intuitive co-ordinate systems. From a linear algebraic perspective, take the following 3x3 matrix:

$$R = \begin{bmatrix} x_i & x_j & x_k \\ y_i & y_j & y_k \\ z_i & z_j & z_k \end{bmatrix} \tag{1}$$

There are three orthonormal unit vectors, each with an x,y,z position in space. Since this is in 3-Dimensions, it is fairly easy to visualise how the rotation takes effect. The object can also be thought of as being in some rotation around three axes, each with an associated angle.

One major downfall of using Euler Angles and Rotation Matrices upon this Cartesian[2] system, is

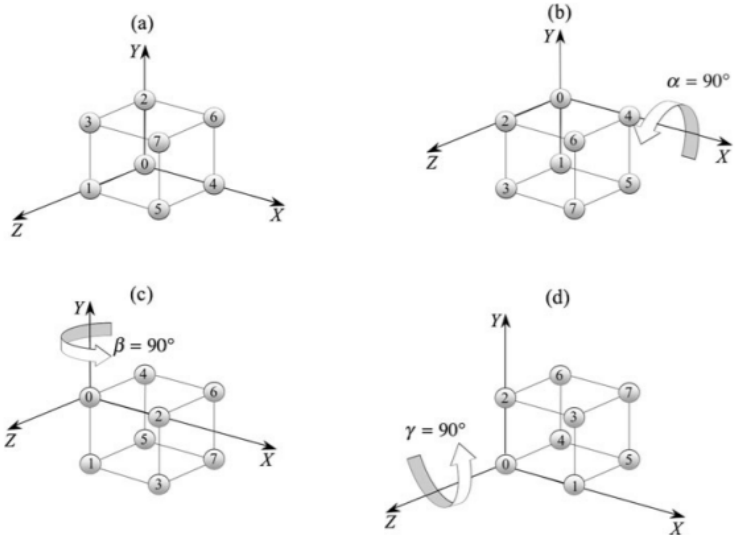


Figure 1: An Example of Gimbal Lock [15]. Despite performing three separate rotations around three separate axes, the net effect is no different than simply rotating around the y-axis here.

[2] Cartesian Co-ordinates - describes positions using x,y,z co-ordinates.

that it is subject to a condition know as "gimbal lock".

Figure 1 illustrates one such gimbal-locked scenario. Here, a 90° rotation is applied to each of the x,y,z-axes, respectively. The axes passing through the cube's vertices of, 0 and 4, and 0 and 1, have been rotated and are in updated positions; however, the axis passing through the 0 and 2 points has been ignored. The net result ('d' in Figure 1) of these three rotations around the three separate axes is that it is no different to simply rotating the initial state ('a' in Figure 1) 90° around the y-axis! Essentially, a degree of freedom has been lost due to having two of the axes of rotation becoming locked.

In another example, this can be seen from the rotation matrices in Equation 2 below. Taking $\beta = 90° = \pi/2$, the sine and cosine terms reduce to 1 and 0 respectively, and the gimbal lock can be mathematically represented.

$$\begin{aligned}
R &= R_{x,\alpha}R_{y,\beta}R_{z,\gamma} \\
&= \begin{pmatrix} 1 & 0 & 0 \\ 0 & cos(\alpha) & -sin(\alpha) \\ 0 & sin(\alpha) & cos(\alpha) \end{pmatrix} \begin{pmatrix} cos(\beta) & 0 & sin(\beta) \\ 0 & 1 & 0 \\ -sin(\beta) & 0 & cos(\beta) \end{pmatrix} \begin{pmatrix} cos(\gamma) & -sin(\gamma) & 0 \\ sin(\gamma) & cos(\gamma) & 0 \\ 0 & 0 & 1 \end{pmatrix} \\
&= \begin{pmatrix} 1 & 0 & 0 \\ 0 & cos(\alpha) & -sin(\alpha) \\ 0 & sin(\alpha) & cos(\alpha) \end{pmatrix} \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{pmatrix} \begin{pmatrix} cos(\gamma) & -sin(\gamma) & 0 \\ sin(\gamma) & cos(\gamma) & 0 \\ 0 & 0 & 1 \end{pmatrix} \\
&= \begin{pmatrix} 0 & 0 & 1 \\ sin(\alpha) & cos(\alpha) & 0 \\ -cos(\alpha) & sin(\alpha) & 0 \end{pmatrix} \begin{pmatrix} cos(\gamma) & -sin(\gamma) & 0 \\ sin(\gamma) & cos(\gamma) & 0 \\ 0 & 0 & 1 \end{pmatrix} \\
&= \begin{pmatrix} 0 & 0 & 1 \\ sin(\alpha)cos(\gamma) + cos(\alpha)sin(\gamma) & -sin(\alpha)sin(\gamma) + cos(\alpha)cos(\gamma) & 0 \\ -cos(\alpha)cos(\gamma) + sin(\alpha)sin(\gamma) & cos(\alpha)sin(\gamma) + sin(\alpha)cos(\gamma) & 0 \end{pmatrix}
\end{aligned} \tag{2}$$

And using the angle addition and subtraction theorem to simplify:

$$R = \begin{pmatrix} 0 & 0 & 1 \\ sin(\alpha+\gamma) & cos(\alpha+\gamma) & 0 \\ -cos(\alpha+\gamma) & sin(\alpha+\gamma) & 0 \end{pmatrix} \tag{3}$$

Here in Equation 3, the first row and last column of the rotation matrix have no dependence on α and γ, meaning that regardless of changing these values, the rotation is fixed; in this case, to the Z-axis.

In contrast, this characteristic is not possible within the quaternion system. The quaternion is defined as follows:

$$\begin{aligned}
q = s + a\hat{\boldsymbol{i}} + b\hat{\boldsymbol{j}} + c\hat{\boldsymbol{k}} \quad , \quad where: \\
\hat{\boldsymbol{i}}^2 = \hat{\boldsymbol{j}}^2 = \hat{\boldsymbol{k}}^2 = \hat{\boldsymbol{i}}\hat{\boldsymbol{j}}\hat{\boldsymbol{k}} = -1 \\
\hat{\boldsymbol{i}}\hat{\boldsymbol{j}} = \hat{\boldsymbol{k}}, \quad \hat{\boldsymbol{j}}\hat{\boldsymbol{k}} = \hat{\boldsymbol{i}}, \quad \hat{\boldsymbol{k}}\hat{\boldsymbol{i}} = \hat{\boldsymbol{j}} \\
\hat{\boldsymbol{j}}\hat{\boldsymbol{i}} = -\hat{\boldsymbol{k}}, \quad \hat{\boldsymbol{k}}\hat{\boldsymbol{j}} = -\hat{\boldsymbol{i}}, \quad \hat{\boldsymbol{i}}\hat{\boldsymbol{k}} = -\hat{\boldsymbol{j}}
\end{aligned} \tag{4}$$

s, a, b, c are all real numbers, and $\hat{\boldsymbol{i}}, \hat{\boldsymbol{j}}, \hat{\boldsymbol{k}}$ are basic quaternions. These *basic quaternions* are analogues to defining a typical vector in terms of its unit Cartesian vectors, although are in this case imaginary.

If a 3D co-ordinate is rotated using quaternions, it's constructed as a vector as expected (say "p"). However, to perform the rotation, not just a single quaternion (take "q") is used, but rather a sandwich of the quaternion acting on the vector, followed by its inverse on the right.

And, much like complex numbers, the real part is represented by a cosine term, and the imaginary

part is represented by a sine term. For 3D rotations, this can be represented as:

$$\mathbf{q} = \left[cos\tfrac{\theta}{2},\ sin\tfrac{\theta}{2}\mathbf{v}\right] = \left[cos\tfrac{\theta}{2},\ sin\tfrac{\theta}{2}(a\hat{\boldsymbol{i}} + b\hat{\boldsymbol{j}} + c\hat{\boldsymbol{k}})\right] \tag{5}$$

And hence, for the rotation method as described above, where p' represents the updated position:

$$\begin{aligned}\mathbf{p}' &= \mathbf{qpq}^{-1} \\ &= \left(cos\tfrac{\theta}{2} + sin\tfrac{\theta}{2}(a\hat{\boldsymbol{i}} + b\hat{\boldsymbol{j}} + c\hat{\boldsymbol{k}})\right)\left(p_x\hat{\boldsymbol{i}} + p_y\hat{\boldsymbol{j}} + p_z\hat{\boldsymbol{k}}\right)\left(cos\tfrac{\theta}{2} - sin\tfrac{\theta}{2}(a\hat{\boldsymbol{i}} + b\hat{\boldsymbol{j}} + c\hat{\boldsymbol{k}})\right)\end{aligned} \tag{6}$$

It is worth noting that the vector p itself is also a quaternion, just with a real part equal to zero. The vector is embedded within the imaginary part of the quaternion.

In the latter sections of this Report, quaternion usage within the Software will be explored; namely, Section II D 2 briefly explains how quaternions are used and updated within each timestep; and the conversion of the quaternions into local-reference-frame x,y,z co-ordinates (along with their handling within the Software) is covered within the Methodology in Section III A.

B. A Brief Overview of DNA Modelling Packages

The isolated study of DNA is an extremely in-depth area of research, even without accounting for its interactions with proteins and other in vivo[3] substances. Some important properties will occur in the time scale of femtoseconds, whilst others require years to evolve. And on a length scale, some effects are only apparent on a quantum mechanical level requiring sub-Angstrom resolution; whilst other effects are only present within millimetre-long DNA segments. Naturally, with such a range of possible behaviours to study, many different DNA models have been created and explored. Each of these models have their own strengths and weaknesses, and are best used within their designed time and length scale ranges. This diversity of DNA modelling scales is shown below in Figure 2.

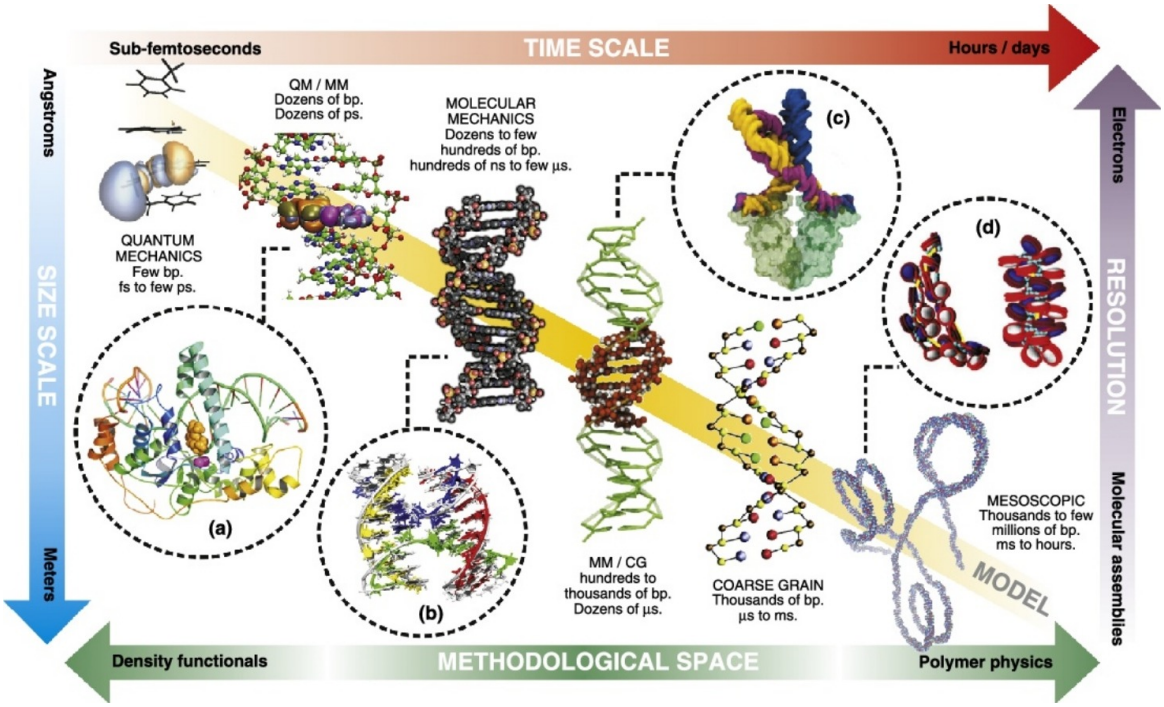


Figure 2: An Illustration of the Intrinsic Multiscale Nature of DNA [6].

[3] In Vivo - biological studies within their natural environment.

In ascending order of resolution, DNA models can thought of as being: electronic, atomistic, coarse-grained, or mesoscopic.

The "electronic" regime takes a complete first-principles (fundamental) approach based on quantum physics; thus, yields extremely high-quality results due to the absence of any simplification or parameterisation within the model. However, with quantum mechanical methods comes excessively high-cost calculations. Thus, these models, whilst providing great detail, can only simulate DNA a few base-pairs in length (Angstrom/nanometre length scale) on a time scale lasting just a few femtoseconds to picoseconds.

On the opposite extreme of this, "mesoscopic" studies are found. DNA within human cells, if stretched out end-to-end, would reach an incredible one-metre in length; yet, is compressed within the cell nucleus to a diameter of only around six-microns [6]. These mesoscopic models encompass the only DNA model type capable of achieving length scales relevant to chromatin[4] studies, and are generally developed from behavioural knowledge of chromatin.

As this Project solely focuses on the coarse-gained oxDNA model, these extreme scales are not of great relevance to this Report. However, they are worth mentioning. This provides a reference point as to where coarse-gained DNA models specialise within the overall DNA modelling picture.

Coarse-grained (CG) DNA models are typically particle based methods, that is to say, the nucleotide[5] is represented by a number of effective beads; many atoms are represented by one singular bead. All of these models where developed from either a "bottom-up" or a "top-down" approach. The former simplifies atomistic force fields by disregarding non-dominant degrees of freedom. The latter seeks reference to experimental observations, parameterising the model in a complimentary fashion; thus, reflecting the experimental findings [2].

As can be seen from Figure 3, the oxDNA model has one rigid body per nucleotide (albeit, with additional interaction sites), as opposed to other models shown that have a handful more.

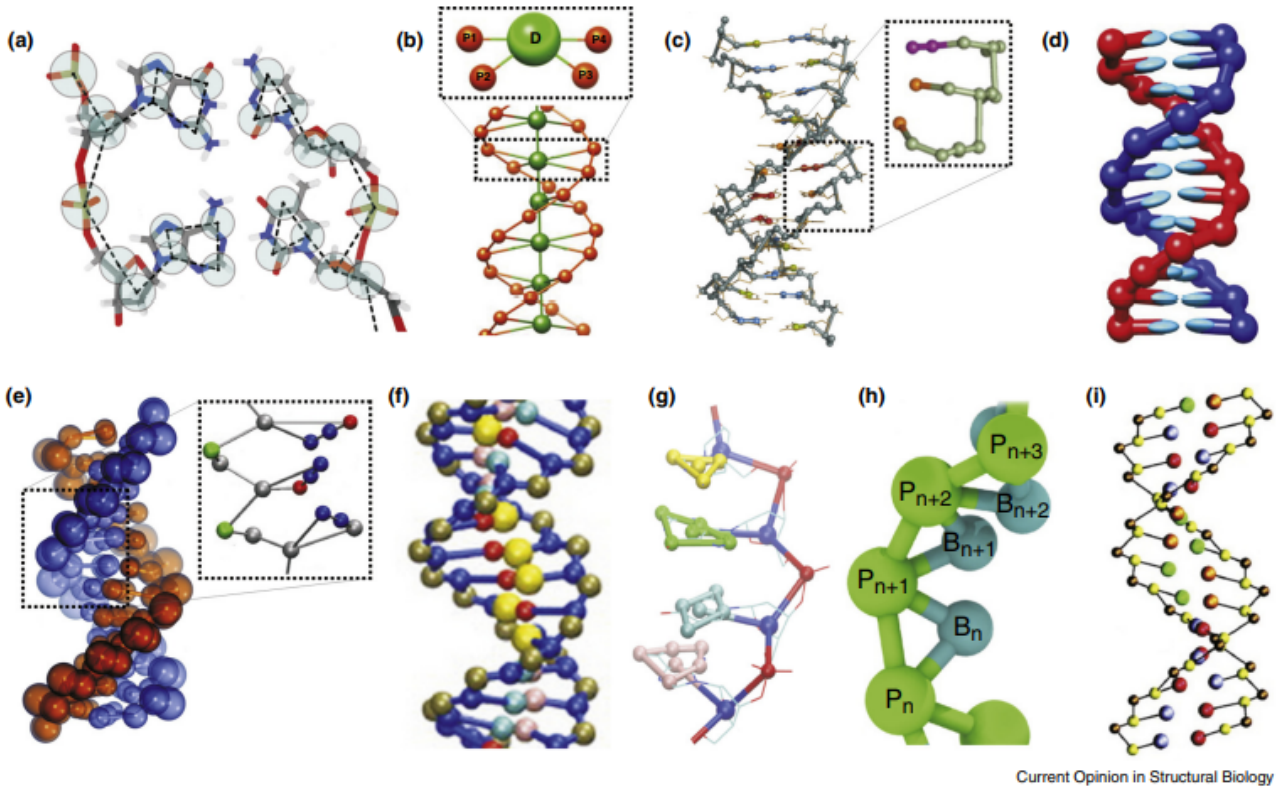


Figure 3: Various CG-DNA Models, with the oxDNA model shown at **(d)** [6].

[4] Chromatin is a substance within a chromosome, a large DNA molecule, consisting of DNA and protein.

[5] Nucleotides form the composition of nucleic acids, and together comprise DNA strands.

The advantage of this is that it can achieve around an order of ten more base-pairs (10x longer DNA strand) than the other CG-DNA models. However, some features are lost as a trade-off. For example, other models shown have multiple beads distinguishing between the sugar-phosphate and base molecules, offering the capability of a flexible connection between the sugar-phosphate backbone and the nucleotide base. This also allows for easier modelling of undesired foreign molecules bonding to the nucleotide, which can represent damaging DNA effects.

These models do however share commonality in their energy descriptions. Classical terms are the preferred option here due to their comparably cheaper computational cost over QM energy descriptions, and are more appropriate given the collective bundling of many implicit atoms into each bead. Harmonic potentials are frequently used for bonded interactions, whilst Lennard-Jones or Coulomb potentials are prime candidates for non-bonded interactions [6].

C. The Functionality and Purpose behind the 'oxDNA' Model

Contributing towards the main focus and partial fulfilment of his PhD [10], Thomas Ouldridge set out to to create a "novel coarse-grained model for DNA" that evolved into the model known today as *oxDNA*. His philosophy behind the model was to:

"embed the thermodynamics of transitions involving ssDNA and dsDNA [single-stranded and double-stranded DNA] into a 3-dimensional, dynamical, coarse-grained representation that provides a reasonable description of the structural and mechanical features of the molecule. This ambition naturally coincides with a top-down approach." [10]

His primary purpose for this model was to develop an accurate model that represented the net effects of intricate chemical details, without the need to explicitly calculate or consider these heavier computations directly. This is his reasoning for committing to a *top-down* modelling approach. Perhaps most conceptually important, he wanted modelling properties to arise only from physically motivated interactions; in other words, the model should not contain any explicitly defined length or loop size dependencies, which would force the model into pre-defined dynamics. This means that any effects arising in larger length simulations are purely forming from a net effect of contributions from each nucleotide, rather than having any length-induced bias.

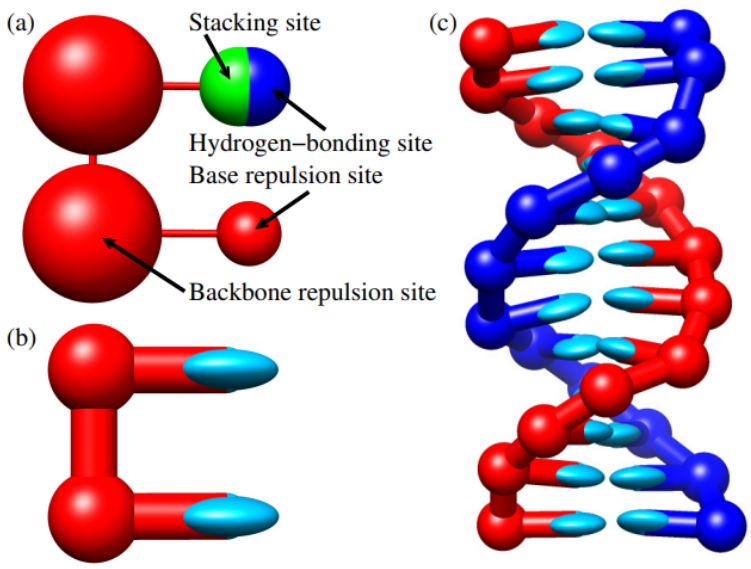


Figure 4: Model Interaction Sites within the oxDNA Model [10].

Figure 4 indicates Ouldridge's definition of interaction sites within oxDNA. Base-stacking[6] and hydrogen-bonding are separated from the backbone site by 6.3Å and 6.8Å, respectively, and thus form rigid nucleotides. The bases themselves also have a defined orientation that alters interaction effects. These interaction potentials have been configured in such a way as to optimise the properties of DNA and significantly reduce the likelihood of physically irrelevant formations. In the original model, the total potential is characterised into six contributing factors:

- The "FENE Bond", which connects backbones.
- Stacking and Hydrogen-Bonded Potentials
- Cross-Stacking and Coaxial-Stacking Potentials
- Excluded Volume Potentials

These potentials are visualised in Figure 5. It is worth noting that the "Excluded Volume" potential comes in the form of a Lennard-Jones (LJ) potential describing intermolecular pair interactions. These LJ interactions are a common choice for describing the soft repulsive and attractive interactions of electronically neutral particles within molecular dynamics. Since this potential converges to zero at larger radial distances between pairwise selections, a cut-off distance is sensibly introduced to avoid unnecessary computation of the negligible interaction potentials. This "excluded volume" (LJ) potential is also negated from Figure 5 - as it is a radially dependant pairwise interaction between any of the beads, including this in the figure would only lead to an over-cluttered and misleading graphic of the model.

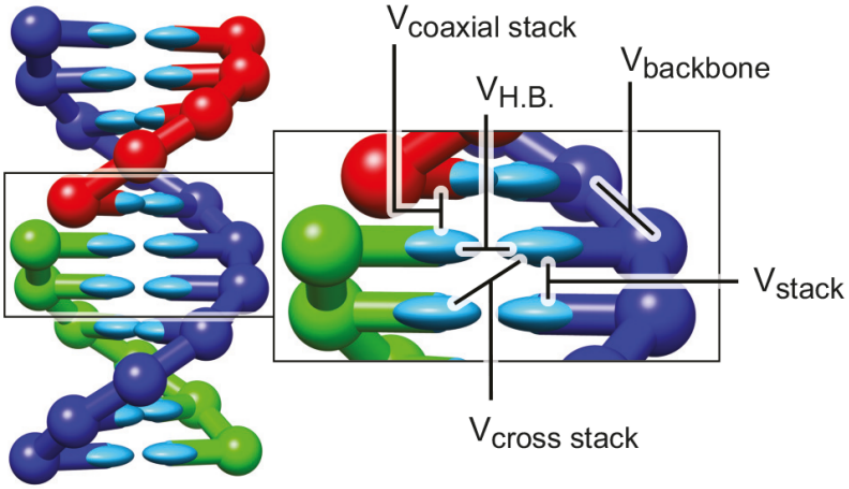


Figure 5: The Main Potentials Involved within the oxDNA Model [2].

In addition to the potentials and nucleotide geometry described above, the updated oxDNA2 version introduced some new features to this selection. Regarding interactions, a Debye-Hückel potential was added [2]. This additional and implicit electrostatic potential is designed to allow for DNA modelling within solutions representing different salt concentrations [11, 17].

The nucleotide interactions sites where also updated in oxDNA2 [2, 17], shown in Figure 6. In the original version (oxDNA), all interaction sites are co-linear. However, oxDNA2 introduces an angular offset between the backbone and stacking/hydrogen-bond positions. The difference between these two configurations can be seen from **(a)** in Figure 6. Of course, this carries with it a slight computational overhead, although does allow for the model to now represent major and minor grooving. This is noticeable from the extra "twist" visible from **(b)** in Figure 6.

[6] Bases in DNA are one of four types, forming complimentary pairs of either, adenine (A) and thymine (T), or cytosine (C) and guanine (G) [16].

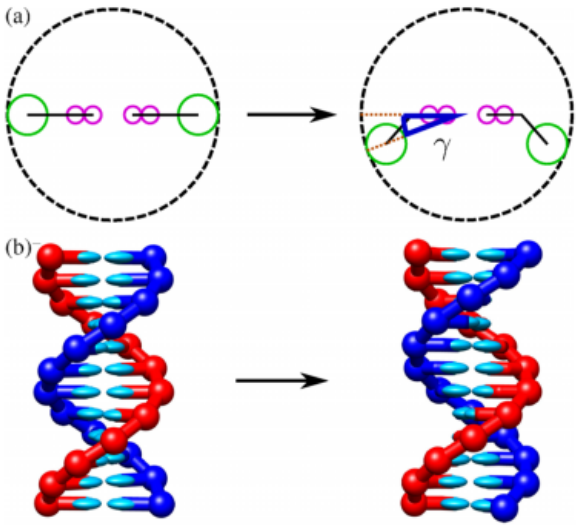


Figure 6: The Positional Differences Between Interaction Sites in oxDNA (left) and oxDNA2 (right) [2]. Note the additional offset between the backbone and stacking/hydrogen-bond positions in oxDNA2, as opposed to the co-linearity in oxDNA.

D. An Insight into LAMMPS and the CG-DNA Package

Within this Section, a brief background of the history of LAMMPS and its features is given. An overview of the general coding structure is also provided, which is then lightly expanded upon in the context of the CG-DNA package.

1. The Purpose and History of LAMMPS

LAMMPS can be summarised as a simulation package catering for "particle-based materials modelling at an atomic, meso, and continuum scale." [18].

As well as providing an in-built platform for easy implementation of particle interaction models, models can also be scaled relatively easily across many CPU cores[7] without any need for any in-depth computational consideration or development from the user. This is in-built ability of LAMMPS, and can be defined through a single command line within the package's command line function call [19]. Regardless of running on a single-core laptop CPU, or a massive supercomputer with thousands of CPU cores, LAMMPS is still adequately capable of running efficiently on the CPU cores it is pre-allocated. Later observations within the Results section shall highlight this.

Since its first public release around two decades ago, LAMMPS has grown dramatically in both popularity and size, going from fifty-thousand lines of code in 2004 to an impressive one-million lines today [18]. In cases where a feature is not available within the package, additional code can be implemented with relative ease; at least in comparison to creating new standalone software. LAMMPS is structured in a class-inherited fashion [5], and so new features can often be defined that grow off of pre-existing and similar attributes.

Altogether, this has resulted in LAMMPS gaining increasingly wide-spread adoption and popularity throughout the years.

[7] A CPU core is the main processing unit within a computer. Here, each program (software) needs to be run on a single core sequentially. However, to improve calculation times in many circumstances, the task can be spilt amongst multiple cores to share the workload. Once the multi-core workload has taken place, the information must be feed to a single process and interpreted correctly, then appropriately summarised to an output.

2. The Process of Verlet Integration within LAMMPS

The Verlet algorithm is an interesting numerical solver used to integrate Newtonian equations of motion, and provides reliable results due to its high numerical stability. It is also time reversible, meaning that if it is run in reverse from its end point in time, the system will return to its initial conditions - total energy is conserved.

One of the most common applications for Verlet Integration comes from the consideration of acceleration in the form of a second-order ODE of position:

$$\ddot{x} = \frac{d^2x}{dt^2} \tag{7}$$

And when $\ddot{x}$ is pre-factored with m to represent mass, this equation then provides an equation for force:

$$\boldsymbol{F} = m\ddot{\boldsymbol{x}} \tag{8}$$

However, the equations for force calculations within the Verlet algorithm will typically use a form of:

$$\boldsymbol{F} = -\frac{dV(\boldsymbol{r})}{d\boldsymbol{r}} \tag{9}$$

This removes dependence on acceleration, and instead relies on position within the context of an energy potential. This is an important consideration for this algorithm, as it means acceleration only has a positional dependency on the specified energy potential.

From here, the "Velocity Verlet" flavour of the algorithm is explained, as this is the version of Verlet Integration that LAMMPS uses. This can be summarise as follows:

1. Velocities are calculated at a half timestep as:

$$v_{i+\frac{1}{2}} = v_i + \frac{\Delta t}{2} f(x_i) \tag{10}$$

2. Positions are calculated at a full timestep as:

$$x_{i+1} = x_i + \Delta t v_{i+\frac{1}{2}} \tag{11}$$

3. Forces at the full timestep are calculated using the new x_{i+1} positions. The exact equation for force will vary depending on the energy potential.

4. Velocities are updated for the full timestep as:

$$v_{i+1} = v_{i+\frac{1}{2}} + \frac{\Delta t}{2} f(x_{i+1}) \tag{12}$$

As mentioned in previous sections, the CG-DNA package also contains orientational dependence as well as positional.

Where the above steps have been described using position, force, and velocity, the same concept is applicable to quaternions, torque, and angular momentum. These are relatable in the sense: quaternions take the place of positions, torques take the place of forces, and angular momenta take the place of velocities.

3. Langevin Dynamics within LAMMPS

Langevin Dynamics are an essential consideration for almost all molecular simulators. DNA modelling, as is the case with most MD situations, does not occur within a vacuum. Instead, DNA is modelled within a solution (or solvent), and thus subjects the DNA strand to both friction and collisions surrounding its neighbouring solution molecules.

It is not necessary to consider these surrounding molecules directly and on an individual basis. Instead, the physical effects of the solution on the DNA strand can be implicitly defined. Friction can be said to be velocity-dependent, and a random force can be applied to represent particle collisions. It is not known in practice how large each individual collision will be, however, the collective effects can be well-defined. The random (stochastic) force can be modelled by drawing random numbers for its input, and is typically drawn from either a Gaussian or uniform distribution. Both serve the same purpose, though their differences lie in speed and accuracy: the uniform distribution is slightly less realistic, although is quicker to solve for. In the interest of speed, LAMMPS takes the random number from a uniform distribution [20].

The Langevin Equation is an extension of Newton's Second Law, and in its general format it takes the form of:

$$\begin{aligned} m\ddot{\boldsymbol{x}} &= \boldsymbol{F}_c + \boldsymbol{F}_f + \boldsymbol{F}_r \\ &= -\frac{dV(\boldsymbol{r})}{d\boldsymbol{r}} - \gamma m\boldsymbol{v} + \sqrt{2m\gamma k_b T}\eta(t) \end{aligned} \tag{13}$$

Where: $\boldsymbol{F}_c$ is a conservative force and is potential-dependent; $\boldsymbol{F}_f$ is the velocity-dependent frictional force, with the Langevin friction coefficient γ being related to the viscosity of the solvent or solution; and $\boldsymbol{F}_r$ is the random force contribution which is temperature-dependent. η represents the stochastic force contribution, and k_bT is the Boltzmann constant multiplied by system temperature. It should be stated that the mean average value of the stochastic term $\eta(t)$ should be zero. The value may be significant or negligible at each timestep, although across a larger enough sample of time it should average as zero across all values. This accurately describes the random collisions from the solvent, without introducing an unwanted bias.

LAMMPS takes a slightly simplified form of Equation 13, where:

$$\boldsymbol{F}_f = -\frac{m}{damp}v \quad and \quad \boldsymbol{F}_r \propto \sqrt{\frac{k_bTm}{dt \cdot damp}} \tag{14}$$

The overall concept here is still the same, only the arrangement of terms is just in a more practical form for the LAMMPS input structure. The Langevin friction coefficient is represented by "$\frac{1}{damp}$", and is given in this division rather that multiplication format since it is more suited to the reduced units that LAMMPS users typically opt for. Reduced units re-define values in a more appropriate format over their SI[8] units counterparts, as this negates large orders of magnitude within values. When reduced correctly, this means that the numbers used are "easier" for the computer to deal with and significantly reduces rounding errors and computer memory overhead.

Whilst the above has been described for translational Langevin dynamics, the same concept applies to rotational Langevin dynamics that are dependent on orientation and angular momenta instead of position and velocity. In LAMMPS, the translational Langevin thermostat is applied through a single command-line within the input script for simulations [20]. In order to apply the Langevin thermostat for orientational degrees of freedom (DoF), an additional *angmom* keyword is added, followed by a numeric scale factor. This is generally taken as a factor of 10 for the CG-DNA input scripts, which means the friction coefficient is ten times larger for orientational DoFs as it is for translational DoFs.

[8] SI Units - Standard International Units.

III. Methodology

The large majority of the Project focused on improving the efficiency of the CG-DNA source code within LAMMPS. Once this was completed and confirmed to operate as desired, it was then used for the purpose of simulating a double-standed DNA sequence featuring a intrastrand hairpin comprised of CTG base-pairs. The main steps taken to fulfil these outcomes are described in this section.

A. Development of the LAMMPS CG-DNA Source Code Improvements

This section will be sub-divided into various categories. Firstly, the detection of the performance flaw within the original source code is discussed, and secondly, the steps taken to rectify this are explained.

1. Detection of the Performance Flaw

Source code can be run through a "profiling tool" that provides details on the time taken by the code to run through its individual subroutines. This is an effective tool available to developers and software engineers that allows them to highlight potential bugs in their code or areas that could offer potential performance improvements through source code optimisation.

As part of his 2018 paper [2], Dr. Henrich performed such an analysis on the original oxDNA model. This highlighted that a substantial percentage of the time taken to run the model was spent within the "MathExtra::q_to_exyz" subroutine (See Figure 7). This was 11.3% in the lesser parallelised of the two runs, representing fewer communication calls between processes[9]. This "MathExtra::q_to_exyz" represents the conversion of the quaternions into local-reference-frame x,y,z coordinates.

The "MathExtra::q_to_exyz" function is in itself a large matrix operation. Consider the 4-Dimensional form of the quaternion from Equation 4, to simply be written as:

$$q = s + a\hat{\boldsymbol{i}} + b\hat{\boldsymbol{j}} + c\hat{\boldsymbol{k}} = \begin{bmatrix} q_0 & q_1 & q_2 & q_3 \end{bmatrix} \tag{15}$$

And the function returns x,y,z:

$$x = \begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix} \quad , \quad y = \begin{bmatrix} y_1 & y_2 & y_3 \end{bmatrix} \quad , \quad z = \begin{bmatrix} z_1 & z_2 & z_3 \end{bmatrix} \tag{16}$$

The full conversion is given by:

$$\begin{bmatrix} q_0 & q_1 & q_2 & q_3 \end{bmatrix} \begin{bmatrix} q_0 & q_3 & -q_2 & -q_3 & q_0 & q_1 & q_2 & -q_1 & q_0 \\ q_1 & q_2 & q_3 & q_2 & -q_1 & q_0 & q_3 & -q_0 & -q_1 \\ -q_2 & q_1 & -q_0 & q_1 & q_2 & q_3 & q_0 & q_3 & -q_2 \\ -q_3 & q_0 & q_1 & -q_0 & -q_3 & q_2 & q_1 & q_2 & q_3 \end{bmatrix} = \begin{bmatrix} x_1 & x_2 & x_3 & y_1 & y_2 & y_3 & z_1 & z_2 & z_3 \end{bmatrix} \tag{17}$$

Thus, the terms can be re-written as:

[9] This goes back to the principles of CPU cores. When more cores are allocated to a job submission and more processes are initiated, there is inherently greater information exchange between the processes. This is why software can't simply be given more cores to run faster. The number of processes generated has to be balanced to size of the simulation, so that the amount of time spent on calculations is well matched to the time spent communicating the information.

$$
\begin{aligned}
x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} &= \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 \\ 2(q_1 q_2 + q_0 q_3) \\ 2(q_1 q_3 - q_0 q_2) \end{bmatrix} \\
y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} &= \begin{bmatrix} 2(q_1 q_2 - q_0 q_3) \\ q_0^2 - q_1^2 + q_2^2 - q_3^2 \\ 2(q_2 q_3 + q_0 q_1) \end{bmatrix} \\
z = \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} &= \begin{bmatrix} 2(q_1 q_3 + q_0 q_2) \\ 2(q_2 q_3 - q_0 q_1) \\ q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix}
\end{aligned}
\tag{18}
$$

Strictly speaking, the "MathExtra::q_to_exyz" function is a "void" function within the source code, and so it does not return anything. Instead, it references values through the use of "pointers". "Pointers" are a useful feature within the C++ language that allow an already existing point in memory to be accessed. So the "MathExtra::q_to_exyz" function is passed the memory locations of the declared quaternion and x,y,z arrays, then sets the x,y,z values as per Equation 18.

The main objective from here was to figure out how to reduce these calculations. It is necessary to again refer to the original source code for insight.

Recall the from Section II C that the potentials within the model are primarily broken down into: the "FENE Bond", Stacking and Hydrogen-Bonded Potentials, Cross-Stacking and Coaxial-Stacking Potentials, and the Excluded Volume Potentials.

In LAMMPS, bonded interactions are given by *"bond_*"* files, and non-bonded pairwise interactions are given by *"pair_*"* files. Each of these files runs through a loop structure that considers all

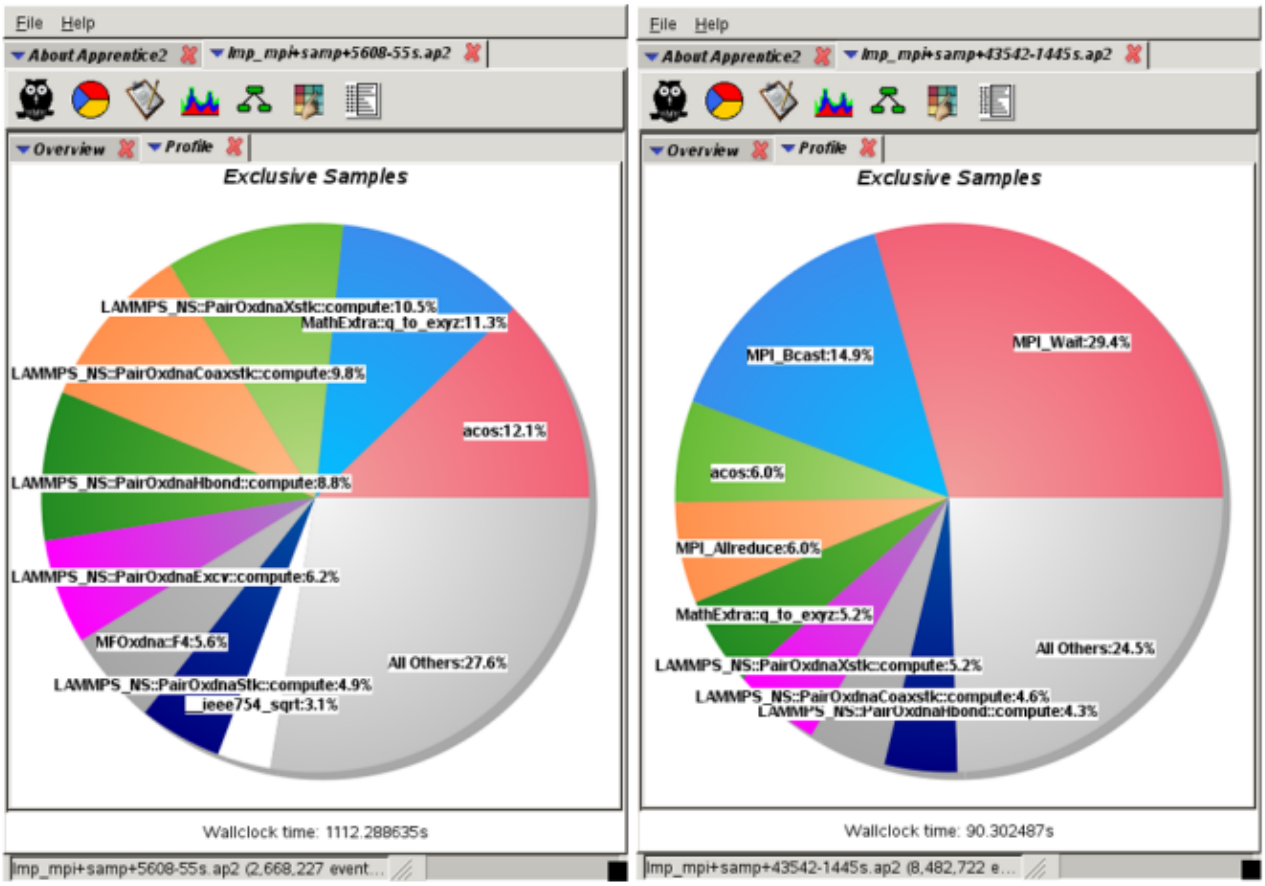


Figure 7: Profiling of the Original LAMMPS CG-DNA Source Code Performance [2]. A low-density run of 60kbp (kilo-base-pairs) is show on 24-Cores (left) and 2048-Cores (right).

the relevant particle interactions, and this is different depending on the interaction. For example, the FENE bond file will run through a list contained within LAMMPS that dictates which backbones are connected to one another. And most of the other interactions rely on a neighbour list, which dictates what particles are within a sufficient proximity to one another to warrant calculating an interaction. The original source code calculated the "MathExtra::q_to_exyz" function at each step within the process, then discarded the result at the end of the loop. This is summarised in Figures 8 and 9, only considering the functions' purpose within the context of the quaternion conversion.

Figure 8 Original Bonded Interaction Compute Procedure (PseudoCode)

```
1:  procedure COMPUTE
2:      i = length of bondlist
3:      for i in length of bondlist do
4:          a = bondlist[i][0]
5:          b = bondlist[i][1]
6:          Call the Quaternion qa associated to a
7:          Call the Quaternion qb associated to b
8:          Perform the "MathExtra::q_to_exyz" function on qa
9:          Perform the "MathExtra::q_to_exyz" function on qb
10:         Use the x,y,z positions from the conversions throughout the rest of the function
```

In Figure 8, it can be seen only a single loop exists. The loop goes from one particle in the list of bonded particles, to the next. There is overhead here in calling and converting the quaternions, although the same quaternion is converted and discarded approximate twice (approximately as this is applicable to all but the two bonds at the beginning and end of the list). However, the repeated quaternion conversions are far more severe in Figure 9. In this case, there is a double loop structure that exists, and in addition, the list is not only of two particles in a bond, but rather all of a particle's neighbours! Testing revealed that this was approximately 18 neighbours per particle at lower salt concentrations where the Debye length is large (larger radius of neighbours).

Figure 9 Original Pairwise Interaction Compute Procedure (PseudoCode)

```
1:  procedure COMPUTE
2:      i = length of particles to loop over
3:      for i in length of particles to loop over do
4:          a = list index
5:          Call the Quaternion qa associated to a
6:          Perform the "MathExtra::q_to_exyz" function on qa
7:          j = number of neighbours
8:          for j in length of number of neighbours do
9:              b = neighbouring particle index
10:             Call the Quaternion qb associated to b
11:             Perform the "MathExtra::q_to_exyz" function on qb
12:             Use the x,y,z positions from the conversions throughout the rest of the function
```

These compute functions are called once each per timestep, and during which the quaternions do not change. Furthermore, each particle has its own associated quaternion. From this information, it can be deduced that the quaternions should not be repeatedly calculated and discarded after use, but rather should be dealt with on a once per particle and once per timestep basis. They need somewhere to be stored within the code, and must have the capability to be called and used between processes as required.

2. *Updating and Improving the Quaternion Handling*

One of the first important decisions to be made was finalising where the converted quaternions could be stored. It is worth first considering the general timestep behaviour within LAMMPS, shown in Figure 10.

The "fix->initial_integrate" and "fix->post_integrate" are essentially Steps 1 and 2 from the description of the Velocity Verlet algorithm in Section II D 2. Next, a large number of functions are called that were not explained as part of the mathematical description of the Verlet. These functions mostly serve the purpose of generating the correct and updated neighbour lists for the particles, and decides what information will be stored where within the computer. As mentioned in Section II D 2, the next main step in the Verlet is to update the forces (and torques) for the new and updated positions (and quaternions) - these are all handled within the "*compute()" functions shown in Figure 10. The last function of importance to the Verlet steps (representing Step 4) is the "fix->final_integrate", which updates velocities (and angular momenta) to complete the main calculations for that timestep.

With this sequential layout in mind, there are two intuitive places where the quaternion conversions could be calculated and stored; the first being somewhere within Step 2 of the Verlet, and the second being somewhere within Step 3 of the Verlet.

However, the issue with Step 2 is that the functions which handle the communication calls and generation of neighbour lists have not been executed yet. The more practical location for the quaternion conversion to take place is therefore within Step 3 of the Verlet where the compute functions are called. This is the same place as where the original code dealt with quaternion conversion, although the question still remains where to store the values.

It was determined, that if each of the compute functions where consistently called in the same sequential order, then the local reference frame co-ordinates from the quaternion conversion could potentially be stored here. Additionally, this would rely on the *"pair_*"* and *"bond_*"* files containing a natural way to utilise LAMMPS' inter-process communication features for scalable parallelisation.

By simply using print statements, the source code was ran with console outputs at the start of each compute function. This was a fast way to determine if these functions where consistently

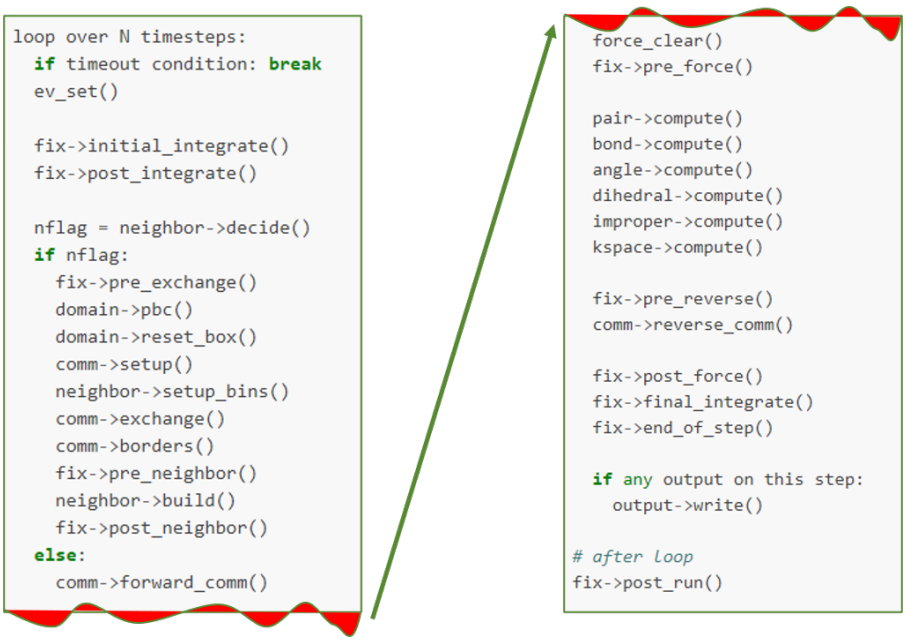


Figure 10: The Format of a Timestep within LAMMPS [4]. The initial Verlet steps are handled by "initial_integrate" and ">post_integrate", the "compute" functions handle the Verlet's force routines, and the "final_integrate" handles the final step of the Verlet integration.

called in a sequential manner. The results showed that they did, and that the compute function within the "excluded volume" pair file was always the first one to be called. As for the specifics on how to implement the coding structure for this, some inspiration was gained from the LAMMPS documentation and the source files of other packages.

In the C++ language, files are usually split into two files for development: a main file (.cpp), and a header file (.h). When the code is being compiled into machine code and the final executable software is being generated, these are combined into an output file (.o). The purpose of the .cpp/.h files is not only to tidy up the code to make it more presentable to read, but also define an interface within the header where the implementation can be performed in the main file - this helps prevent errors from potentially ambiguous definitions and commands. This means that the storage of the quaternion conversion results can be saved to values within the header file of the excluded volume pair style. If the code was simply limited to a single core and did not have any parallelisation, the task would be essentially completed now. Whenever these values where required, they could easily be called upon through the use of pointers.

However, in parallel, this is not enough. Particles (simple known as atoms in LAMMPS) are stored in one process only (owned atoms), and other atoms that process requires information on but does not own instead have their details saved (ghost atoms). All other atoms outwith a cut-off range are disregarded. Each process must, therefore, have a way to obtain the values it requires from other process in order to perform is calculations.

To understand how communication of these values was achieved, the *"comm->forward_comm()"* and *"comm->reverse_comm()"* functions must first be understood. The "forward" communication sends information from owned atom attributes to neighbouring processes to become attributes of their ghost atoms. Similarly, the "reverse" communication does the opposite of this: computed ghost atom attributes are sent back to the process that owns the atom [21]. This crucial for the correct operation of the "newton" command in LAMMPS.

This "newton" command can be added to its input script for simulations. This turns on or off Newton's Third Law for pairwise and bonded interactions. The optimal choice here is situational and depends on the exact simulation parameters, problem size, the machine's compute-to-communication ratio, and the force cut-off lengths. With "newton on", if two interacting atoms are on different processes, then information is calculated on one and communicated to the other. In contrast, "newton off" instead has each process compute the interaction separately and the information is not communicated. Disregarding round-off errors, the numerical answers from both settings should be identical. So with "newton on", the purpose of the "reverse" communication is to sum the partial forces on ghost atoms, and add to the complete force on owned atoms.

With regards to the quaternion conversion values however, there is no need to sum these values and doing so would lead to errors - they are fixed positions. For this reason, these co-ordinates are communicated via "forward" communication only. This means that quaternion conversion values are communicated from owned atoms to all the other processes that require that information, where the neighbours add these values as attributes of their ghost atoms. This operates correctly for both "newton on" and "newton off".

The "forward" communication was achieved through two functions added to the excluded volume cpp file: "pack_forward_comm" and "unpack_forward_comm". The packing adds the quaternion conversion values to a memory buffer on a given process, which is then unpacked by a neighbouring process for its use.

However, these above communication steps are exclusively completed within the excluded volume pair style only. Thankfully, these values are now easily extracted by the other pair and bond styles through the use of various pointers. Each of these files can extract a local copy of the quaternion conversion values from their location within the excluded volume pair style.

Figures 11-13 show the new loop structure employed within the updated source code.

Figure 11 The Once-per-Timestep Quaternion Conversion within the Excluded Volume Pair Style (PseudoCode)

```
1: procedure COMPUTE
2:     for loop over all local atoms do
3:         n = local atom index
4:         Call the Quaternion qn associated to n
5:         Perform the "MathExtra::q_to_exyz" function on qn
6:         Store these values in their global location under "nx","ny","nz"
7:     Continue with the rest of the compute function, calling "nx","ny","nz" when necessary
```

Figure 12 Improved Bonded Interaction Compute Procedure (PseudoCode)

```
1: procedure COMPUTE
2:     i = length of bondlist
3:     for i in length of bondlist do
4:         a = bondlist[i][0]
5:         b = bondlist[i][1]
6:         Extract and use the local reference frame "n(x/y/z)" positions throughout the rest of
7:         the function, indexing accordingly to "a/b(x/y/z)"
```

Figure 13 Improved Pairwise Interaction Compute Procedure (PseudoCode)

```
1: procedure COMPUTE
2:     i = length of particles to loop over
3:     for i in length of particles to loop over do
4:         a = list index
5:         Extract and use the local reference frame "n(x/y/z)" positions throughout the rest of
6:         the function, indexing accordingly to "a(x/y/z)"
7:         j = number of neighbours
8:         for j in length of number of neighbours do
9:             b = neighbouring particle index
10:            Extract and use the local reference frame "n(x/y/z)" positions throughout the rest
11:            of the function, indexing accordingly to "b(x/y/z)"
```

All intermediate coding steps, as well as the final updated source code, can be found here on the GitHub repository used to develop these updates [22]. This is a fork of Dr. Henrich's repository [23], which is in of itself a fork of the official LAMMPS repository [24].

For the latest stable release (currently the 29 Sep 2021 release), the "stable" branch of the official LAMMPS repository [24] should be used. However, the updated source code outlined within this Report is currently only available via the development ("develop") branch on the repository. This will likely be updated into the next stable release of LAMMPS over the course of the coming months.

B. Configuration of the 64-Base-Pair CTG-Sequence DNA Hairpin

In this section, the set-up procedure for running the updated Software is described. The purpose of this was not only to act as a test-run for the new code, but more importantly, it aimed to investigate the hairpin dynamics within a short double-stranded DNA (dsDNA) sequence. The intended outcome from this, is to witness any potential "slippage events" like those observed in the literature [13, 14]. These slippage events can occur when the T-junction between the hairpin and main strand is large, thus reducing the associated energy barriers at this point. Base-pairs within the hairpin can break their hydrogen bond (denature), and are free to reconnect (hybridise) within the main sequence. This behaviour is associated to a number of genetic disorders within humans [13, 14], hence the motivation behind the study of these slippage events.

For visual clarity, Figure 14 highlights the geometry of this hairpin formation within a dsDNA sequence[10], and as a side-note, this happens to be the strand at its initial positions for all simulations in this section.

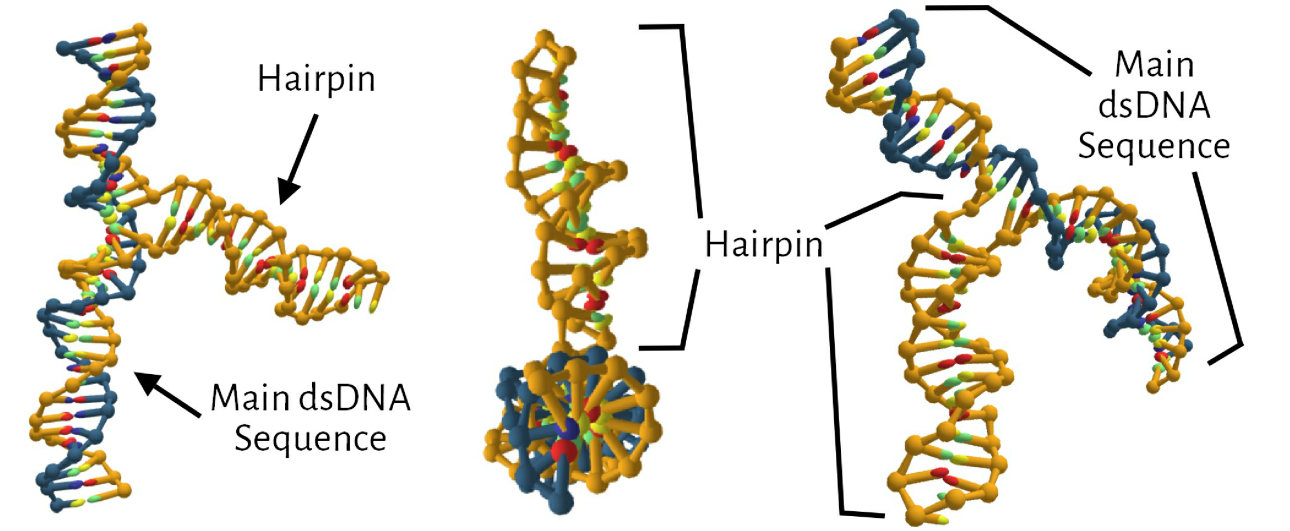


Figure 14: Visual Distinguishment between the Hairpin of a dsDNA Strand and its Main Sequence.

The complete simulation parameters and conditions are given in Section III B 2. However, the addition of "Metadynamics" must first be explained in Section III B 1. Metadynamics offer the capability to accelerate the likelihood of rare events occurring within computational simulations, which is achieved by manipulating the thermodynamic free energy of the system [25]. In its most basic definition, it artificially alters the internal energy of the system against its thermostat. Without introducing metadynamics to the dsDNA-hairpin model, it would take an unrealistic length of time to observe any change-of-state within the T-junction or base-pairs themselves.

1. Applying Metadynamics to the Simulations

To increase the sampling of configuration space (the states by which a system can be in), metadynamics utilise "collective variables". These add a time-dependent energy biases to coarse-gained variables in the form of Gaussian energy packets. A depiction of how this works is shown in Figure 15.

There are various metadynamic methods currently available [25–30], however the specific method used for this simulation is known as "Well-Tempered Metadynamics" [28].

[10] Herein, all non-referenced figures where generated using the DTP software "Serif PagePlus X6", with the DNA images themselves generated from "oxdna-viewer".

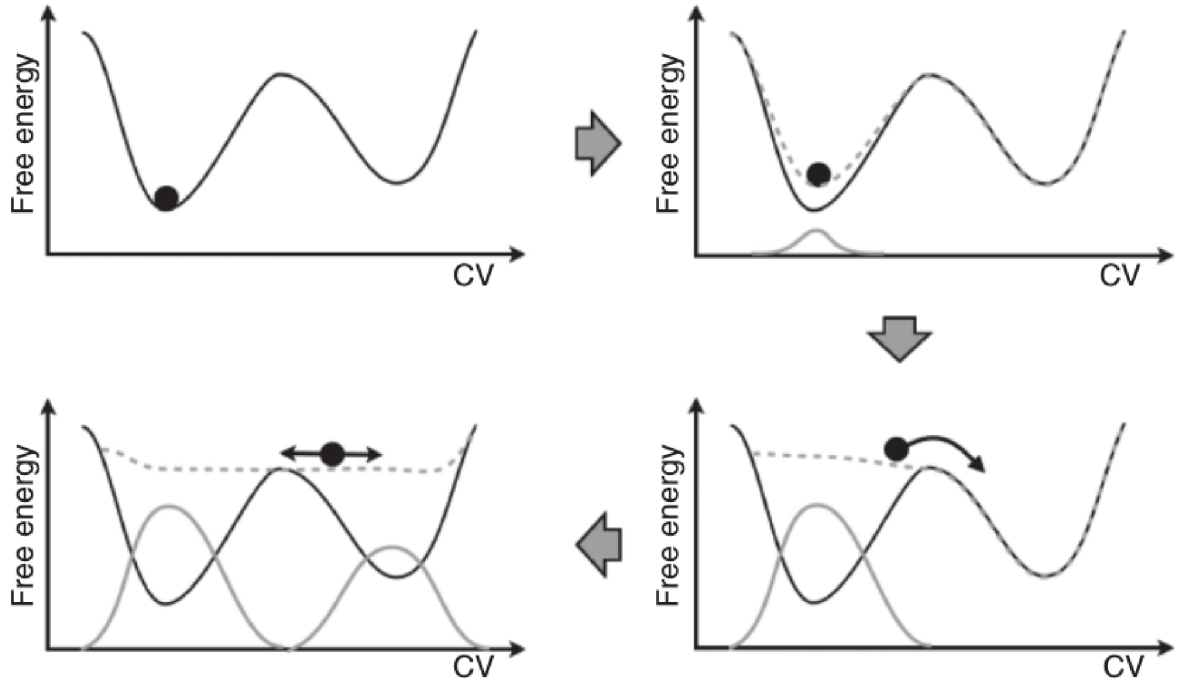


Figure 15: A Sketch of the Process of Metadynamics [26]. First, the system evolves as per its standard dynamics, then a Gaussian potential is introduced (solid grey line) which alters the free-energy landscape (dashed grey line). After some time, the first metastable state is filled and the system can move into the second basin. Once this is all filled, the system evolves into a flat free-energy landscape, where the summation of Gaussian biases provides an estimation of the total free-energy profile.

In the "Well-Tempered" approach, Gaussian potentials (known as "hills") are at first applied in an aggressive fashion, with substantial height and spread. This is then reduced over time, so that each hill applied has less contribution over the free-energy landscape than the previous hill. This leads to faster, and even more accurate, convergence to other states within the configuration space compared to simply applying hills of the same size over a longer time period.

The metadynamics software of choice was "PLUMED" [31] - it has the necessary metadynamics library for the dsDNA-Hairpin simulation, and has plug-in capability into LAMMPS. Once LAMMPS is recompiled with the PLUMED library, its metadynamics can be implemented via two additional input scripts. The first is called by the LAMMPS input script, and defines all the general metadynamic utilities, such as the hills parameters and data output set-up (See Appendix A). Within this PLUMED configuration file, another file is called to define the collective variables (See Appendix B). A "contact map" is defined here, which calculates the distances between a number of atom pairs and transforms this by a switching function to a single scalar output. The purpose of this is to highlight how many bases are hybridised and how many are denatured. Additional information on this can be found in the contact map documentation via the PLUMED website [32].

For reference, PLUMED version 2.7.3 was the library used throughout all simulations in this Project.

2. The Complete Simulation Conditions

As previously stated in Section III A, the efficiency improvements have yet to be updated into a public stable release, although can be found within the "develop" branch of the LAMMPS GitHub [24], or in one of the other two referenced repositories [22, 23]. Any of these repositories can be

used to obtain the Software used in this Project. In addition, the compilation of LAMMPS together with the PLUMED plug-in was performed on Unix based systems; the reason being that many of the features and aspects of these toolkits are not available on Windows. However, a virtual machine was used on a Windows 10 system without issue, using Ubuntu 20.04 on Windows after the addition of the Windows Subsystem for Linux version 2 (WSL2) package.

In order to run the simulation, the following packages must be included when compiling LAMMPS: asphere, molecule, cg-dna, and plumed. This is called in terminal via the command line:

```
make yes-asphere yes-molecule yes-cg-dna yes-plumed
```

Depending how one installs the plumed library, LAMMPS may also require a reference to the plumed installation directory. There is an automated command within LAMMPS that handles this, however it was found to return errors during compilation that prevented the ability to run in parallel. Instead, plumed was installed separately, and then referenced into the LAMMPS compilation via:

```
make lib-plumed args="-p -DIRC_OF_PLUMED_INSTALL-"
```

Where *-DIRC_OF_PLUMED_INSTALL-* is replaced with the actual directory of wherever the plumed installation exists.

After successful compilation of the LAMMPS executable, the Software is ready to be used. LAMMPS requires two files as inputs: an input file containing the simulation parameters, and also a data file containing information of the particles within the simulation. The number of cores is set via the "-np" argument, and since the dsDNA-Hairpin simulation was formed of a relatively short strand of only 64-base-pairs, all runs where given two cores to utilise. Any more is unnecessary or even detrimental to performance, as this would incur a sub-optimal calculation-to-communication ratio. For reference, the Software was ran using the command line:

```
mpirun -np 2 ./lmp_mpi < ./RunX/in.trplet ./RunX/data.ctg_10
```

Shorter 200M timestep runs where first completed in order to determine an appropriate bias factor for the simulation, defined within the PLUMED configuration file. When the bias factor is too low, the metadynamics converge too low within the free-energy landscape (FES) and fail to explore the necessary states to observe slippage events. In contrast, bias factors that are too large converge well beyond the ideal FES, and can cause the system to lock into higher-energy states that that overlook finer details. These test runs where conducted at bias factors of: 4,6,8,10,12, and 14. A bias factor of 10 produced the best results, and so this was then applied to 10 separate runs for an increased duration of 500M timesteps. These final 10 runs where identical, with the exception of their Langevin thermostat, given in the LAMMPS input script as:

```
fix 2 all langevin ${T} ${T} 0.25 -SEED- angmom 10
```

Here, T represents the temperature of the system, and the 0.25 is the damping parameter. As described in Section II D 3, the *angmom* and its scale factor (10 here) applies the Langevin thermostat to orientational DoFs.

As for the *-SEED-*, this is given by a 6-figure integer. Each of the 10 runs had a different value imputed here, which results in different random forces being modelled for both the translational and orientational Langevin thermostats. The purpose of this is to simulate and account for variance, as would be expected in a real-world experiment.

IV. Results

All results are shown in this section. The source code improvements are first discussed, then secondly both visual and numerical conclusions are drawn from the dsDNA-Hairpin simulations.

A. Time-Reductions of the Updated LAMMPS CG-DNA Source Code

Overall, performance improvements varied between around 10% to 20%. This is quite a significant difference, especially considering that many of these DNA simulations can often take multiple days to complete, even on a dedicated compute cluster.

One of the first trail runs is shown in Figure 17. These show the "Total Wall Time" in the final console line output from both the old and new source code, which signifies the total simulation completion time and equates to a difference of around 18%. In order to confirm the absence of any errors in the numerical data, note the top line in each console - all values are exactly the same between the two runs. These represent the energies associated to a given timestep, and displays: timestep, kinetic energy, rotational energy, potential energy, and total energy. If there where any errors between either the calculation or communication of the quaternion conversions, these values would be different.

Figure 17 was conducted under the "newton on" condition, although "newton off" had to be tested as well to confirm the amendments to the code where correct under these conditions. This is shown between the old code in Figure 18 and the new code in Figure 19. Again, as all the energy values where indistinguishable between the two, the code was confirmed to function as desired and without errors. Figure 17 was run on 4-cores, and Figures 18 and 19 on 2 cores.

Quite conveniently, the primary LAMMPS developers require that all packages must provide an "examples" folder containing examples of input and data scripts for a given package. Naturally, this was chosen as a benchmark suite to compare the performance times between the old and new code, shown in Figure 16. Since, collectively, these cover all pair and bond styles with the CG-DNA package, these also acted as a final test run against any numerical errors, of which there where none in the final code review.

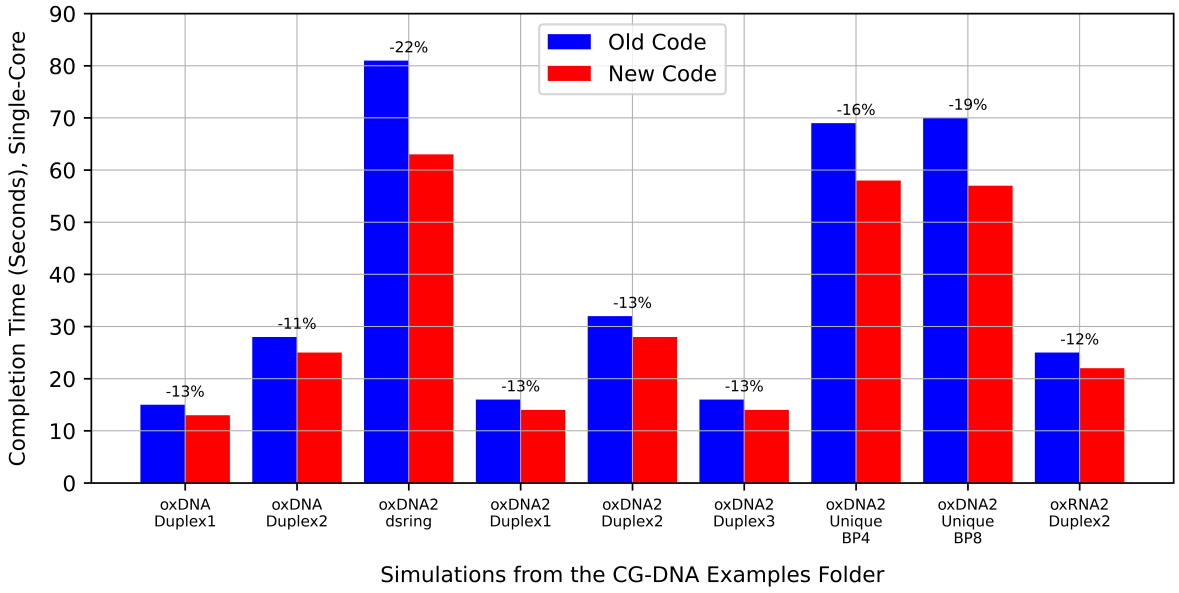
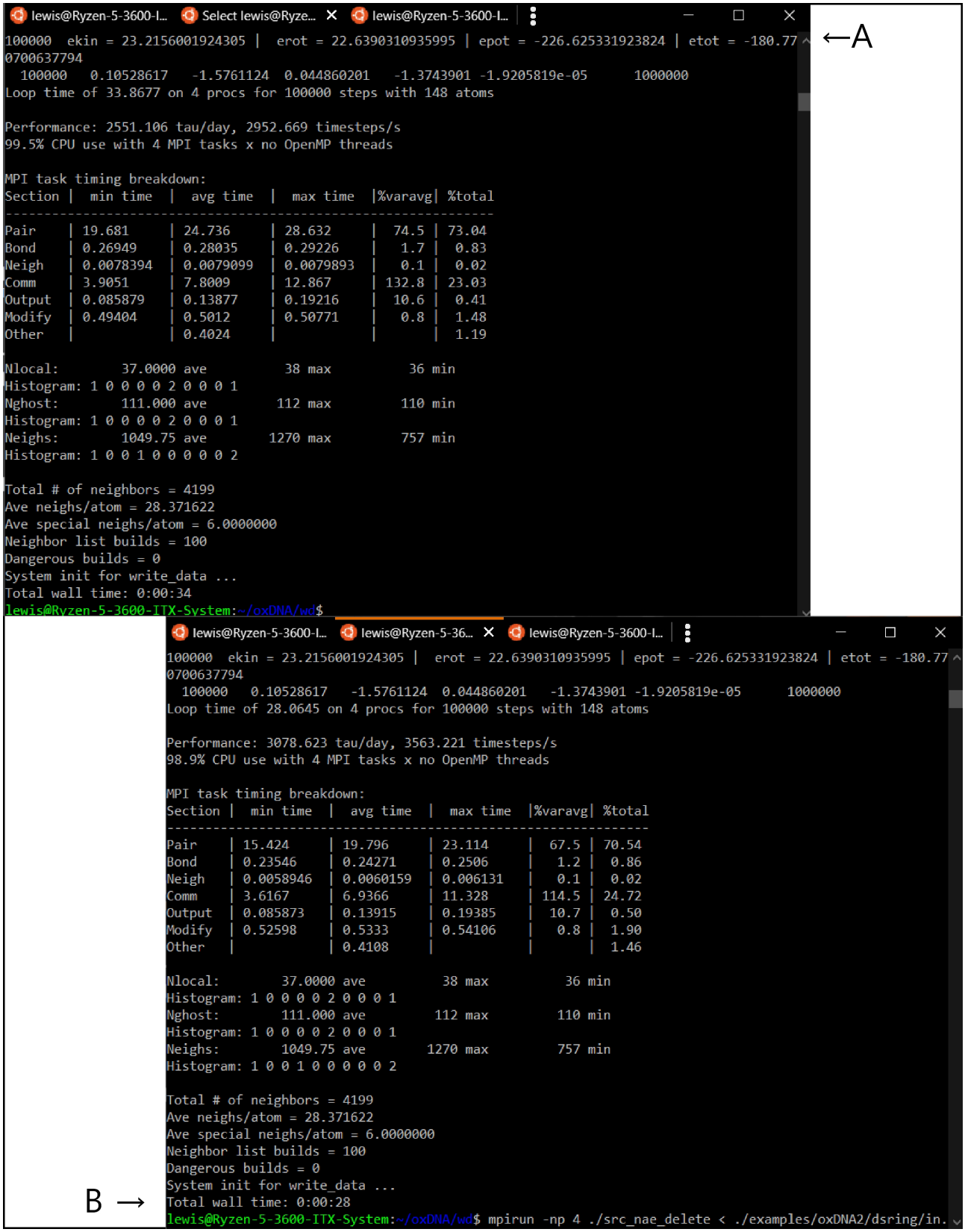


Figure 16: Time Comparisons Between the Old and New Source Code. The time reductions as a percentage are displayed above each pair of bars.




```
100000  ekin = 23.2156001924305 |  erot = 22.6390310935995 | epot = -226.625331923824 | etot = -180.77
0700637794
  100000   0.10528617   -1.5761124  0.044860201   -1.3743901 -1.9205819e-05      1000000
Loop time of 33.8677 on 4 procs for 100000 steps with 148 atoms

Performance: 2551.106 tau/day, 2952.669 timesteps/s
99.5% CPU use with 4 MPI tasks x no OpenMP threads

MPI task timing breakdown:
Section |  min time  |  avg time  |  max time  |%varavg| %total
---------------------------------------------------------------
Pair    | 19.681     | 24.736     | 28.632     |  74.5 | 73.04
Bond    | 0.26949    | 0.28035    | 0.29226    |   1.7 |  0.83
Neigh   | 0.0078394  | 0.0079099  | 0.0079893  |   0.1 |  0.02
Comm    | 3.9051     | 7.8009     | 12.867     | 132.8 | 23.03
Output  | 0.085879   | 0.13877    | 0.19216    |  10.6 |  0.41
Modify  | 0.49404    | 0.5012     | 0.50771    |   0.8 |  1.48
Other   |            | 0.4024     |            |       |  1.19

Nlocal:        37.0000 ave          38 max          36 min
Histogram: 1 0 0 0 0 2 0 0 0 1
Nghost:        111.000 ave         112 max         110 min
Histogram: 1 0 0 0 0 2 0 0 0 1
Neighs:        1049.75 ave        1270 max         757 min
Histogram: 1 0 0 1 0 0 0 0 0 2

Total # of neighbors = 4199
Ave neighs/atom = 28.371622
Ave special neighs/atom = 6.0000000
Neighbor list builds = 100
Dangerous builds = 0
System init for write_data ...
Total wall time: 0:00:34
lewis@Ryzen-5-3600-ITX-System:~/oxDNA/wd$
```

```
100000  ekin = 23.2156001924305 |  erot = 22.6390310935995 | epot = -226.625331923824 | etot = -180.77
0700637794
  100000   0.10528617   -1.5761124  0.044860201   -1.3743901 -1.9205819e-05      1000000
Loop time of 28.0645 on 4 procs for 100000 steps with 148 atoms

Performance: 3078.623 tau/day, 3563.221 timesteps/s
98.9% CPU use with 4 MPI tasks x no OpenMP threads

MPI task timing breakdown:
Section |  min time  |  avg time  |  max time  |%varavg| %total
---------------------------------------------------------------
Pair    | 15.424     | 19.796     | 23.114     |  67.5 | 70.54
Bond    | 0.23546    | 0.24271    | 0.2506     |   1.2 |  0.86
Neigh   | 0.0058946  | 0.0060159  | 0.006131   |   0.1 |  0.02
Comm    | 3.6167     | 6.9366     | 11.328     | 114.5 | 24.72
Output  | 0.085873   | 0.13915    | 0.19385    |  10.7 |  0.50
Modify  | 0.52598    | 0.5333     | 0.54106    |   0.8 |  1.90
Other   |            | 0.4108     |            |       |  1.46

Nlocal:        37.0000 ave          38 max          36 min
Histogram: 1 0 0 0 0 2 0 0 0 1
Nghost:        111.000 ave         112 max         110 min
Histogram: 1 0 0 0 0 2 0 0 0 1
Neighs:        1049.75 ave        1270 max         757 min
Histogram: 1 0 0 1 0 0 0 0 0 2

Total # of neighbors = 4199
Ave neighs/atom = 28.371622
Ave special neighs/atom = 6.0000000
Neighbor list builds = 100
Dangerous builds = 0
System init for write_data ...
Total wall time: 0:00:28
lewis@Ryzen-5-3600-ITX-System:~/oxDNA/wd$ mpirun -np 4 ./src_nae_delete < ./examples/oxDNA2/dsring/in.
```

Figure 17: An Example Comparison Between the Old and New Source Code. Note the last console outputs stating the 34-second completion time in A (Original Code), and the 28-second completion time in B (New Code). This is an improvement equating to an 18% reduction in simulation completion time.

```
1000000  ekin = 1.96656123354318 |  erot = 2.75024292256728 | epot = -19.9991166182833 | etot = -15.2823124621728
 1000000  0.087402721   -1.2738149  0.023870123   -1.1270347 -3.75617e-05        64000
Loop time of 25.9304 on 2 procs for 1000000 steps with 16 atoms

Performance: 33319.925 tau/day, 38564.728 timesteps/s
98.4% CPU use with 2 MPI tasks x no OpenMP threads

MPI task timing breakdown:
Section |  min time  |  avg time  |  max time  |%varavg| %total
---------------------------------------------------------------
Pair    | 19.187     | 19.208     | 19.229     |   0.5 | 74.08
Bond    | 0.51717    | 0.51808    | 0.51899    |   0.1 |  2.00
Neigh   | 0.012504   | 0.012514   | 0.012525   |   0.0 |  0.05
Comm    | 1.4571     | 1.5068     | 1.5565     |   4.0 |  5.81
Output  | 0.54392    | 0.72925    | 0.91458    |  21.7 |  2.81
Modify  | 2.0357     | 2.0824     | 2.1292     |   3.2 |  8.03
Other   |            | 1.873      |            |       |  7.22

Nlocal:        8.00000 ave           8 max           8 min
Histogram: 2 0 0 0 0 0 0 0 0 0
Nghost:        8.00000 ave           8 max           8 min
Histogram: 2 0 0 0 0 0 0 0 0 0
Neighs:        85.0000 ave          85 max          85 min
Histogram: 2 0 0 0 0 0 0 0 0 0

Total # of neighbors = 170
Ave neighs/atom = 10.625000
Ave special neighs/atom = 3.7500000
Neighbor list builds = 1000
Dangerous builds = 0
System init for write_data ...
Total wall time: 0:00:26
lewis@Ryzen-5-3600-ITX-System:~/oxDNA/wd$
```

Figure 18: A "newton off" Example Run Using the Old Source Code.

```
1000000  ekin = 1.96656123354318 |  erot = 2.75024292256728 | epot = -19.9991166182833 | etot = -15.2823124621728
 1000000  0.087402721   -1.2738149  0.023870123   -1.1270347 -3.75617e-05        64000
Loop time of 23.2145 on 2 procs for 1000000 steps with 16 atoms

Performance: 37218.093 tau/day, 43076.497 timesteps/s
98.4% CPU use with 2 MPI tasks x no OpenMP threads

MPI task timing breakdown:
Section |  min time  |  avg time  |  max time  |%varavg| %total
---------------------------------------------------------------
Pair    | 16.673     | 16.748     | 16.823     |   1.8 | 72.14
Bond    | 0.47553    | 0.48447    | 0.4934     |   1.3 |  2.09
Neigh   | 0.010274   | 0.010311   | 0.010348   |   0.0 |  0.04
Comm    | 1.404      | 1.4606     | 1.5173     |   4.7 |  6.29
Output  | 0.52689    | 0.7049     | 0.88291    |  21.2 |  3.04
Modify  | 2.1944     | 2.2065     | 2.2186     |   0.8 |  9.50
Other   |            | 1.6        |            |       |  6.89

Nlocal:        8.00000 ave           8 max           8 min
Histogram: 2 0 0 0 0 0 0 0 0 0
Nghost:        8.00000 ave           8 max           8 min
Histogram: 2 0 0 0 0 0 0 0 0 0
Neighs:        85.0000 ave          85 max          85 min
Histogram: 2 0 0 0 0 0 0 0 0 0

Total # of neighbors = 170
Ave neighs/atom = 10.625000
Ave special neighs/atom = 3.7500000
Neighbor list builds = 1000
Dangerous builds = 0
System init for write_data ...
Total wall time: 0:00:23
lewis@Ryzen-5-3600-ITX-System:~/oxDNA/wd$
```

Figure 19: A "newton off" Example Run Using the New Source Code.

B. Genome Slippage Observed in the CTG-Sequence DNA Hairpin

Using the methods described in Section III B 2, slippage events where observed in 7 out of the 10 runs. Some data analysis of the collective variables, contact map, and free energy landscape also provided insightful information regarding the metadynamic properties of the system.

Figure 20 outlines the contact map for one of these 10 runs. When "cmform" is low and "cmopen" is high, this represents the T-Junction being closed (See Figure 22, Step 1). Inversely, "cmform" being high and "cmopen" low equates to the T-Junction being in an opened state (See Figure 22, Steps 2,3,6, and 7). The slippage events appear to happen incrementally. When the T-junction opens and the surrounding bases are denatured, this is often followed by a few bases from the hairpin re-hybridising into the main strand. As can be seen from the visualisation across timesteps in Figure 22, this is a recurring feature. Similarly, Figure 20 encapsulates this opening and closing of the T-junction via the contact map data. In the runs where slippage events did not occur or where poorly represented, the reversal of the "cmform" and "cmopen" values was minimal. However, Figure 20 shows a clear reversal of these values and slippage events where present. Towards the end of this run, the T-Junction returned to a predominately closed state, and most of the slippage events where seen between the timestep range of 100M to 350M.

In order to quantify the applied energy biases, a heatmap of the free-energy landscape was generated from extracted information contained within the "HILLS" files, shown in Figure 21. The HILLS file contains information regarding the Gaussian parameters for the hills, and where in time they are added. Combining all of this information via post-processing allows the free-energy landscape to be visualised [33, 34].

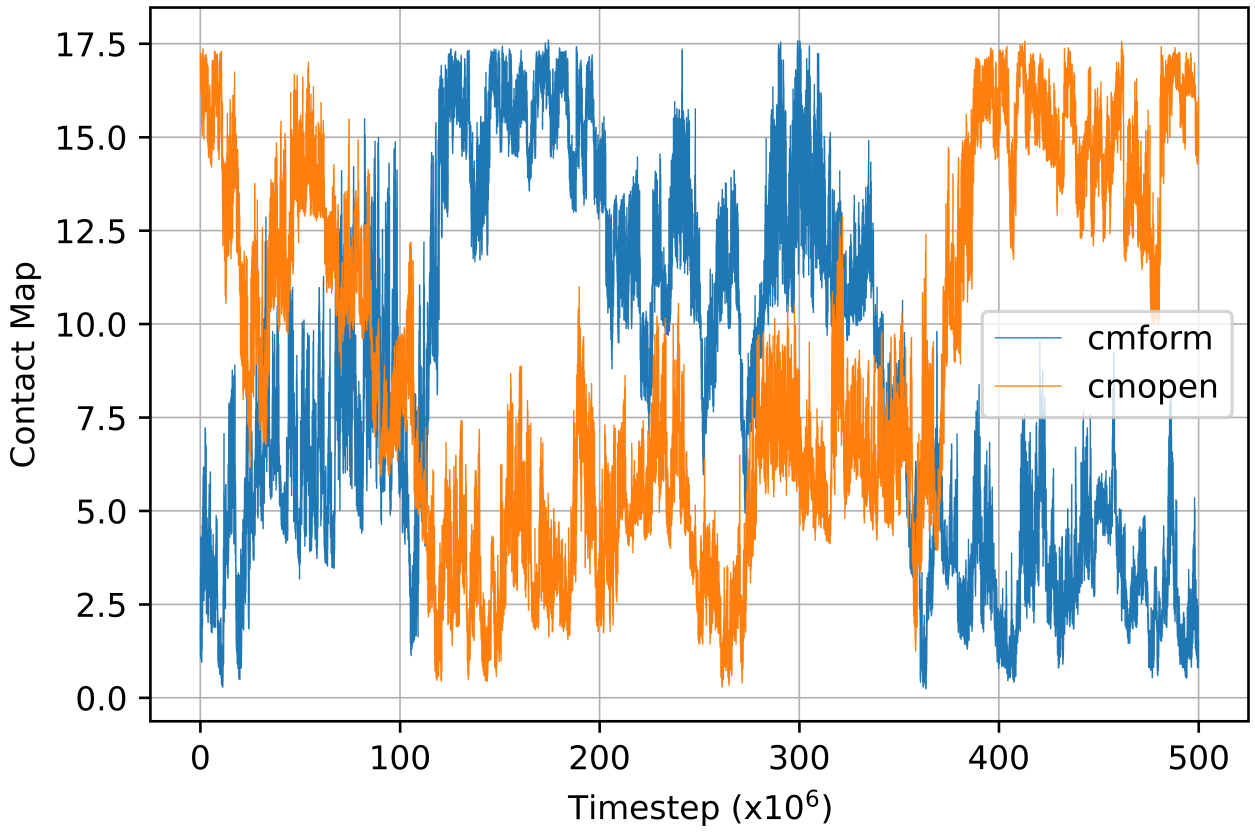


Figure 20: Contact Map Showing the Opened verses Closed State of the dsDNA-Hairpin T-Junction.

Figure 21 depicts the complete free-energy landscape, post-processed from the "HILLS" file information. Two main energy wells are shown to exist, representing the closed or opened state of the dsDNA-Hairpin T-junction. This highlights that a large internal energy within the system is required in order to transition between these states, as can be seen from the blue diagonal within the FES of Figure 21. The most stable states are those closest to having $0K_bT$ free-energy from the hills, and exist when the T-junction is closed and many base-pairs are hybridised - naturally, this makes sense, as the hydrogen bonding in these states is strong and helps keep the base-pairs together. Again, this is why the applied metadynamics are necessary to overcome this high energy potential.

Lastly, Figure 22 is provided which displays snapshots in time from one of the successful simulations. By highlighting sequences of bases from both the hairpin and main dsDNA sequence, it is easy to see that these are initially separated, though are hybridised come the end of the simulation. The hairpin is effectively transversing the main dsDNA sequence, and in doing so, it is altering the genetic information contained within the base-pair sequences of the main strand.

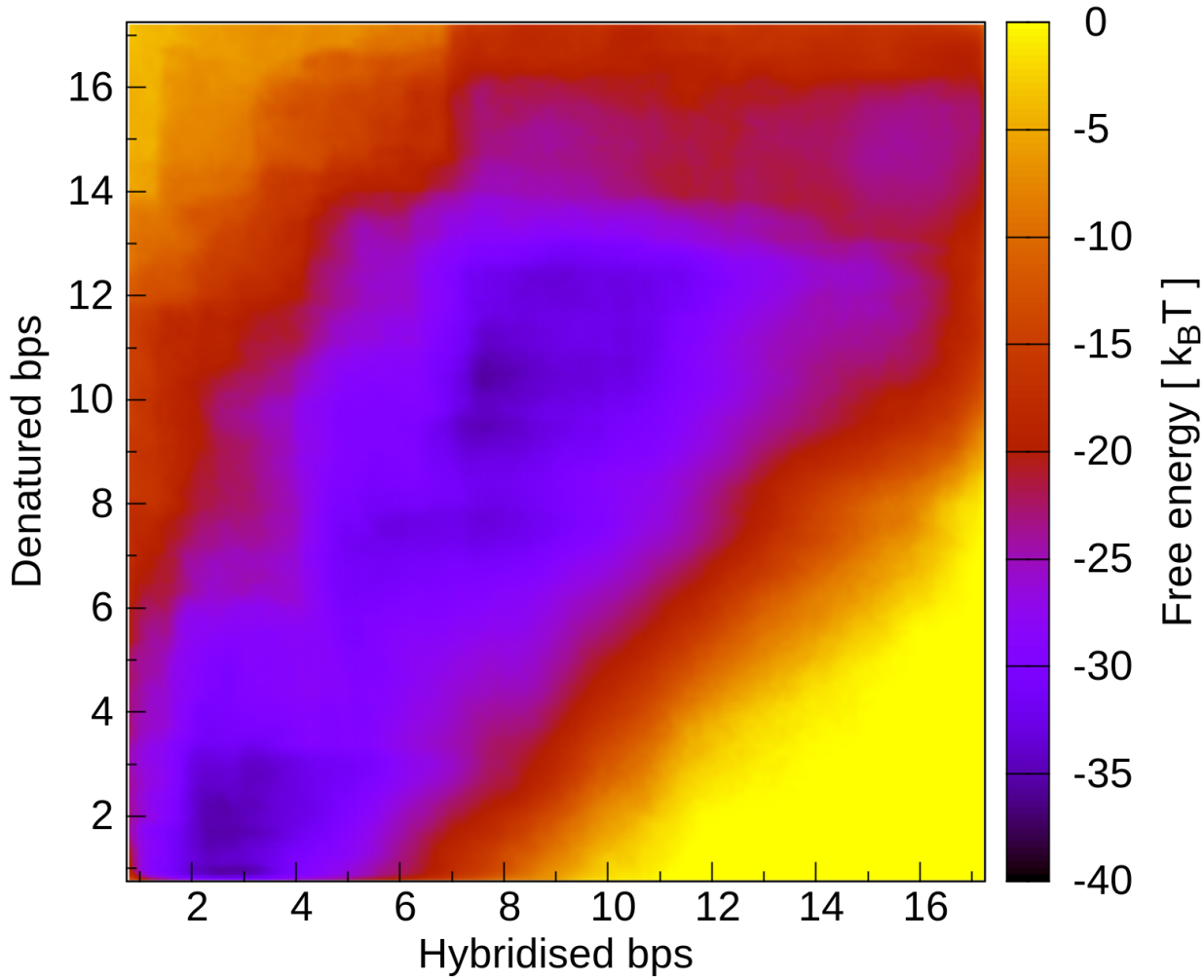


Figure 21: The Free-Energy Landscape Sampled Across Simulations of the dsDNA-Hairpin Sequence. This was sampled and averaged across runs that contained slippage events. There are two key energy potential wells here, occurring around (3,2) and (8,10). The initial state of the system is in the first of these two wells, and the large potential barrier signifies the necessity of the applied metadynamics. With the applied hills, the system can evolve into a state that is conducive to slippage events. This region is represented by the energy well at (8,10).

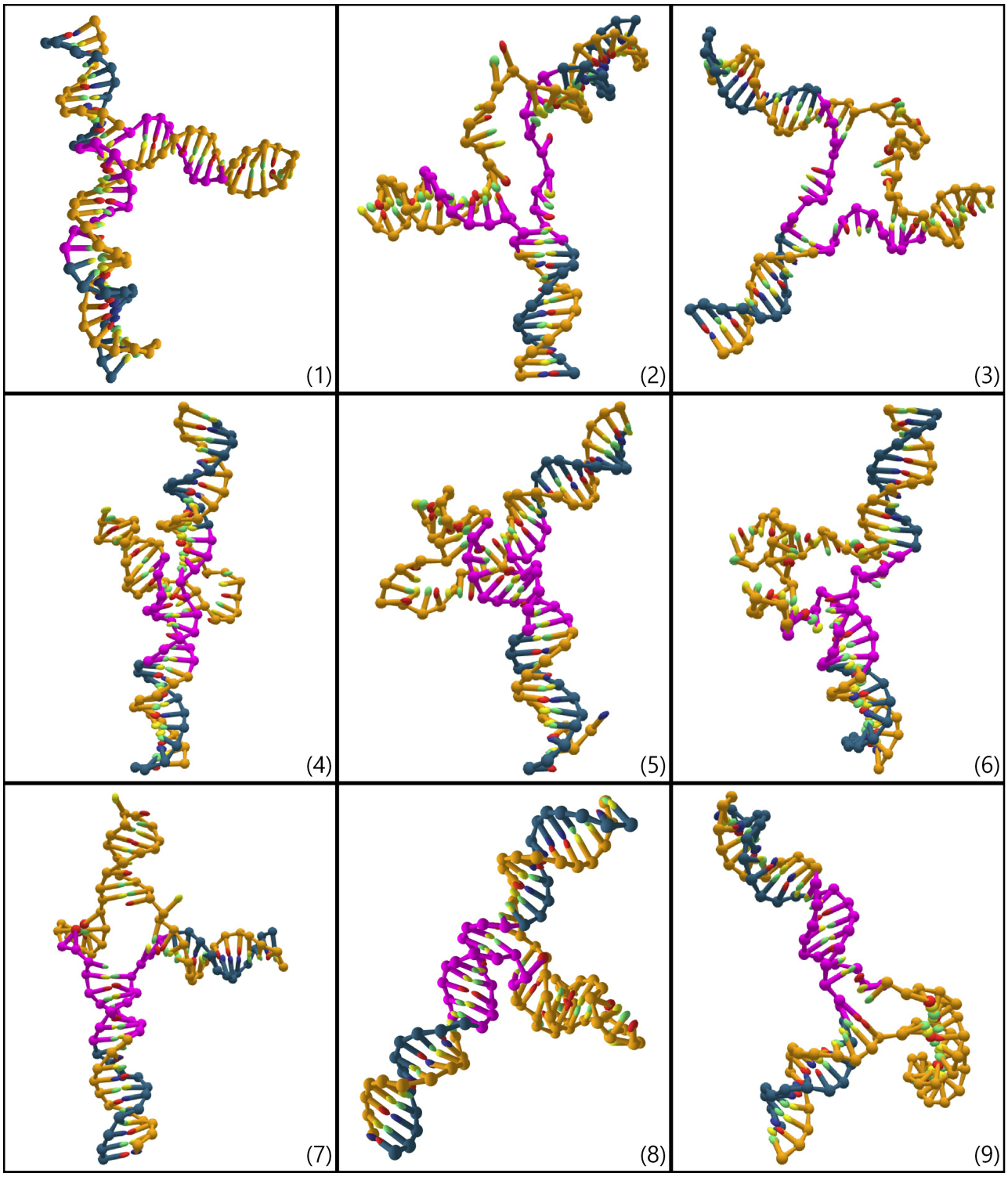


Figure 22: The Time Evolution of the dsDNA-Hairpin Simulation. Step 1 represents the system state at t=0, and Step 9 shows the state at the final timestep of t=500M. Various timestep snapshots within this range are arranged sequentially between Steps 2-8. Sequences of base-pairs are highlighted in purple in order to highlight the slippage event. As can be seen through each snapshot, the hairpin is essentially transversing along the main dsDNA strand. Step 1 clearly shows two separate nucleotide sequences (purple) that are non hybridised: one within the main dsDNA sequence, and the other within the hairpin. By the final timestep (Step 9) within the simulation, the majority of these nucleotides have become hybridised within the main dsDNA sequence.

V. Discussion and Conclusion

As was highlighted in Section IV A, the source code improvements had a significant impact on reducing the simulation times across all test runs, and the productively benefit from this can be well-recognised and appreciated from users of the Software. The results from the dsDNA-Hairpin investigations (Section IV B) warrant greater discussion however.

As was mentioned in Section III B 1, "Well-Tempered" metadynamics where used in these simulations. Whilst this did serve its purpose and allowed for the exploration of slippage events, it had its drawbacks. The user has to manually search for an appropriate bias factor to apply to the Gaussian energy packets, and so the hills will most likely be slightly off from the most optimal bias factor. The overall desired metadynamic effect is still achieved, although some states within the configuration space may be slightly overlooked. There is also the loss of productivity to consider, in the sense that various simulations must be run at different bias factors in order to obtain a practical bias factor for that specific system. All the runs that do not provide the desired results have to be disregarded. In the case of the dsDNA-Hairpin simulations explored in this Project, this equated to around two days of work, accounting for the fact the simulations themselves take many hours to complete.

Going forward, it is suggested that an alternative flavour of metadynamics is used. Perhaps the best candidate for this is "Transition-Tempered" metadynamics (TTMetaD) [30], which is able to achieve asymptotic convergence of the FES without the inadvertent overcompensation of hills (this would otherwise lead to a sacrificed exploration of the complete collective variable space). TTMetaD currently predicts the most accurate and reliable estimates of hill parameters over other metadynamic flavours, and does so in a shorter time-frame. This would reduce the number of timesteps that where required in dsDNA-Hairpin simulations. For reference, each of these runs took approximately 60hrs to complete, and so any reduction to this calculation time would be welcomed.

In TTMetaD, the hills have a variable bias factor that is not only time-dependent like the Well-Tempered approach, but dependent on the free-energy landscape. Unlike the Well-Tempered approach, TTMetaD analyses the potential barriers within the system, and so if a large potential well is present, it will apply a large Gaussian energy packet; and in contrast, areas within the collective variable space that require little energy to overcome only have minuet hills applied in these regions. This is why TTMetaD provides faster and more accurate convergence of the FES.

Overall, these dsDNA-Hairpin simulations provided excellent preliminary insight into the slippage events that can occur within such DNA configurations, regardless of using the somewhat sub-optimal Well-Tempered metadynamics. There is great potential here for further exploration of these dynamics, and it is hoped that this can be studied in greater depth as part of the fifth-year Project. Ultimately, this Project has laid solid foundations for further research in this area, and the objectives hoped to be achieved by the end of the Project have been met and even exceeded. The computation improvements where only expected to achieve a 10% reduction to simulation times at most, which was estimated from the profiling shown in Figure 7 - to gain consistently greater percentage reductions was a welcomed bonus. These source code improvements where also completed ahead of schedule, hence the extra investigation into the hairpin dynamics. This was not originally part of the Project objectives, and has added an astute simulation based aspect to an otherwise purely software development focused project.

References

[1] Lammps website. `https://docs.lammps.org` [Date accessed: 25/03/2022].

[2] Henrich O, Gutiérrez Fosado YA, Curk T, and Ouldridge TE. Coarse-grained simulation of dna using lammps: An implementation of the oxdna model and its applications. *The European Physical Journal. E, Soft Matter*, 41(57), 2018.

[3] Lammps documentation (24 mar 2022 version). `https://docs.lammps.org/Manual.html` [Date accessed: 25/03/2022].

[4] How a timestep works. `https://docs.lammps.org/Developer_flow.html` [Date accessed: 25/03/2022].

[5] Class topology. `https://docs.lammps.org/Developer_org.html#class-topology` [Date accessed: 25/03/2022].

[6] Dans PD, Walther J, Gómez H, and Orozco M. Multiscale simulation of dna. *Current Opinion in Structural Biology*, 37(693710):29–45, 2016.

[7] Kmiecik Sebastian, Gront Dominik, Kolinski Michal, Wieteska Lukasz, Dawid Aleksandra Elzbieta, and Kolinski Andrzej. Coarse-grained protein models and their applications. *Chemical Reviews*, 116(14):7898–7936, 2016. PMID: 27333362.

[8] Ingólfsson H I, Lopez C A, Uusitalo J J, de Jong D H, Gopal S M, Periole X, and Marrink S J. The power of coarse graining in biomolecular simulations. *Wiley interdisciplinary reviews. Computational molecular science*, 4(3):225–248, 2014.

[9] Sengar A, Ouldridge TE, Henrich O, Rovigatti L, and Sulc P. A primer on the oxdna model of dna: When to use it, how to simulate it and how to interpret the results. *Frontiers in Molecular Biosciences*, 8(693710), 2021.

[10] Ouldridge TE. Coarse-grained modelling of dna and dna self-assembly. *Oxford University, UK [PhD thesis]*, 2011.

[11] Benedict E.K. Snodin, Ferdinando Randisi, Majid Mosayebi, Petr Šulc, John S. Schreck, Flavio Romano, Thomas E. Ouldridge, Roman Tsukanov, Eyal Nir, Ard A. Louis, and Jonathan P.K. Doye. Introducing improved structural properties and salt dependence into a coarse-grained model of DNA. *J. Chem. Phys.*, 142(23):1–12, 2015.

[12] Daniel M. Hinckley, Gordon S. Freeman, Jonathan K. Whitmer, Juan J. De Pablo, Biological Engineering, Wisconsin Madison, Daniel M. Hinckley, Gordon S. Freeman, Jonathan K. Whitmer, and Juan J De. An experimentally-informed coarse-grained 3-site-per-nucleotide model of DNA: Structure, thermodynamics, and dynamics of hybridization. *J. Chem. Phys.*, 139:144903, 2013.

[13] Tianyu Hu, Michael J Morten, and Steven W Magennis. Conformational and migrational dynamics of slipped-strand DNA three-way junctions containing trinucleotide repeats. *Nat. Commun.*, 12(204):1–8, 2021.

[14] Masayuki Nakamori, Gagan B. Panigrahi, Stella Lanni, Terence Gall-Duncan, Hideki Hayakawa, Hana Tanaka, Jennifer Luo, Takahiro Otabe, Jinxing Li, et al. A slipped-CAG DNA-binding small molecule induces trinucleotide-repeat contractions in vivo. *Nat. Genet.*, 52(2):146–159, 2020.

[15] J. Vince. *Quaternions for Computer Graphics*. SpringerLink : Bücher. Springer London, 2011. `https://books.google.co.uk/books?id=CDgrYqDsSBAC`.

[16] Watson JD and Crick FH. Molecular structure of nucleic acids: A structure for deoxyribose nucleic acid. *Nature*, 171:737–738, 1953.

[17] Benedict E. K. Snodin, Ferdinando Randisi, Majid Mosayebi, Petr Šulc, John S. Schreck, Flavio Romano, Thomas E. Ouldridge, et al. Introducing improved structural properties and salt dependence into a coarse-grained model of dna. *The Journal of Chemical Physics*, 142(23):234901, 2015.

[18] A. P. Thompson, H. M. Aktulga, R. Berger, D. S. Bolintineanu, W. M. Brown, P. S. Crozier, P. J. in 't Veld, A. Kohlmeyer, S. G. Moore, T. D. Nguyen, R. Shan, M. J. Stevens, J. Tranchida, C. Trott, and S. J. Plimpton. LAMMPS - a flexible simulation tool for particle-based materials modeling at the

atomic, meso, and continuum scales. *Comp. Phys. Comm.*, 271:108171, 2022.

[19] Basics of running lammps. `https://docs.lammps.org/Run_basics.html` [Date accessed: 25/03/2022].

[20] fix langevin command. `https://docs.lammps.org/fix_langevin.html` [Date accessed: 25/03/2022].

[21] Communication. `https://docs.lammps.org/Developer_par_comm.html` [Date accessed: 25/03/2022].

[22] My (lrussell676) lammps repository. `https://github.com/lrussell676/lammps` [Date accessed: 25/03/2022].

[23] Dr. Henrich's lammps repository. `https://github.com/ohenrich/lammps` [Date accessed: 25/03/2022].

[24] The official lammps repository. `https://github.com/lammps/lammps` [Date accessed: 25/03/2022].

[25] Giovanni Bussi and Alessandro Laio. Using metadynamics to explore complex free-energy landscapes. *Nat. Rev. Phys.*, 2(4):200–212, 2020.

[26] Giovanni Bussi and Davide Branduardi. Free-energy calculations with metadynamics: Theory and practice. pages 1–49, 05 2015.

[27] Giovanni Bussi, Alessandro Laio, and Pratyush Tiwary. *Metadynamics: A Unified Framework for Accelerating Rare Events and Sampling Thermodynamics and Kinetics*. 2018.

[28] Alessandro Barducci, Giovanni Bussi, and Michele Parrinello. Well-tempered metadynamics: A smoothly converging and tunable free-energy method. *Phys. Rev. Lett.*, 100(2):1–4, 2008.

[29] Alessandro Laio and Francesco L. Gervasio. Metadynamics: A method to simulate rare events and reconstruct the free energy in biophysics, chemistry and material science. *Reports Prog. Phys.*, 71(12), 2008.

[30] James F. Dama, Grant Rotskoff, Michele Parrinello, and Gregory A. Voth. Transition-tempered metadynamics: Robust, convergent metadynamics via on-the-fly transition barrier estimation. *J. Chem. Theory Comput.*, 10(9):3626–3633, 2014.

[31] Plumed. `https://www.plumed.org/` [Date accessed: 25/03/2022].

[32] Plumed, contact map. `https://www.plumed.org/doc-v2.6/user-doc/html/_c_o_n_t_a_c_t_m_a_p.html` [Date accessed: 25/03/2022].

[33] Bonomi Massimiliano, Bussi Giovanni, Camilloni Carlo, Tribello Gareth, and Banáš Pavel. The plumed consortium. promoting transparency and reproducibility in enhanced molecular simulations. *Nat Methods*, 16:670–673, 05 2019.

[34] Gareth A. Tribello, Massimiliano Bonomi, Davide Branduardi, Carlo Camilloni, and Giovanni Bussi. Plumed 2: New feathers for an old bird. *Computer Physics Communications*, 185(2):604–613, 2014.

A. Appendix: Main PLUMED Configuration File

"plumed.wtmeta2.conf"

```
#########################################
### restart previous simulation ###
#########################################
RESTART

##############
### Units ###
##############
# Since in our LAMMPS script we use LJ units, setting natural units
    here means that
# all quantities are represented in the oxDNA simulation units.
UNITS NATURAL

#########################################
### Define collective variables ###
#########################################
INCLUDE FILE=colvar2.conf

###################################
### Activate metadynamics ###
###################################

metad: METAD ARG=cmform,cmopen  ...
SIGMA=0.05,0.05
HEIGHT=0.1
PACE=250
GRID_MIN=0,0
GRID_MAX=20,20
GRID_SPACING=0.1,0.1
BIASFACTOR=10.0
TEMP=0.1
CALC_RCT
RCT_USTRIDE=1
FILE=HILLS
WALKERS_MPI
...

###################################
### Data before reweighting ###
###################################
# monitor the two variables and the metadynamics bias potential
PRINT STRIDE=1000 ARG=cmform,cmopen,metad.bias,rda FILE=COLVAR

### Bin average over time of the CVs.
av1: AVERAGE ARG=cmform STRIDE=1000 CLEAR=100000
av2: AVERAGE ARG=cmopen STRIDE=1000 CLEAR=100000
```

```
av3: AVERAGE ARG=rda STRIDE=1000 CLEAR=100000
PRINT ARG=av1,av2,av3 STRIDE=100000 FILE=colvar_av_vs_t.dat

### Block analysis of the Probability Distribution Function
hB: HISTOGRAM ARG=cmform,cmopen ...
GRID_MIN=0,0
GRID_MAX=20,20
GRID_SPACING=0.1,0.1
BANDWIDTH=0.1,0.1
STRIDE=1000
CLEAR=0
...
DUMPGRID GRID=hB FILE=histogram.dat STRIDE=1000000 FMT=%8.4f

#################################
### Reweight metadynamics ###
#################################
rw: REWEIGHT_METAD TEMP=0.1

#################################
### Data after reweighting ###
#################################
### Bin average over time of the CVs.
avrw1: AVERAGE ARG=cmform STRIDE=1000 LOGWEIGHTS=rw CLEAR=100000
avrw2: AVERAGE ARG=cmopen STRIDE=1000 LOGWEIGHTS=rw CLEAR=100000
avrw3: AVERAGE ARG=rda STRIDE=1000 LOGWEIGHTS=rw CLEAR=100000
PRINT ARG=avrw1,avrw2,avrw3 STRIDE=1000 FILE=colvar_avrw_vs_t.dat

### Block analysis of the Probability Distribution Function
hBrw: HISTOGRAM ARG=cmform,cmopen ...
GRID_MIN=0,0
GRID_MAX=20,20
GRID_SPACING=0.1,0.1
BANDWIDTH=0.1,0.1
STRIDE=1000
CLEAR=0
LOGWEIGHTS=rw
...
DUMPGRID GRID=hBrw FILE=histogram_rw.dat STRIDE=1000000 FMT=%8.4f
```

B. Appendix: PLUMED ColVar Configuration File

"colvar2.conf"

```
# Rebuild molecules split by PBCs
WHOLEMOLECULES ENTITY0=1-63 ENTITY1=64-96

# Hydrogen-bonds to form during 3 slipped triplets
# Receiving arm
```

```
CONTACTMAP ...
ATOMS1=13,84
ATOMS2=14,83
ATOMS3=15,82
ATOMS4=16,81
ATOMS5=17,80
ATOMS6=18,79
ATOMS7=19,78
ATOMS8=20,77
ATOMS9=21,76
# Hairpin
ATOMS10=22,51
ATOMS11=24,49
ATOMS12=25,48
ATOMS13=27,46
ATOMS14=28,45
ATOMS15=30,43
ATOMS16=31,42
ATOMS17=33,40
ATOMS18=34,39
SWITCH={RATIONAL R_0=1.9 NN=6 MM=10}
LABEL=cmform
SUM
... CONTACTMAP

# Hydrogen-bonds to open during 3 slipped triplets
# Providing arm
CONTACTMAP ...
ATOMS1=43,84
ATOMS2=44,83
ATOMS3=45,82
ATOMS4=46,81
ATOMS5=47,80
ATOMS6=48,79
ATOMS7=49,78
ATOMS8=50,77
ATOMS9=51,76
# Hairpin
ATOMS10=13,42
ATOMS11=15,40
ATOMS12=16,39
ATOMS13=18,37
ATOMS14=19,36
ATOMS15=21,34
ATOMS16=22,33
ATOMS17=24,31
ATOMS18=25,30
SWITCH={RATIONAL R_0=1.9 NN=6 MM=10}
LABEL=cmopen
SUM
... CONTACTMAP
```

```
#We will keep track of other CVs without biasing them
rda: DISTANCE ATOMS=40,89
```

C. Appendix: LAMMPS Input File

"in.triplet"

```
variable number equal 1
variable ofreq  equal 10000
variable efreq  equal 10000
variable T      equal 0.1
variable rhos   equal 0.2

variable nstep  equal 500000000

# ntype = number of atom types
variable ntype equal 64

units lj

dimension 3

newton on

boundary  p p p

atom_style hybrid bond ellipsoid oxdna
atom_modify sort 0 1.0

# Pair interactions require lists of neighbours to be calculated
neighbor 2.0 bin
neigh_modify every 10 delay 0 check yes

read_data data.ctg_10

mass * 3.1575

group all type 1:${ntype}

special_bonds lj 0 1 1

# oxDNA2 bond interactions - FENE backbone
bond_style hybrid oxdna2/fene harmonic
bond_coeff 1 oxdna2/fene 2.0 0.25 0.7564
bond_coeff 2 harmonic 0.0 0.25

# oxDNA2 pair interactions
```

```
pair_style hybrid/overlay oxdna2/excv oxdna2/stk oxdna2/hbond oxdna2
   /xstk oxdna2/coaxstk oxdna2/dh

pair_coeff 1*${ntype} 1*${ntype} oxdna2/excv    2.0 0.7 0.675 2.0
   0.515 0.5 2.0 0.33 0.32
pair_coeff 1*${ntype} 1*${ntype} oxdna2/stk     seqdep ${T} 1.3523
   2.6717 6.0 0.4 0.9 0.32 0.75 1.3 0 0.8 0.9 0 0.95 0.9 0 0.95 2.0
   0.65 2.0 0.65
pair_coeff 1*${ntype} 1*${ntype} oxdna2/hbond   seqdep 0.0 8.0 0.4
   0.75 0.34 0.7 1.5 0 0.7 1.5 0 0.7 1.5 0 0.7 0.46
   3.141592653589793 0.7 4.0 1.5707963267948966 0.45 4.0
   1.5707963267948966 0.45

label loop
variable base loop ${ntype}
  variable basemod equal ${base}%4
  if "${basemod} == 1" then &
    "variable comp equal ${base}+3" &
    "pair_coeff ${base} ${comp} oxdna2/hbond   seqdep 1.0678 8.0 0.4
       0.75 0.34 0.7 1.5 0 0.7 1.5 0 0.7 1.5 0 0.7 0.46
      3.141592653589793 0.7 4.0 1.5707963267948966 0.45 4.0
      1.5707963267948966 0.45"
  if "${basemod} == 2" then &
    "variable comp equal ${base}+1" &
    "pair_coeff ${base} ${comp} oxdna2/hbond   seqdep 1.0678 8.0 0.4
       0.75 0.34 0.7 1.5 0 0.7 1.5 0 0.7 1.5 0 0.7 0.46
      3.141592653589793 0.7 4.0 1.5707963267948966 0.45 4.0
      1.5707963267948966 0.45"
next base
jump in.triplet loop

pair_coeff 1*${ntype} 1*${ntype} oxdna2/xstk    47.5 0.575 0.675
   0.495 0.655 2.25 0.791592653589793 0.58 1.7 1.0 0.68 1.7 1.0 0.68
    1.5 0 0.65 1.7 0.875 0.68 1.7 0.875 0.68
pair_coeff 1*${ntype} 1*${ntype} oxdna2/coaxstk 58.5 0.4 0.6 0.22
   0.58 2.0 2.891592653589793 0.65 1.3 0 0.8 0.9 0 0.95 0.9 0 0.95
   40.0 3.116592653589793
pair_coeff 1*${ntype} 1*${ntype} oxdna2/dh      ${T} ${rhos} 0.815

# NVE ensemble
fix 1 all nve/asphere
fix 2 all langevin ${T} ${T} 0.25 457145 angmom 10

timestep 1e-3

# comm_style tiled
fix 3 all balance 1000 1.03 shift xyz 10 1.01
comm_modify cutoff 3.8

#PLUMED (it is important that the timstep is called before the fix
   plumed)
```

```
fix 5 all plumed plumedfile plumed.wtmeta2.conf outfile log.plumed

compute quat all property/atom quatw quati quatj quatk

compute erot all erotate/asphere
compute ekin all ke
compute epot all pe

compute ebond all bond
compute eexcv all pair oxdna2/excv
compute estk all pair oxdna2/stk
compute ehbond all pair oxdna2/hbond
compute exstk all pair oxdna2/xstk
compute ecoaxstk all pair oxdna2/coaxstk
compute edh all pair oxdna2/dh

variable erot equal c_erot
variable ekin equal c_ekin
variable epot equal c_epot
variable etot equal c_erot+c_ekin+c_epot

variable ebond equal c_ebond[1]
variable eexcv equal c_eexcv
variable estk equal c_estk
variable ehbond equal c_ehbond
variable exstk equal c_exstk
variable ecoaxstk equal c_ecoaxstk
variable edh equal c_edh

fix 6 all print ${efreq} "$(step)  ekin = ${ekin} |  erot = ${erot}
   | ebond = ${ebond} | eexcv = ${eexcv} | estk = ${estk} | ehbond =
    ${ehbond} | exstk = ${exstk} | ecoaxstk = ${ecoaxstk} | edh = ${
   edh} | epot = ${epot} | etot = ${etot}" screen yes

dump out all custom ${ofreq} out.${number}.lammpstrj id mol type x y
    z ix iy iz vx vy vz c_quat[1] c_quat[2] c_quat[3] c_quat[4]
   angmomx angmomy angmomz
dump_modify out sort id append yes
dump_modify out format line "%d %d %d  %22.15le %22.15le %22.15le  %
   d %d %d  %22.15le %22.15le %22.15le  %22.15le %22.15le %22.15le
   %22.15le  %22.15le %22.15le %22.15le"

#restart 1000000 restart.0 restart.1

run ${nstep}

#write_data data.${number}.* nocoeff
#write_restart restart.*
```

D. Appendix: LAMMPS Data File

"data.ctg_10"

```
# LAMMPS oxDNA data file
96 atoms
96 ellipsoids
94 bonds

64 atom types
2 bond types

# System size
-50.00000 50.00000 xlo xhi
-50.00000 50.00000 ylo yhi
-50.00000 50.00000 zlo zhi

Masses

1 3.15750
2 3.15750
3 3.15750
4 3.15750
5 3.15750
6 3.15750
7 3.15750
8 3.15750
9 3.15750
10 3.15750
11 3.15750
12 3.15750
13 3.15750
14 3.15750
15 3.15750
16 3.15750
17 3.15750
18 3.15750
19 3.15750
20 3.15750
21 3.15750
22 3.15750
23 3.15750
24 3.15750
25 3.15750
26 3.15750
27 3.15750
28 3.15750
29 3.15750
30 3.15750
31 3.15750
```

```
32 3.15750
33 3.15750
34 3.15750
35 3.15750
36 3.15750
37 3.15750
38 3.15750
39 3.15750
40 3.15750
41 3.15750
42 3.15750
43 3.15750
44 3.15750
45 3.15750
46 3.15750
47 3.15750
48 3.15750
49 3.15750
50 3.15750
51 3.15750
52 3.15750
53 3.15750
54 3.15750
55 3.15750
56 3.15750
57 3.15750
58 3.15750
59 3.15750
60 3.15750
61 3.15750
62 3.15750
63 3.15750
64 3.15750

# Atom-ID, type, position, molecule-ID, ellipsoid flag, density
Atoms

1 1 -1.6851434183053861e+01 -4.5058711382916776e+00
   7.5951807146183850e+00 1 1 1
2 7 -1.7002860636381389e+01 -4.0011246077618390e+00
   7.5890093538083541e+00 1 1 1
3 12 -1.7165625865030901e+01 -3.5746536821308235e+00
   7.5582157609004996e+00 1 1 1
4 15 -1.7515320163482308e+01 -3.2246928331185796e+00
   7.7791497821683393e+00 1 1 1
5 19 -1.7759295670962477e+01 -2.9540769797318567e+00
   8.2767577020084051e+00 1 1 1
6 24 -1.7786037247189629e+01 -2.8214460525133771e+00
   8.7942791099626643e+00 1 1 1
7 26 -1.7670086463973050e+01 -2.6606442867383628e+00
   9.3361605307788302e+00 1 1 1
```

```
8 29 -1.7348726171838951e+01 -2.4284399325758894e+00
   9.7339526632304239e+00 1 1 1
9 35 -1.6849615841286766e+01 -2.1621602804425937e+00
   9.8329564565966212e+00 1 1 1
10 37 -1.6468664364929424e+01 -1.7453094425286297e+00
   9.7681209038375059e+00 1 1 1
11 42 -1.6317758165141136e+01 -1.2737539646414515e+00
   9.8990539282897920e+00 1 1 1
12 47 -1.6330898393412077e+01 -7.6222173656251435e-01
   9.8585891089841429e+00 1 1 1
13 2 -1.5668810001244005e+01 -6.7735774994027942e-01
   8.9289597276684596e+00 1 1 1
14 1 -1.5408216089356324e+01 -2.8822542589455524e-01
   8.5902412236273094e+00 1 1 1
15 3 -1.5347105580499456e+01 2.5952789094659390e-01
   8.4475408952178430e+00 1 1 1
16 2 -1.5231860551226733e+01 7.7994116354087462e-01
   8.4985621327833840e+00 1 1 1
17 1 -1.4907114411349601e+01 1.0996580718389930e+00
   8.6956176758089860e+00 1 1 1
18 3 -1.4594682337129312e+01 1.4053104119726645e+00
   9.0992058838545624e+00 1 1 1
19 2 -1.4123174026301097e+01 1.4457033358017881e+00
   9.3877391838968798e+00 1 1 1
20 1 -1.3627832941837832e+01 1.4280871814905454e+00
   9.4779666114892294e+00 1 1 1
21 3 -1.3098311117666332e+01 1.4618828427896244e+00
   9.4616449724919125e+00 1 1 1
22 2 -1.2634793945859228e+01 1.3612887206353281e+00
   9.1824101945402870e+00 1 1 1
23 1 -1.2302659330409373e+01 1.5146374059259602e+00
   8.7947079341876631e+00 1 1 1
24 3 -1.1957342603793172e+01 1.7595578132505101e+00
   8.4944199080240477e+00 1 1 1
25 2 -1.1814274985602566e+01 2.1235117785896600e+00
   8.1191473560257830e+00 1 1 1
26 1 -1.1428532742198975e+01 2.4775357492995385e+00
   7.9569477255929737e+00 1 1 1
27 3 -1.0724930501438699e+01 2.8748948519165478e+00
   8.1962719850998873e+00 1 1 1
28 2 -1.0999389824684833e+01 2.7860708797898330e+00
   8.6633810868557024e+00 1 1 1
29 1 -1.1445393956505441e+01 2.9559357433748006e+00
   8.9188369237334921e+00 1 1 1
30 3 -1.1931134181381466e+01 2.8634304354116762e+00
   9.0461150229645924e+00 1 1 1
31 2 -1.2424069532555515e+01 2.7329648497979337e+00
   8.9916659913554771e+00 1 1 1
32 1 -1.2911606442728047e+01 2.4851766331544245e+00
   9.0105296198522407e+00 1 1 1
```

```
33 3 -1.3120149430040803e+01 2.3160117624532512e+00
   8.6165652041204641e+00 1 1 1
34 2 -1.3401389943380501e+01 1.9403501387306685e+00
   8.4429970264157870e+00 1 1 1
35 1 -1.3541333903179870e+01 1.4845253416384785e+00
   8.3161444630047257e+00 1 1 1
36 3 -1.3624178969754038e+01 9.5346848089913683e-01
   8.4461351397209974e+00 1 1 1
37 2 -1.3861300204215183e+01 5.5823029923429923e-01
   8.7856780960709244e+00 1 1 1
38 1 -1.4191617729596530e+01 2.8298662536234565e-01
   9.1203007142318526e+00 1 1 1
39 3 -1.4650492183416002e+01 2.5109834165865141e-01
   9.4536624308589960e+00 1 1 1
40 2 -1.5107330132535921e+01 3.2116209055168593e-01
   9.6020219253641805e+00 1 1 1
41 1 -1.5658728408361348e+01 3.8402268559994213e-01
   9.5367898713054746e+00 1 1 1
42 3 -1.6065615026568882e+01 3.6482808375122205e-01
   9.3598273440855095e+00 1 1 1
43 2 -1.6506875928574988e+01 -1.1527344808372085e-01
   1.0489705539503952e+01 1 1 1
44 1 -1.6823469645164348e+01 6.5806174397844419e-02
   1.0820133945817672e+01 1 1 1
45 3 -1.7116619884764756e+01 7.5109742107850198e-04
   1.1289077385968934e+01 1 1 1
46 2 -1.7206335763287257e+01 -1.1000137570113229e-01
   1.1786311560839625e+01 1 1 1
47 1 -1.6958969979826648e+01 -9.8963219215490644e-02
   1.2312128930275115e+01 1 1 1
48 3 -1.6749477067819765e+01 4.1290623536025396e-02
   1.2764859633356551e+01 1 1 1
49 2 -1.6456707193463220e+01 4.0130433351110323e-01
   1.3080228515953548e+01 1 1 1
50 1 -1.6366253753561736e+01 8.7753307647555168e-01
   1.3283011275993639e+01 1 1 1
51 3 -1.6399907758537601e+01 1.4122825357655846e+00
   1.3163277692046444e+01 1 1 1
52 46 -1.6570066545357093e+01 1.8455872465475482e+00
   1.3052081661441571e+01 1 1 1
53 51 -1.6918132608079251e+01 2.2998884941860327e+00
   1.3005834368919871e+01 1 1 1
54 56 -1.7334749984335410e+01 2.5864415396024460e+00
   1.3050326216855810e+01 1 1 1
55 59 -1.7717011819602725e+01 2.8622683831095093e+00
   1.3336125062285301e+01 1 1 1
56 61 -1.7973027189528391e+01 3.0902440296390408e+00
   1.3775321020224226e+01 1 1 1
57 4 -1.7990147522205717e+01 3.5011406941981744e+00
   1.4126330967945947e+01 1 1 1
```

```
58 8 -1.7823635863442295e+01 3.8202791326719208e+00
   1.4472939555706603e+01 1 1 1
59 12 -1.7443482933516695e+01 4.0985356852404440e+00
   1.4689346021460274e+01 1 1 1
60 15 -1.7075101459223408e+01 4.5404202262229125e+00
   1.4685931789897973e+01 1 1 1
61 19 -1.6832033140877463e+01 4.9389651206266905e+00
   1.4527930602639945e+01 1 1 1
62 24 -1.6748455723119225e+01 5.4805660109653740e+00
   1.4416308075387581e+01 1 1 1
63 26 -1.6889494593624296e+01 5.9248808239452337e+00
   1.4154850285531481e+01 1 1 1
64 27 -1.7786886242832420e+01 5.8532583666681850e+00
   1.5022362731559998e+01 2 1 1
65 21 -1.7875648111391843e+01 5.3868918035228237e+00
   1.4747757704342330e+01 2 1 1
66 18 -1.7975299475532097e+01 5.1406365484602894e+00
   1.4358240939908828e+01 2 1 1
67 14 -1.7891278360847259e+01 4.9132621743711233e+00
   1.3909193546963269e+01 2 1 1
68 9 -1.7601627374833676e+01 4.5965679969638078e+00
   1.3589637843385034e+01 2 1 1
69 5 -1.7255838657890667e+01 4.2481918203506597e+00
   1.3510236295272859e+01 2 1 1
70 1 -1.6949903895679622e+01 3.7397941817151534e+00
   1.3547825014318773e+01 2 1 1
71 64 -1.6781592314427911e+01 3.3208166214914399e+00
   1.3860920601985576e+01 2 1 1
72 58 -1.6786918029872396e+01 2.7821142041997335e+00
   1.4030749645917744e+01 2 1 1
73 53 -1.7013673757362064e+01 2.2881289499570512e+00
   1.4166908011561597e+01 2 1 1
74 50 -1.7338409513275778e+01 1.8935639160236666e+00
   1.4054788846257493e+01 2 1 1
75 47 -1.7476625401537955e+01 1.4482384069717269e+00
   1.3778971571857992e+01 2 1 1
76 2 -1.7580738230039078e+01 1.1877870016338936e+00
   1.3262190181312494e+01 2 1 1
77 4 -1.7417584312609353e+01 1.0536824698838319e+00
   1.2759876049013455e+01 2 1 1
78 3 -1.7061517885377452e+01 1.0772523304066122e+00
   1.2267594242186625e+01 2 1 1
79 2 -1.6663805213802238e+01 9.2525882604643606e-01
   1.1949607292461769e+01 2 1 1
80 4 -1.6195215810319137e+01 6.4974837023431875e-01
   1.1772325294536811e+01 2 1 1
81 3 -1.6063255201229904e+01 8.9997359738088922e-02
   1.1706149818974255e+01 2 1 1
82 2 -1.6037804962674535e+01 -4.3724510604370304e-01
   1.1568645634887043e+01 2 1 1
```

```
83 4 -1.6262033509132088e+01 -8.8793862335908003e-01
   1.1264647074591689e+01 2 1 1
84 3 -1.6641245985791500e+01 -1.1553533898045254e+00
   1.1066067591213130e+01 2 1 1
85 46 -1.7138712958960770e+01 -1.0345961570550075e+00
   1.0758853619753124e+01 2 1 1
86 43 -1.7418214326212247e+01 -1.0895902336960932e+00
   1.0292609585564271e+01 2 1 1
87 40 -1.7447401529757695e+01 -1.1069745129932456e+00
   9.6819440492947013e+00 2 1 1
88 34 -1.7317199905910694e+01 -1.2777973358734946e+00
   9.1946126400981925e+00 2 1 1
89 32 -1.7210346019228226e+01 -1.6391192096490816e+00
   8.8317459558810807e+00 2 1 1
90 27 -1.6984866780828259e+01 -2.0883590532028800e+00
   8.5309634296992911e+00 2 1 1
91 21 -1.6677432455283242e+01 -2.5627117955895775e+00
   8.5303681042545403e+00 2 1 1
92 18 -1.6620142717540482e+01 -3.0123794033514626e+00
   8.5926041927962675e+00 2 1 1
93 14 -1.6787182814705101e+01 -3.5507285214623514e+00
   8.6602041424192731e+00 2 1 1
94 9 -1.7128588959128013e+01 -3.9016043188205431e+00
   8.7196416372100902e+00 2 1 1
95 6 -1.7495180518333452e+01 -4.2765975564673253e+00
   8.6110888539984920e+00 2 1 1
96 4 -1.7761109435582149e+01 -4.4878358985578481e+00
   8.2849839433616488e+00 2 1 1

# Atom-ID, translational velocity, angular momentum
Velocities

1 2.7716900000000000e-01 -2.4194199999999999e-02 -2.0377500000000001
   e-01 9.7712740076713642e-02 2.1024372496854704e-01
   3.6100075671810589e-02
2 -2.7070499999999997e-01 6.6462800000000000e-01 -1.3143199999999999
   e-01 9.9890317708636822e-02 2.8206022767438782e-01
   3.3409349160824990e-02
3 1.9302200000000000e-01 5.4609200000000002e-01 2.6294400000000001e
   -01 6.5579450419018123e-02 1.4712944885937293e-01
   -1.7516754040111049e-02
4 -1.5697900000000001e-01 6.5866599999999997e-01 3.9529399999999998e
   -01 -2.3843318283043724e-01 1.0450551411616047e-02
   6.7376643262314032e-02
5 -5.3024800000000005e-01 4.1695199999999999e-01 7.7390299999999995e
   -02 -2.7517268110296295e-01 -6.6060582255795641e-02
   -6.2288283258280376e-02
6 -3.7356400000000001e-01 3.5216399999999998e-01 -3.3896200000000001
   e-02 -2.6666324827626459e-01 -3.7631230675194825e-02
   8.3605747310862899e-02
```

```
7 −1.5952500000000000e−01 −4.5836300000000003e−02 1.5551100000000000
   e−02 1.5679690634428997e−01 −2.0009366102825402e−01
   7.8625813380465967e−02
8 −3.3878300000000000e−01 −1.4972600000000000e−01
   −1.9497100000000001e−01 1.0571159161621022e−02
   −1.1603114944445367e−01 −1.6637145694737310e−01
9 −6.5622000000000003e−01 2.8595999999999999e−01 5.2749999999999997e
   −01 −2.1338536580906980e−01 1.1649337999412120e−02
   −2.3103039205676999e−03
10 1.1750500000000000e−01 −1.4621600000000001e−01
   −1.4360200000000001e−01 1.2960293762624933e−01 3.8470561116921403
   e−02 1.6166717574467301e−01
11 −7.0017099999999999e−02 1.5703100000000001e−02
   −9.7916600000000006e−02 4.9120757170027660e−02 1.8987751014002185
   e−01 −5.5660169699651943e−02
12 −9.5634600000000000e−02 1.5747600000000000e−01
   −3.1742800000000002e−02 −1.5075115641526568e−01
   −2.1706834470298787e−02 5.8531411537161973e−02
13 6.8342499999999995e−01 −4.2301100000000003e−01
   −9.5042900000000000e−02 2.4601594105903007e−01 1.6590897814375652
   e−01 −1.1831397639890447e−01
14 −9.5537899999999995e−02 4.4547599999999998e−01
   −2.2455500000000000e−01 −7.4784739933363459e−02
   −6.9472602282879131e−02 −1.1636477269439807e−01
15 5.1701100000000003e−03 −2.0722099999999999e−01
   −7.0585700000000001e−02 −2.1515922965034245e−01
   −8.4882558641622419e−02 −2.6294971183330040e−01
16 2.6272899999999999e−01 −3.7985499999999998e−02
   −2.9943100000000000e−01 −1.7336077614321074e−01
   2.2545748941850130e−03 −1.1497110495511621e−01
17 −1.2444200000000000e−01 5.1474900000000001e−01
   −9.2670600000000006e−02 −5.2704945522267219e−02
   −1.1317629145491481e−01 7.2523506241444755e−02
18 −9.0774499999999994e−02 1.8527299999999999e−01 1.7018400000000000
   e−01 2.7431988586921877e−02 3.1463929499602888e−02
   −5.4960011859656273e−02
19 1.6769600000000001e−01 −1.2929499999999999e−01 3.6735700000000003
   e−02 8.2079572540654358e−02 −1.5462715764193424e−01
   1.3946612590684065e−01
20 2.0230600000000001e−01 2.3384800000000000e−01 −1.5591800000000000
   e−01 3.7651134028848045e−02 −6.7441120561011508e−02
   9.6463378817310277e−02
21 −1.7822399999999999e−01 4.4145499999999999e−01 6.3765000000000002
   e−02 3.6733518471631366e−02 −1.4511615933494044e−02
   −6.3520677447231749e−02
22 −9.7933800000000004e−01 −2.1199999999999999e−01
   −2.8934799999999999e−01 1.8117613338985636e−01 2.3277960197812847
   e−01 2.4332218354866011e−02
23 4.9250100000000002e−01 −1.4289900000000000e−01 2.5012099999999998
   e−01 −2.3329839972124924e−02 1.0594361463694128e−02
   1.1267878881982734e−02
```

```
24 1.4057500000000001e-01 -5.2353200000000000e-01
   -2.9383300000000001e-01 1.6722864621371480e-01
   -9.3170697635283489e-02 -1.3658020677063912e-01
25 -1.1528099999999999e-01 -2.6820200000000000e-01
   6.6398000000000001e-01 -8.2956595024502117e-02 4.8228529358499216
   e-02 2.4083400992523443e-02
26 -3.7268200000000001e-01 -1.4637300000000000e-01
   3.0713299999999999e-01 6.7712150829638815e-02 5.7398458274219591e
   -02 -3.4828601323305744e-02
27 6.4106399999999994e-02 -3.0670699999999999e-02
   -3.0606400000000000e-01 -1.0538114011279862e-01
   8.8358424034728295e-02 2.1755919170592282e-02
28 1.4630000000000001e-01 -7.3746400000000004e-02
   -3.3717900000000001e-01 7.0192865248082078e-03 1.9577135074019478
   e-01 -2.2331055230938604e-01
29 4.8518299999999998e-01 2.8240999999999999e-01 1.4880199999999999e
   -01 1.7772898734886938e-01 1.6410170097779000e-01
   8.1428206318784446e-02
30 -1.1505000000000000e-01 5.2415299999999998e-01
   -1.9082700000000000e-01 1.7012379070457889e-01 3.0894016866844322
   e-02 -1.5360217259418077e-01
31 -2.8403000000000000e-01 -4.2997200000000002e-01
   -2.9864700000000000e-01 7.2048181870177572e-02 2.0907547829757830
   e-01 -3.8775834412222314e-02
32 -4.9581800000000002e-02 -5.0939300000000000e-02
   -3.1978600000000003e-02 -2.0483093887634033e-02
   4.0464228201659125e-02 -5.3094823366573798e-03
33 -4.4076399999999999e-01 -1.0672900000000001e-02
   9.8979499999999998e-02 3.0181969317736035e-01 9.8421719422210327e
   -02 3.9179079041951018e-02
34 -4.4451600000000002e-01 2.4242400000000000e-01 3.4305000000000002
   e-02 5.5885622690865756e-02 -1.5095512701111258e-01
   8.3375794929373206e-02
35 5.3470099999999998e-01 -7.7174900000000005e-02 6.0896200000000000
   e-01 -5.4231949027611966e-03 -1.3798092588175645e-01
   -1.6984665419656941e-01
36 3.2789099999999999e-01 2.9992500000000000e-01 -4.8586400000000002
   e-01 2.3271410396871771e-02 1.2068722568688370e-01
   -3.3891395976003791e-03
37 -1.6086100000000000e-01 3.1535500000000000e-01 5.0912299999999999
   e-01 7.7731943314829841e-02 -1.5164202813265204e-01
   -7.1867078095258843e-02
38 -5.6647899999999995e-01 -4.3262099999999998e-01
   6.0365200000000001e-02 -2.1456180792264110e-01 1.2352750677797958
   e-02 9.4918224369176466e-03
39 -1.3770299999999999e-01 3.6583100000000000e-02
   -1.6641000000000000e-01 2.0455720990574904e-02
   -1.2479546752527684e-02 -1.1705052851275330e-01
40 -1.0684700000000000e-01 9.0120700000000005e-03
   -7.5077000000000005e-01 1.2423432554623678e-01
   -1.5549834063665466e-01 -1.4931093661286526e-01
```

```
41 1.1048700000000000e-02 -1.3551800000000000e-01 3.7819999999999998
   e-01 -7.9131833123996798e-02 3.3786613085688734e-02
   -3.1296321314974365e-03
42 -6.9830400000000004e-01 4.8366100000000001e-01
   -6.5909499999999999e-01 -1.6024989606867668e-01
   7.6528822066567442e-02 -8.4486608929173548e-02
43 -1.6101699999999999e-01 1.7226400000000000e-01
   -3.6498199999999997e-01 -2.6585469291535317e-02
   -1.4965134229302585e-01 4.1853763027715235e-02
44 2.8378799999999998e-01 1.2892799999999999e-01 2.9405199999999998e
   -01 -1.4597555469737214e-01 -1.3194943304329047e-02
   1.1160556182093830e-01
45 6.0323900000000000e-02 -3.4229100000000001e-01
   -1.5871800000000000e-01 5.7748155664119727e-02
   -1.0159540314193732e-01 5.3281912044179501e-02
46 1.3703599999999999e-01 8.7992399999999998e-02 2.4609300000000001e
   -01 1.8840921056555660e-02 -1.0537902680059715e-02
   -7.1085272028855312e-02
47 5.9767800000000004e-01 2.8967699999999999e-02 4.6805400000000003e
   -01 1.9779044900871345e-01 -3.9285300395369163e-02
   1.2904820223616756e-01
48 1.5564000000000000e-01 -3.1912400000000002e-01
   -6.6691100000000003e-02 4.0299799799594936e-02
   -8.7433219120011646e-02 3.5514249964704918e-02
49 9.9530099999999999e-01 6.3581900000000002e-01 9.5887000000000000e
   -02 3.4966751167887715e-02 -6.9871378485775237e-02
   -7.6813791340056239e-02
50 1.8463700000000000e-01 -1.7349000000000001e-01 4.7509999999999997
   e-02 -9.9795575211930762e-02 2.2968334794261772e-01
   1.7121198297868036e-02
51 2.1652099999999999e-01 -5.0061400000000000e-01 1.3691900000000001
   e-01 5.4270076628117427e-02 3.2305227991058130e-02
   1.9274590695184002e-01
52 -6.4402700000000002e-01 -1.8277700000000000e-01
   1.8745899999999999e-01 -1.8520802816581855e-01 2.1564251752945414
   e-02 -5.4498934359754912e-02
53 -1.6870800000000000e-01 -9.2450799999999993e-03
   2.1813800000000000e-01 2.8461232472546877e-01 -1.2205066162573899
   e-01 1.6217419257631878e-02
54 -3.3228600000000003e-01 5.2683300000000000e-01
   -3.6260399999999998e-01 -1.8218311696987642e-02
   1.6099350724885267e-01 -2.6827533887982740e-02
55 -6.7102399999999995e-01 -5.9716000000000002e-01
   -3.3996300000000002e-01 1.8220167333577970e-02 2.1154139532974975
   e-01 9.7817107177321783e-02
56 4.7085700000000001e-02 4.4553100000000001e-01 5.2663300000000003e
   -02 -5.5832778933324320e-02 -7.0151855634465908e-03
   -1.4760180018189742e-01
57 5.7126200000000002e-02 -7.0906300000000005e-02
   -3.4549400000000002e-01 -2.3582838581084963e-02
   1.4028781191276246e-01 -7.7179370631113881e-02
```

```
58 -4.0925200000000000e-01 -1.5370600000000001e-01
   5.7790300000000003e-02 1.1591859572370010e-01 -4.2291060635050934
   e-02 3.0166466375321398e-02
59 -8.7905300000000006e-02 -2.1221200000000001e-01
   8.5137900000000000e-01 3.3644079718665565e-02 -1.2187698766915282
   e-01 -1.4324528342565773e-01
60 7.1609599999999995e-02 -3.3621400000000001e-01
   -1.4027000000000001e-01 6.2552477678577509e-02 1.3251783240366240
   e-02 2.5054400747961986e-01
61 -1.8194099999999999e-01 -3.7339600000000001e-01
   -4.8322900000000002e-01 5.9872376498868500e-02 1.0481433987533692
   e-01 1.4593941610045957e-01
62 -2.8018300000000002e-01 -9.1408299999999998e-02
   2.6663700000000001e-01 -8.7385127983858418e-02 6.1749788706694113
   e-02 7.8491774335240766e-02
63 2.3602100000000001e-01 1.1980700000000000e-01 1.8808900000000001e
   -01 9.5119230632240814e-02 -9.5506218142536632e-03
   5.0392923484410920e-02
64 -1.6956700000000000e-01 -2.6338899999999998e-02
   -4.4328099999999997e-03 2.4488531187987134e-02 5.2189232689496663
   e-02 1.6111380934743694e-02
65 4.9978500000000002e-02 -8.4993399999999997e-02 1.8901399999999999
   e-01 1.0497136640276128e-01 -7.0040752061815625e-03
   1.0061754908365407e-01
66 -3.6576100000000000e-01 -2.4417100000000000e-01
   -1.6466600000000001e-01 -1.2778439895343621e-01
   1.3749027220995582e-02 -1.3013287035242291e-01
67 9.5234799999999994e-02 -1.0075099999999999e-01 6.8076600000000001
   e-02 -1.0545733202885445e-01 -1.2549569433190724e-01
   -8.2326361178752264e-02
68 -4.0818700000000002e-01 -5.1212899999999999e-02
   -1.9381599999999999e-01 -1.6888624924663967e-01
   -2.4456798981332775e-01 -7.7024022468437064e-02
69 -5.6218700000000003e-02 -5.7826500000000003e-02
   2.7286700000000003e-01 4.2247595360228041e-02 2.0983687596947093e
   -01 1.6802057143050289e-01
70 6.3759699999999997e-01 -1.7954000000000001e-01
   -3.1061100000000003e-01 -1.8898170647236667e-01
   2.2998505211401857e-01 -8.1058569403790512e-02
71 -2.5678400000000001e-01 2.3745600000000000e-01
   -2.6933600000000002e-01 9.9939316592774191e-02 2.6458183044254992
   e-02 3.4146945303261734e-03
72 1.0113200000000000e-01 1.0549300000000000e-01 -6.0445899999999997
   e-01 4.7063459544320611e-02 2.9489038328238032e-01
   -1.0959895706728313e-01
73 3.3447600000000000e-01 -3.2610400000000000e-01 8.6393300000000006
   e-02 1.5218196962058594e-01 -3.2723515960581889e-01
   9.8975911579428241e-02
74 8.2678399999999996e-01 1.0337100000000000e-01 2.1208199999999999e
   -01 3.5013856790741781e-02 3.5336794018052994e-02
   1.3818757522139968e-02
```

```
75 2.6382800000000001e-01 4.6635399999999999e-01 3.0230499999999999e
   -01 -7.6394056064948770e-02 -4.7424201596534869e-02
   5.3998580673955757e-02
76 -1.5899199999999999e-01 -2.3860700000000001e-01
   -1.6417799999999999e-01 -2.0742424483117198e-02
   -1.2219722736802144e-01 1.1451136906075286e-01
77 3.2534800000000003e-01 -5.9061699999999995e-01 2.9714400000000002
   e-01 7.9405843478490418e-02 -9.5096240863656895e-02
   -7.2671613070615468e-02
78 -5.8195200000000002e-01 7.9702200000000001e-03
   -5.5859000000000003e-01 -9.4148426168900290e-02
   -3.4168932929153531e-02 4.8848278399735642e-02
79 -6.8417499999999998e-01 -7.3096599999999998e-02
   -3.3468599999999998e-01 -2.5853520732057628e-01
   2.2126245580113149e-01 2.3241031320449071e-01
80 -7.7701299999999995e-01 -1.7017299999999999e-01
   3.3595199999999997e-01 -1.5318644368591728e-02 1.0282729578390429
   e-01 1.2455993329112391e-01
81 -4.9548500000000001e-01 -1.5494800000000000e-01
   1.9013800000000000e-01 1.6031970962737468e-01 -9.7770257827109089
   e-02 -1.6096608410896285e-01
82 2.6535300000000001e-01 1.1779899999999999e-02 -5.4642599999999997
   e-01 1.1700675778002670e-01 1.2482172069752102e-01
   -7.3828582611710020e-02
83 4.2582199999999998e-01 -2.3155300000000001e-01 1.2155299999999999
   e-01 -8.2639003564543442e-02 -7.0213918481943263e-02
   -1.7068642300199569e-01
84 -1.3241100000000000e-01 2.6350200000000001e-01
   -3.3182299999999998e-01 2.7063268737935218e-02 8.7976782501009515
   e-02 -9.2313502955833862e-02
85 -4.7405199999999997e-01 1.1384099999999999e-02
   -2.1505199999999999e-02 -9.1085974834755723e-02
   -1.3750009888830353e-01 1.6985858948381824e-01
86 2.4250400000000000e-01 -3.8162500000000002e-02
   -1.0846699999999999e-01 -4.3986948550113834e-02
   -2.2923169873497282e-02 2.6770804318339447e-01
87 2.7591800000000000e-02 3.6234400000000000e-01 -1.7568500000000001
   e-02 -1.1496442864896764e-01 -7.9640493705210966e-02
   1.3540735425925243e-01
88 -3.0650899999999998e-01 1.5838200000000000e-01
   -1.0223000000000000e-01 -1.4252716886337594e-01
   1.7153506364344148e-01 8.9841840754233071e-02
89 7.8933699999999996e-02 -1.2954199999999999e-01
   -6.4618100000000001e-01 3.2332171448251248e-02 4.9220554837052857
   e-02 -6.6466901115827925e-02
90 1.9717199999999999e-01 -8.3087700000000000e-02
   -3.6247800000000002e-01 -9.1571146206444395e-02
   5.7887182082388598e-02 9.0778637656292946e-02
91 -9.4446500000000003e-02 3.5015900000000000e-01 2.2438900000000001
   e-02 -7.1579751608060391e-02 -1.1992916287126421e-01
   -4.2512374836654061e-03
```

```
92 1.7640700000000001e-01 3.9530599999999999e-01 -4.9729800000000002
   e-01 1.2462711781815039e-01 8.4689915727157172e-02
   4.2455350530501874e-02
93 -8.2763299999999995e-01 -6.7418400000000003e-02
   4.2135800000000001e-01 1.3841951459283722e-01 2.3513349618262562e
   -02 1.2164848445240994e-01
94 -6.5293299999999999e-01 -1.1300600000000000e-01
   4.1076600000000002e-01 -3.6871958177318413e-02 1.3519327241265097
   e-01 -1.5081710983637808e-01
95 9.2887300000000006e-02 -2.7870800000000001e-01 2.0031800000000000
   e-01 4.7888990175781812e-03 -4.5288286208410203e-02
   3.6990462545797596e-02
96 2.1174999999999999e-01 -6.8386100000000005e-01
   -3.7445099999999998e-01 -1.6779275913517128e-01
   1.4879391521205471e-01 3.9187264067839810e-02

# Atom-ID, shape, quaternion
Ellipsoids

1 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 -1.9598710000000000e-01 3.0211739999999998e-01
   6.3263239999999998e-01 6.8563140000000000e-01
2 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 -4.0794439999999998e-01 3.4777550000000002e-01
   5.4831770000000002e-01 6.4185760000000003e-01
3 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 -6.1485509999999999e-01 4.2217230000000000e-01
   4.3983870000000003e-01 5.0026559999999998e-01
4 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 -7.8425940000000005e-01 4.8078759999999998e-01
   2.8219549999999999e-01 2.7229789999999998e-01
5 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 -8.9851639999999999e-01 4.0908460000000002e-01
   1.3101979999999999e-01 9.0287999999999993e-02
6 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 -9.0182649999999998e-01 3.6293189999999997e-01
   -8.7343420000000005e-02 -2.1762470000000000e-01
7 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 -7.9533200000000004e-01 3.5522799999999999e-01
   -2.2074420000000000e-01 -4.3878479999999997e-01
8 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 -7.7887519999999999e-01 1.6137380000000001e-01
   -3.9965270000000003e-01 -4.5562009999999997e-01
9 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 -4.8212539999999998e-01 -1.2406789999999999e-02
   -4.8701470000000002e-01 -7.2816060000000005e-01
10 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 -2.4026990000000001e-01 -1.3142780000000001e-01
   -4.7277789999999997e-01 -8.3754289999999998e-01
11 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 3.9663270000000000e-02 -2.0102619999999999e-01
```

```
    −5.0058979999999997e−01 −8.4108570000000005e−01
12 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
    +00 2.6790370000000002e−01 −3.7151020000000001e−01
    −2.7654479999999998e−01 −8.4482590000000002e−01
13 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
    +00 1.3418780000000000e−01 5.8808260000000001e−01
    6.3511989999999996e−01 4.8246790000000001e−01
14 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
    +00 2.8130440000000000e−02 −6.1879289999999998e−01
    −4.8564350000000001e−01 −6.1680990000000002e−01
15 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
    +00 2.6519630000000000e−01 −7.1139810000000003e−01
    −3.0313319999999999e−01 −5.7592869999999996e−01
16 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
    +00 −3.9844879999999999e−01 7.6142770000000004e−01
    2.9343939999999999e−02 5.1049529999999999e−01
17 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
    +00 −5.8620950000000005e−01 6.7018080000000002e−01
    −1.7835750000000000e−01 4.1881350000000001e−01
18 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
    +00 −7.1490220000000004e−01 5.5351790000000001e−01
    −3.3066530000000000e−01 2.7054240000000002e−01
19 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
    +00 −7.0355069999999997e−01 4.2094169999999997e−01
    −5.7213820000000004e−01 −2.1963980000000001e−02
20 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
    +00 −6.7554899999999996e−01 1.3847419999999999e−01
    −6.9436070000000005e−01 −2.0572219999999999e−01
21 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
    +00 −5.8026639999999996e−01 −2.9715470000000001e−02
    −7.6883020000000002e−01 −2.6703569999999999e−01
22 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
    +00 −4.7755379999999997e−01 −2.5988309999999998e−01
    −7.0121849999999997e−01 −4.6118940000000003e−01
23 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
    +00 −2.9032320000000000e−01 −4.8210789999999998e−01
    −6.1203189999999996e−01 −5.5560900000000002e−01
24 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
    +00 −1.0332550000000000e−01 −4.8510639999999999e−01
    −6.5338209999999997e−01 −5.7191559999999997e−01
25 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
    +00 5.5945120000000003e−03 6.8249499999999996e−01
    3.8225850000000000e−01 6.2293469999999995e−01
26 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
    +00 −2.2760839999999999e−01 6.6419539999999999e−01
    7.8647289999999995e−02 7.0770999999999995e−01
27 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
    +00 3.7458730000000001e−01 −2.2778240000000000e−01
    −5.0247010000000003e−01 7.4520019999999998e−01
28 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
    +00 −9.3994380000000002e−02 3.9434669999999999e−01
```

```
   3.4469300000000003e-01  -8.4666549999999996e-01
29 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 3.3418310000000001e-01 5.0804600000000000e-01
   1.5083810000000000e-01  -7.7939639999999999e-01
30 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00  -4.2138650000000000e-01  -6.2427569999999999e-01
   9.2805559999999995e-02 6.5122999999999998e-01
31 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00  -5.6343790000000005e-01  -6.1919029999999997e-01
   2.5100020000000001e-01 4.8594240000000000e-01
32 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00  -5.1139129999999999e-01  -7.1395189999999997e-01
   2.2382630000000001e-01 4.2267430000000000e-01
33 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00  -6.4323580000000002e-01  -5.3214470000000003e-01
   5.1163230000000004e-01 2.0322910000000000e-01
34 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00  -6.7660039999999999e-01  -3.6989770000000000e-01
   6.3560280000000002e-01  -3.7372790000000003e-02
35 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 6.0683209999999999e-01 2.4309510000000001e-01
   -7.0330680000000001e-01 2.7931899999999998e-01
36 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 4.7237540000000000e-01 4.5822849999999998e-02
   -7.0742490000000002e-01 5.2374770000000004e-01
37 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 3.9344289999999998e-01  -1.6301920000000000e-01
   -6.2370919999999996e-01 6.5544970000000002e-01
38 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 2.1434860000000000e-01  -3.8211020000000001e-01
   -5.0816700000000004e-01 7.4149359999999997e-01
39 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 1.2019230000000000e-01  -4.8349090000000000e-01
   -3.2333240000000002e-01 8.0451629999999996e-01
40 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00  -2.0166529999999999e-01  -6.2021879999999996e-01
   -1.1300000000000000e-01 7.4959370000000003e-01
41 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 4.5347389999999999e-01 6.3879480000000000e-01
   -3.3152639999999997e-02  -6.2064770000000002e-01
42 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 4.3235620000000002e-01 6.9338469999999996e-01
   -3.4556039999999999e-01  -4.6138249999999997e-01
43 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00  -5.5244649999999995e-01 1.7837130000000001e-01
   1.0926880000000000e-01 8.0687480000000000e-01
44 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00  -8.1313290000000005e-01 2.3033470000000000e-01
   7.1005910000000005e-02 5.2982910000000005e-01
45 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00  -9.6572239999999998e-01 1.8700410000000001e-01
```

```
   5.8640499999999998e-02 1.7020879999999999e-01
46 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 9.7972830000000000e-01 -1.3518120000000000e-01
   5.0829180000000002e-02 1.3883390000000001e-01
47 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 8.7819449999999999e-01 -1.8803259999999999e-01
   1.9082360000000001e-01 3.9623799999999998e-01
48 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 -7.1718420000000005e-01 2.3088259999999999e-01
   -2.9545480000000002e-01 -5.8740669999999995e-01
49 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 -3.9302799999999999e-01 1.5045130000000001e-01
   -3.8011610000000001e-01 -8.2365350000000004e-01
50 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 2.2007510000000000e-01 -1.3310630000000001e-01
   5.2417689999999995e-01 8.1184250000000002e-01
51 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 -6.5807130000000005e-02 -1.4302620000000000e-02
   5.7397600000000004e-01 8.1609830000000005e-01
52 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 -2.7328350000000001e-01 1.0536050000000000e-01
   6.6367169999999998e-01 6.8829879999999999e-01
53 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 -4.7928589999999999e-01 1.7245120000000000e-01
   6.1341780000000001e-01 6.0354310000000000e-01
54 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 -6.7976099999999995e-01 4.6572110000000000e-01
   4.0413100000000002e-01 3.9712330000000001e-01
55 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 -7.8905809999999998e-01 5.1038059999999996e-01
   2.8389660000000000e-01 1.9053010000000001e-01
56 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 -8.3578609999999998e-01 5.2873159999999997e-01
   1.1436040000000000e-01 -9.3947420000000004e-02
57 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 8.4339790000000003e-01 -4.6204260000000003e-01
   1.2027820000000000e-01 2.4643399999999999e-01
58 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 -7.6484660000000004e-01 3.8774039999999999e-01
   -3.1997579999999998e-01 -4.0284300000000001e-01
59 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 -5.5620610000000004e-01 2.2338160000000001e-01
   -5.2387689999999998e-01 -6.0521760000000002e-01
60 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 2.9862889999999997e-01 -1.8822230000000001e-01
   6.0599700000000001e-01 7.1285390000000004e-01
61 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 9.0816209999999994e-02 5.0644570000000000e-02
   6.4146680000000000e-01 7.6007100000000005e-01
62 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 -1.1448240000000000e-01 7.7040680000000000e-02
```

```
   6.7921339999999997e-01 7.2085200000000005e-01
63 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 -3.1463180000000002e-01 3.1009789999999998e-01
   5.8539509999999995e-01 6.7982260000000005e-01
64 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 -5.6478399999999995e-01 -7.2332830000000004e-01
   -2.7354990000000001e-01 2.8807230000000000e-01
65 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 -5.5684940000000005e-01 -8.0736810000000003e-01
   -1.2619469999999999e-01 1.4883009999999999e-01
66 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 5.1456199999999996e-01 8.4440910000000002e-01
   -1.4164969999999999e-01 4.6200919999999999e-02
67 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 4.8097060000000003e-01 7.7620069999999997e-01
   -3.6181489999999999e-01 1.8780230000000001e-01
68 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 4.0935420000000000e-01 5.9263359999999998e-01
   -5.9070809999999996e-01 3.6370110000000000e-01
69 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 2.4009910000000001e-01 4.3352910000000000e-01
   -7.0993609999999996e-01 5.0039549999999999e-01
70 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 1.6627200000000000e-01 2.4624130000000000e-01
   -7.2235400000000005e-01 6.2443850000000001e-01
71 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 -3.1668929999999998e-02 5.8351859999999998e-02
   -7.6915180000000005e-01 6.3560819999999996e-01
72 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 -1.8732670000000001e-01 -2.0840620000000001e-01
   -7.4265919999999996e-01 6.0822109999999996e-01
73 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 -3.5768889999999998e-01 -4.2100830000000000e-01
   -6.8092459999999999e-01 4.8078310000000002e-01
74 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 -4.6608250000000001e-01 -6.5323249999999999e-01
   -4.9566090000000002e-01 3.3222679999999999e-01
75 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 -3.9016489999999998e-01 -8.2260480000000002e-01
   -3.4446420000000000e-01 2.2899140000000001e-01
76 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 -3.5930980000000001e-01 -9.2233750000000003e-01
   -1.4192140000000000e-01 6.9531970000000004e-03
77 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 3.1367640000000002e-01 9.1642179999999995e-01
   -1.2685060000000001e-01 2.1374540000000000e-01
78 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 2.5915959999999999e-01 8.0336149999999995e-01
   -4.5772610000000002e-01 2.7916540000000001e-01
79 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 9.7604839999999998e-02 7.2319690000000003e-01
```

```
   −5.8856620000000004e−01 3.4792149999999999e−01
80 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 1.4891689999999999e−01 4.2226989999999998e−01
   −8.6876209999999998e−01 2.1157580000000001e−01
81 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 5.3645680000000001e−03 7.0991269999999995e−02
   −9.2672849999999996e−01 3.6892520000000001e−01
82 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 −8.0655210000000005e−02 −2.5663560000000002e−01
   −9.3322600000000000e−01 2.3816430000000000e−01
83 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 −4.0490920000000000e−02 −4.7408650000000002e−01
   −8.0883309999999997e−01 3.4553080000000003e−01
84 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 −1.9697670000000000e−02 −7.1217830000000004e−01
   −5.9347019999999995e−01 3.7444240000000001e−01
85 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 −1.7704860000000000e−01 −8.7430249999999998e−01
   −2.6797300000000002e−01 3.6392239999999998e−01
86 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 −2.6807449999999999e−01 −9.4829110000000005e−01
   6.5601450000000006e−02 1.5676909999999999e−01
87 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 −3.2911170000000001e−01 −9.1510840000000004e−01
   2.2124450000000001e−01 7.2889800000000005e−02
88 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 3.5379470000000002e−01 7.2474139999999998e−01
   −5.8367150000000001e−01 9.4375169999999994e−02
89 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 3.2261459999999997e−01 6.1335689999999998e−01
   −6.6672500000000001e−01 2.7420970000000000e−01
90 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 2.5060389999999999e−01 4.5975440000000001e−01
   −7.1887869999999998e−01 4.5720549999999999e−01
91 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 1.0454610000000000e−01 1.6200800000000001e−01
   −8.2825009999999999e−01 5.2614200000000000e−01
92 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 −5.0237629999999998e−02 −1.0271880000000000e−01
   −7.3795820000000001e−01 6.6508849999999997e−01
93 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 −1.7405180000000001e−01 −3.9286240000000000e−01
   −7.1740119999999996e−01 5.4836169999999995e−01
94 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 −3.3430169999999998e−01 −6.1729239999999996e−01
   −5.3822159999999997e−01 4.6637960000000001e−01
95 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 −3.2724039999999999e−01 −7.7407309999999996e−01
   −3.9365240000000001e−01 3.7250820000000001e−01
96 1.1739845031423410e+00 1.1739845031423410e+00 1.1739845031423410e
   +00 5.2509269999999997e−01 8.2774040000000004e−01
```

```
    7.3003910000000005e-02  -1.8383150000000001e-01

# Bond topology
Bonds

1   1   25   26
2   1   23   24
3   1   38   39
4   1   27   28
5   1   19   20
6   1   16   17
7   1   93   94
8   1   35   36
9   1   1   2
10   1   2   3
11   1   29   30
12   1   5   6
13   1   91   92
14   1   24   25
15   1   37   38
16   1   4   5
17   1   21   22
18   1   13   14
19   1   14   15
20   1   36   37
21   1   15   16
22   1   89   90
23   1   92   93
24   1   3   4
25   1   26   27
26   1   90   91
27   1   31   32
28   1   40   41
29   1   6   7
30   1   7   8
31   1   8   9
32   1   28   29
33   1   94   95
34   1   34   35
35   1   18   19
36   1   20   21
37   1   32   33
38   1   30   31
39   1   33   34
40   1   17   18
41   1   95   96
42   1   22   23
43   1   88   89
44   1   39   40
45   1   41   42
46   1   42   43
```

```
47   1   87   88
48   1   80   81
49   1   65   66
50   1   56   57
51   1   62   63
52   1   52   53
53   1   83   84
54   1   74   75
55   1   55   56
56   1   57   58
57   1   58   59
58   1   48   49
59   1   67   68
60   1   61   62
61   1   86   87
62   1   66   67
63   1   72   73
64   1   49   50
65   1   68   69
66   1   54   55
67   1   73   74
68   1   69   70
69   1   60   61
70   1   75   76
71   1   85   86
72   1   46   47
73   1   51   52
74   1   76   77
75   1   71   72
76   1   84   85
77   1   78   79
78   1   44   45
79   1   53   54
80   1   77   78
81   1   50   51
82   1   59   60
83   1   64   65
84   1   82   83
85   1   70   71
86   1   47   48
87   1   79   80
88   1   45   46
89   1   81   82
90   1   43   44
91   1   11   12
92   1   12   13
93   1   10   11
94   1   9   10
```