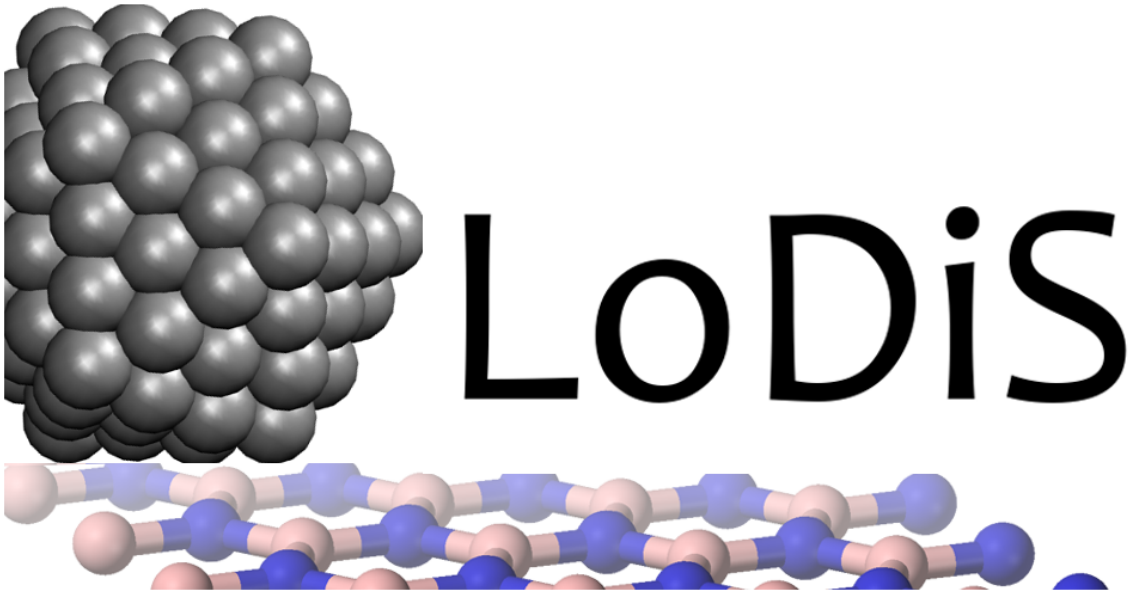

LoDiS MD manual

Kevin Rossi, Raphael Pinto-Miles, Francesca Baletto*
Physics Department, King's College London, WC2R 2LS, UK

Contents

*Electronic address: francesca.baletto@kcl.ac.uk

Disclaimer

Introduction and set-up

Low Dimensional System molecular dynamics (LoDiS) software is a FORTRAN90 computational engine specifically tailored to the study of processes at the nanoscale in finite-size systems by means of Molecular Dynamics. The program primarily focuses on investigations into growth, coalescence, phase transition and free energy sampling of mono- and bi-metallic nanoclusters. Implementations to probe carbon systems and noble gases have also been developed, although less testing has been done for such systems. Extensions to the program allow the choice between a ligand or a vacuum environment with/without a MgO substrate present in an molecular dynamics run. First born during a collaboration between groups in Marseille (Treglia, Mottet) and Genoa (Ferrando, Rossi), it has been fully developed in King's College London with the collaboration of few undergraduate and master students (Ellaby, Soon, Parsina, Bazeley, Khan, Pinto-Miles) by Pavan, Rossi and Baletto.

Installing the program simply requires compiling all the .f90 files via the makefile by using the following command on terminal:

$$make \quad -f \quad Makefile$$

Before running the command, modify the makefile to use the version of fortran and it's libraries present on your computer. Several options are already available within the makefile and can be chosen by removing the # symbol. During a successful compilation, the makefile will produce multiple warnings of dummy/unused variables and equality comparisons which

can be ignored.

This manual is intended to offer an overview of the capabilities of this software from a pedagogical point of view, it contains both examples on how to use the program and in depth explanation of the physical processes that can be simulated *in silico* with this research tool. The first part of the manual is dedicated to the describe the theoretical grounds on which LoDiS's software is built upon (Sections I, II, III and IV). Section I discusses the algorithms used in the main Molecular Dynamics loop. The second section deals with the potential formulation used to model interactions between the components of the analysed system. Procedures that do not involve physical transformation of the clusters are discussed in Section III. Lastly, Section IV is dedicated to the physical processes available to LoDiS. The second half of the manual is a tutorial on preparing and running LoDiS as well as paradigmatic case studies discussing the results of each process (Section V).

I. ALGORITHMS

A. Molecular Dynamics background

Molecular Dynamics is a computational algorithm which integrates Newton's equation of motion according to a finite difference method to simulate the time evolution of a system from specified initial boundary conditions. Lyapunov instabilities determines the proceedings of a dynamic given certain initial condition, such that statistics should be gathered from a large ensemble of trajectories and episodes to maximise exploration of the energy landscape. A flow diagram of the main code is reported in Figure 1

B. Velocity-Verlet Algorithm

A Velocity-Verlet algorithm(Verlet, 1967) is exploited to evolve Newton's Equation of motion. It is an energy-conserving, time reversible, order four error on position algorithm where positions and velocities are calculated throughout a Taylor expansion of both as shown in equation 1.

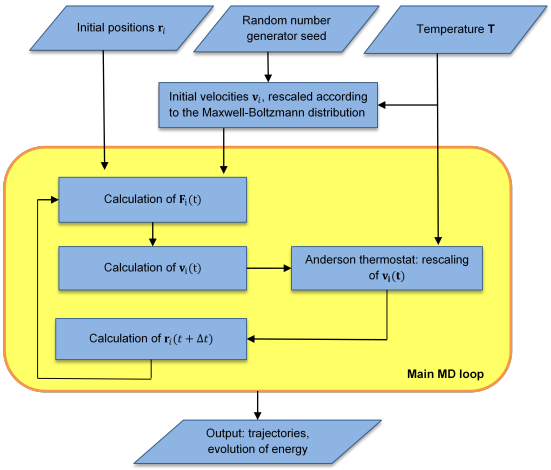

FIG. 1 Flow chart of algorithm calls to run a Molecular Dynamics in LoDiS. The system is initialised with the input positions, velocities and temperature, undergoing a set number of loops highlighted by the yellow box before ending the program. Outputs are written into external files every n incremental loops, n being a real number chosen by the user.

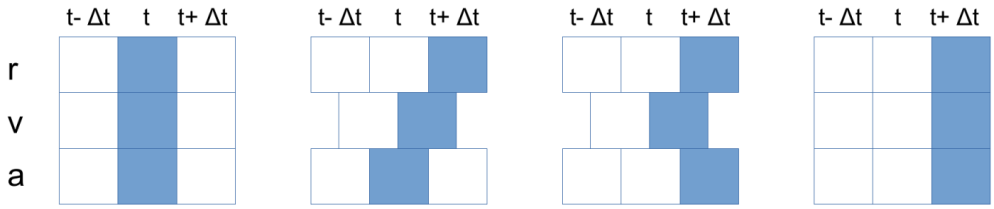

FIG. 2 Schematic of position (r), velocity (v) and acceleration (a) updates according to a Velocity Verlet algorithm.

$$
\begin{aligned}
r(t+\delta t) &= r(t) + v(t)\delta t + \frac{f(t)}{2m}\delta t^2 + \mathcal{O}(\delta t^3) \\
v(t+\delta t) &= v(t) + \frac{f(t)+f(t+\delta t)}{2m}\delta t^2 + \mathcal{O}(\delta t^3)\ .
\end{aligned}
\tag{1}
$$

such that the calculation of position, velocities and forces proceeds following a two step procedure: first velocities at half time step are evaluated by integration of forces given positions at time t, then positions are updated at full step using the just calculated velocities at half-step. A schematic of the algorithm flow is also reported in Figure 2

C. Andersen Thermostat

An Andersen thermostat(Andersen, 1980) is used to mimic the interaction of the system with a stochastic bath that allows the temperature of the system to fluctuate around a selected temperature through virtual instantaneous interactions. It replaces the momentum of randomly selected atoms with a new one drawn from a Maxwell-Boltzmann distribution at the chosen temperature. In the elapsed time from one stochastic collisions to the next one, the system evolves at constant energy obeying Newton's equations of motion. The distribution of the durations of the elapsed times between two interactions follows a Poisson process. The strength of the coupling to the bath is determined by the thermostat frequency *vnu* which is chosen by considering two opposite needs: it has to be high enough in order to have an efficient thermostat and not too large in order to not influence the diffusive properties of the particle. A good compromise between the two conditions is obtained by setting *vnu* as 5.d11. An example of how the algorithm is implemented in LoDiS is hereby reported in Figure 3.

```
DO i=1,nat3
qx(i)=(vx(i)+t2m(itype(i))*(fx(i)+dfx(i))) !velocity x component following verlet
qy(i)=(vy(i)+t2m(itype(i))*(fy(i)+dfy(i))) !velocity y component following verlet
qz(i)=(vz(i)+t2m(itype(i))*(fz(i)+dfz(i))) !velocity z component following verlet
!Andersen Thermostat with thermalization of atoms with frequency vdfa
vdfae=dlaran(irand_seed) ! random number to check whether thermostat acts
IF (vdfae > vdfa) THEN
vx(i)=qx(i)!updated velocity x component
vy(i)=qy(i)!updated velocity y component
vz(i)=qz(i)!updated velocity z component
ELSE
!variance of the gaussian dependent on temperature of the system (tfin)
vargau=SQRT(2.d0*t2m(itype(i))*cbol*tfin)
CALL gauss      !extracts values g1, g2, g3 from a gaussian distribution
vx(i)=g1*vargau  !updated velocity x component
vy(i)=g2*vargau  !updated velocity y component
vz(i)=g3*vargau  !updated velocity z component
ENDIF
END DO
```

FIG. 3 Andersen Thermostat algorithm as implemented in LoDiS. The algorithm updates the velocity components according to the force underpinning Newton's equations of motion if vdfae > vdfa is '.TRUE.'. Otherwise the new set of velocities are randomly selected from Gaussians centred on the temperature. The coupling strength variable vdfa is the product of *vnu* and the Verlet time step *tstep*.

D. Coordination number formalism

Interactions approximated by two-body contributions are strongly tied to pair distances (pdf) and consequently the coordination number of components (CN_i). As such, the choice of either a discrete or continuous CN_i as well as the method of calculation is an important factor to consider. To prevent the chance of discontinuities which can potentially halt the program, two continuous formalisms for the CN_i were chosen, the first being the Fermi formalism:

$$CN_i \;=\; \Sigma_{j\neq i}\frac{1}{1 \;+\; e^{m\frac{r_{ij}}{r_c}-1}} \tag{2}$$

where the cut-off radius, rc, and the exponent, m, tune the smoothness of the Fermi distribution function (Atanasov *et al.*, 2013). The second, more widely used formalism within the code is the polynomial equation, which is better equipped to handle heterogeneous interactions (bimetallic clusters, ligands and substrate). The Heterogeneous CN and Homogeneous CN follow the same rule, the difference being the parameters chosen and consequently the algorithm behind choosing said parameters:

$$HeCN = \Sigma_{i,j\;\; j\neq i}\; f(r_{ij}) \tag{3}$$

$$f(r_{ij}) = \begin{cases} 1 & \text{if } r_{ij} \;\leq\; \rho_0 \\ \frac{1-\left(\frac{r_{ij}-\rho_0}{q_0}\right)^n}{1-\left(\frac{r_{ij}-\rho_0}{q_0}\right)^m} & \text{if } r_{ij} \;>\; \rho_0 \end{cases} \tag{4}$$

The parameter ρ_0 is the nearest neighbour bulk reference distance whilst q_0 is the width of the descending branch of the sigmoid function f(r_{ij}) The shape of the function is tuned by n and m. The HoCN ρ_0 should mimic bulk distances between atoms in mono-metallic lattice whereas HeCN will have three ρ_0 to choose from depending on the pair observed.

II. INTERATOMIC POTENTIALS

Investigation of physical systems by means of $in-silico$ experiment are constrained by a trade-off between accuracy and computational expanses. Semi-empirical potentials cast within the second moment tight binding approximation established as reliable and efficient in the studies of transition metal nanosystems with size between tens and thousands of atoms. The implemented potential formalism according to Rosato-Guillope-Legrande(Rosato *et al.*, 1989) formulation is detailed in the following subsection. Further available Lennard-Jones

and Girifalco potentials are available to simulate noble gas clusters and carbon based systems respectively.

A. Gupta (Rosato-Guillope-Legrand) Potential

Metal-Metal interaction is modelled according to the Gupta or Rosato-Guillope-Legrande formulation(Rosato *et al.*, 1989). This potential consists of a repulsive and an attractive term. The first is well approximated by a Born-Meyer term, the latter by a many-body term derived according to the second moment tight binding approximation framework. Considering the local density of states of electrons centered at a specific site and on a certain orbital, the Hamiltonian of the system is given by an atomic level orbital contribution and by an hopping integral. Approximating the local density of states to a rectangular shape, according to a second moment approximation, the attractive term can be found. The aforementioned approximation is well suited for transition metals given the strong d character of their valence states.

The interaction potential, $V_{TBSMA}(r_{ij})$, depends on the set of all atomic coordinates r_{ij} and is the sum of atomic contributions E^i_{TBSMA}, evaluated as in Equation 5. The sum runs up to the number of atoms n_v within an appropriate cut-off distance from atom i, a and b refer to the chemical species of the two atoms, r^0_{ab} is the bulk nearest neighbour distance for the homometallic atomic pairs and their arithmetical average for the bimetallic case. A_{ab} and ξ_{ab} are related to the cohesive energy, while p_{ab} and q_{ab} and their product tune the stickyness of the potential. Parametrization in the case of homo-atomic interaction are reported in table I

$$E^i_{TBSMA} = \sum_{j\neq i}^{n_v} A_{ab} e^{-p_{ab}\left(\frac{r_{ij}}{r^0_{ab}}-1\right)} - \sqrt{\sum_{j\neq i}^{n_v} \xi^2_{ab} e^{-2q_{ab}\left(\frac{r_{ij}}{r^0_{ab}}-1\right)}} \,. \tag{5}$$

M-M	A (eV)	ξ (eV)	p	q	E_{coh} (eV)
Ni-Ni	0.096	1.56	11.34	2.27	4.46
Pd-Pd	0.175	1.71	10.87	4.00	3.89
Pt-Pt	0.274	2.62	10.71	3.84	5.86
Cu-Cu	0.089	1.28	10.55	2.43	3.50
Ag-Ag	0.103	1.19	10.85	3.18	2.95
Au-Au	0.206	1.79	10.23	4.04	3.94

TABLE I Resume' of parametrization of the RGL potential and cohesive energy, E_{coh}, for homoatomic metal-metal interaction (Cleri and Rosato, 1993; Baletto *et al.*, 2002).

B. Lennard-Jones potential

Lennard-Jones (Jones and Chapman, 1924) potential is also available as a simple and elegant formulation to model interaction between the components of the investigated system: it is made of an attractive interaction and a short range repulsive interaction:

$$E_{ij} = 4\epsilon \left(\frac{\sigma}{r}^{12} - \frac{\sigma}{r}^{6} \right) \tag{6}$$

where ϵ and σ describe respectively the depth of the energy minima and its position.

C. C60 Pacheco-Girifalco Potential

The interaction between C_{60} molecules can be modelled by the Girifalco potential.(Girifalco, 1991) It is a pair potential obtained assuming a spherical shape for the molecule and an uniform distribution of Lennard-Jones centers on its surface. The potential energy of interaction between carbon atoms at distance r_{ij} can be described by

$$E_{ij} = -\frac{A}{r_{ij}} + \frac{B}{r_{ij}^{12}} \tag{7}$$

where A and B are constants.

D. Implicit environment

The energy landscape of a nanoparticle system is known to differ when subjected to a vacuum and an inert environment as surface atoms form bonds with ligand atoms. To maximise the potential investigative powers of LoDiS, the option to choose the presence of a tunable environment is available. Metal-environment interactions are obtained as the collective of atomic contributions dependent on the coordination number (CN_i) using Huerto-Cortes, Goniakowski, Noguera formalism (Cortes-Huerto *et al.*, 2013)

$$E_i{}^{M-E} \;=\; -\,\epsilon * {CN_{open}}^{\rho} \tag{8}$$

The difference between the bulk coordination number (CN_{bulk}) and CN_i is given as CN_{open} which is analogous to the number of absent bonds with respect to bulk. For fcc metals, $CN_{bulk} = 12$. The nature of the interaction is determined by ρ:

- $\rho = 1$, pairwise interaction
- $\rho\ < 1$, covalent bonding
- $\rho\ >$, strongly interacting environments

The ϵ parameter fixes the interaction strength. Different ρ and ϵ parameters set the ratio between the surface energies of low Miller index terminations, thus introducing a tunable parameter to favour an architecture or another.

E. Implicit metal-substrate interaction potential

Metal nanoparticle stability is significantly impacted by the presence of a support and the geometrical interplay between the two. Consequently, LoDiS is installed with an Mg0 implicit force field based on double Morse potentials dependent on the underling symmetries in the substrate. This approach captures preferential binding to the inequivalent sites at low computational expenses whilst allowing for the possible thermal corrugation of the substrate to be neglected. The interaction energy E^{M_S} using Vervisch-Mottet-Goniakowski formulation (Vervisch *et al.*, 2002), is a function of the distance between the cluster atom i and the substrate,z_i.

$$E_i{}^{M-S}(x,y,z) \;=\; a_1(x,y,z)e^{-2a_2(x,y,z)(z-a_3(x,y,z))} \;-2e^{a_2(x,y,z)(z-a_3(x,y,z))} \tag{9}$$

Coefficients $a_{i,\alpha}$ encode information on the CN_i (≤ 12) and the substrate geometry with respect to the relative coordinates x_i and y_i between the two.

$$a_i(x,y,z) \;=\; b_{i1}(x,y) \;+\; b_{i2}(x,y)e^{-\frac{CN_i}{b_{i,3}(x,y)}} \tag{10}$$

The b_{ij} parameters similarly each depend on 3 parameters c_{ijk}, weighted by the parameter a and trigonometric functions that determine periodicity and symmetry. In total, the metal-metal oxide interactions require fitting 27 parameters (with the addition of r_c and m for the CN_i) which tune the strength, shape and symmetries of the force field. These parameters are input via a .pot file specific for the metal-MgO potential. There is currently only one available geometry for the MgO support, the **double square**:

$$b_{ij}(x,y) = c_{1ij} \;+\; c_{2ij}\left(cos(\frac{2\pi x}{a}) + cos(\frac{2\pi y}{a})\right) \;+\; c_{3ij}\left(cos(\frac{2\pi(x+y)}{a}) + cos(\frac{2\pi(xy)}{a})\right) \tag{11}$$

III. PROCEDURES

Within the LoDiS package several protocols are available: simulated annealing by quenching procedure, canonical ensemble molecular dynamics, gas phase and nanocluster growth via atom by atom deposition, nanocluster melting and freezing processes, Metadynamics conformational space enhanced sampling. Each of this process will be described in a dedicated section. A flow diagram of the full LoDis capabilities is reported in Figure 4

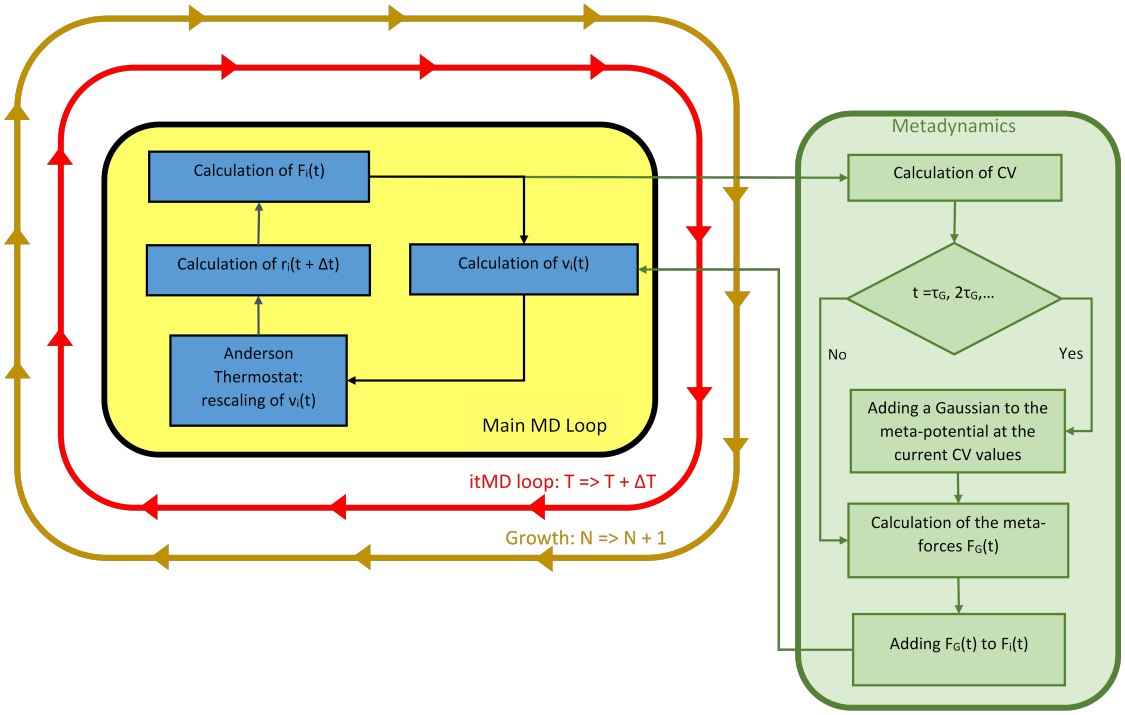

FIG. 4 Flow chart of LoDiS algorithm loops for various MD procedures. The main MD loop describes NVE, NVT, coalescence and quenching procedures which only differ in the method of scaling the Anderson thermostat and calculations of $v_i(t)$. Both growth and itMD add an overarching loop on the NVT MD algorithm in regards to atom number and temperature respectively. There is a divergence following the calculation of $F_i(t)$, following a separate algorithm path before converging back into the main loop for metadynamics.

A. Quenching

Quenching procedures simulate the relaxation of the system towards the closest local minimum energy configuration on the Potential Energy Surface (PES). This computationally efficient process is achieved by rapidly decreasing the temperature associated with the Anderson Thermostat whilst reducing atomic velocity components opposite to the force components to zero. The accelerated annealing ensures other basins on the PES are unlikely to be explored given the time restraints allowing for a near identical output structure with negligible kinetic energy.

B. Canonical NVT

The dynamical behaviour of a nanoparticle can be simulated using the canonical Molecular Dynamics procedure where all the atoms are free to move accordingly to the interatomic potential and temperature is calculated according to the equipartion theorem and tuned by the Andersen Thermostat. In essence, heat is exchanged between the system and the bath by stochastic interactions, retaining the temperature of the system but not the total energy.

C. Microcanonical NVE

An isolated system uncoupled to a heat bath will evolve solely under the interatomic potentials without any heat exchange. This is an adiabatic process where the total energy of the system is conserved and the dynamics of the system is the consequence of energy transfer carried out via work.

IV. PHYSICAL PROCESSES

A. Iterative canonical Molecular Dynamics (itMD)

Freezing/Melting are concatenated canonical runs where the temperature of the system is lowered/increased by a factor ΔT every $\Delta\tau$ time. Between temperature increments, the dynamics of the system advances under canonical NVT. This allows to get caloric curves, determine freezing and melting points and to investigate solid-liquid and liquid-solid transitions kinetics. The ratio $\Delta T/\Delta\tau$ tunes the cooling/heating parameter which impacts the kinetics of the solid-liquid and liquid-solid transitions.

B. Growth

Formation process of a metallic nanoparticle can be modelled via atom by atom growth and freezing simulations.(Baletto *et al.*, 2000) Atom by atom deposition happens on an initial metallic nucleus, thermalization and relaxation of the cluster are allowed between two depositions, implementation for supported nanoclusters has been conducted. This is a simulation where the temperature is fixed and on the original NP cluster, n atoms, one at the time, are deposited every τ steps and allowed to thermalize for $\tau 2$ steps.

In case of a gas phase growth, no preferential direction for atom deposition is present. Atoms are shot towards the center of mass from a sphere 3 times larger of the NP with a velocity extracted from a Boltzmann Distribution coherent with the temperature of the source. Depending on the chosen parameters, atoms of a single chemical species or of two different chemical species may be deposited on the initial cluster.

Comparative energetic analysis of structural stability between the initial and final structures can be carried out with the excess energy ΔE. It is defined as the energy in excess with respect to having N atoms in the bulk, weighted by the number of atoms in the surface (Baletto, 2019):

$$\Delta E = \frac{N E_{coh} - E_{pot}}{N^{2/3}} \tag{12}$$

where N is the number of atoms in the cluster, E_{coh} the cohesive energy of the atoms in the bulk and E_{pot} the potential energy of the relaxed nanocluster.

C. Coalescence

Two clusters with identical components can be built at an appropriate separation from one another to enable minimal interaction. Following thermalization, one of the clusters can be translated towards the other with kinetic energy comparable to the temperature of the metal vapour. Once in contact, the clusters are left to evolve and aggregate under the potentials with a temperature lower than the vapour tuned by the Anderson thermostat. The impact velocity plays a significant role in the formation process of the final aggregate. Whereas growth involves a step by step increase in total energy and atomic number allowing for thermalization between depositions, coalescence is an immediate process with an aggregation time shorter than the time scale to equilibrate (Baletto, 2019). Clusters with N atoms borne of coalescence will therefore have a larger ΔE compared to clusters with the same N components that are simply grown.

D. Metadynamics

Metadynamics is an enhanced sampling algorithm coupled to Molecular Dynamics to simulate rare events.(Laio and Parrinello, 2002) It coarse grains the dynamics of the system in an order parameter space, also known as collective variables space, where a history dependent

potential is added to the Hamiltonian of the system. According to classical Metadynamics formalism, the biasing potential evolves in time as the sum of Gaussians iteratively added every t_G which fill local minima. Minimum-minimum transition is therefore encouraged whilst preventing retreading previous configurations allowing for wide exploration of the energy landscape.

$$\Delta V\left(CV(r_{ij,t}),t\right) = \sum_{t'=t_G,2t_G..} \omega e^{-\frac{\left[CV(r_{ij},(t))-CV(r_{ij}(t'))\right]^2}{2\sigma^2}} , \tag{13}$$

The height and width of the Gaussians are given as ω and σ respectively. $CV(r_{ij})$ represents the set of collective variables chosen and defines the order parameter space where the Meta potential evolves, $CV(r_{ij}(t'))$ is the value of the collective variable at time t'.

After a transient, the Metadynamics potential provides in principal an unbiased estimate of the underlying free energy, F:

$$\Delta V(CV, t \to \infty) = -F + constant \ . \tag{14}$$

To carry out accurate investigations into the energy landscape, appropriate CVs with the ability to describe a breadth of nanoparticles without characteristic atoms are required (Tribello *et al.*, 2011). For this software, they are based around the pair distribution function and common neighbour analysis as geometrical descriptors to identify different nanoclusters due to it's cheap computational cost. A metadynamics run is able to make use of either one or two sets of Collective Variables with LoDiS. Due to common ground shared by most of the CVs, similar notation is used to allow for ease of input when referring to this manual.

Coordination number (CN)

The sum of the nearest neighbour numbers for all the atoms in the cluster. The CN is calculated via an analytical sigmoid function f :

$$S = CN = \Sigma_{i,ji\neq j} f(r_{ij}) \tag{15}$$

$$f(r_{ij}) = \begin{cases} 1 & \text{if} \quad r_{ij} \leq d_0 \\ \frac{1-\left(\frac{r_ij-d_0}{r_0}\right)^n}{1-\left(\frac{r_ij-d_0}{r_0}\right)^m} & \text{if} \quad r_{ij} \geq d_0 \end{cases} \tag{16}$$

The smoothness and shape of the function is determined by parameters m and n, whereas d_0 characterises the bulk reference nearest neighbour distance and r_0 is the width of the sigmoid function's descending branch. Both d_0 and r_0 are given as ratios of the bulk lattice parameter for all sets of CV.

Additional calculations and checks allow for Heterogeneous CN (HeCN)/Homogenous CN (HoCN) analysis for bi-metallic clusters with LoDiS, only requiring prior knowledge of the presence of mixed and/or homogeneous atomic pairs for the chemical species.

Stacking fault number (SFN) and d3.4N

A function that counts the number of atomic pair distances that equate to 1.354 times the bulk reference distance. This is a marker for a local hcp structure characterised by a ABA stacking. The SFN is given by an analytical window function similar to the CN with pair distances r_{ij} and the characteristic distance $d_0 = 1.354$:

$$f(r_i j) = \frac{1 - \left(\frac{r_{ij} - d_0}{r_0}\right)^n}{1 - \left(\frac{r_{ij} - d_0}{r_0}\right)^m} \tag{17}$$

The parameters work in very much the same fashion as those for the coordination number.

Another set of CVs called d3.4N or C2N makes use of the above window function centred around the characteristic distance ratio $d_0 = 3.4$. It is based on the broadening of characteristic pdf peaks experienced by clusters at high temperatures.

Common neighbour number (CNN)

Be aware that this algorithm has not been fully tested and is likely to return errors or even halt the procedure if chosen!!!

The common neighbour number counts the number of atomic pairs sharing the number of pairs with common neighbours, albeit in a continuous approach. This is achieved by applying a window function λ (CNN win) centred around the number of common neighbours

considered to the discrete common neighbour analysis function l_{ij}

$$\lambda(l_{ij}) = \begin{cases} (h_\lambda - 1)\left(\frac{l_{ij}-d_\lambda}{r_\lambda}\right)^{m_\lambda} + 1 & \text{if} \quad d_\lambda \leq l_{ij} \leq d_\lambda + r_\lambda \\ (h_\lambda - 1)\left(\frac{d_\lambda-l_{ij}}{r_\lambda}\right)^{m_\lambda} + 1 & \text{if} \quad d_\lambda - r_\lambda \leq l_{ij} \leq d_\lambda \\ h_\lambda \left(\frac{l_{ij}-d_\lambda-c_\lambda}{r_\lambda-c_\lambda}\right)^{n}_{\lambda} & \text{if} \quad d_\lambda + r_\lambda < l_{ij} < d_\lambda + c_\lambda \\ h_\lambda \left(\frac{d_\lambda-l_{ij}-c_\lambda}{r_\lambda-c_\lambda}\right)^{n}_{\lambda} & \text{if} \quad d_\lambda - c_\lambda < l_{ij} < d_\lambda - r_\lambda \\ 0 & \text{if} \quad l_{ij} \leq d_\lambda - c_\lambda, l_{ij} \geq d_\lambda + c_\lambda \end{cases} \tag{18}$$

where

$$h_\lambda = \frac{1}{1 - \frac{nr_\lambda}{m(r_\lambda - c_\lambda)}} \tag{19}$$

l_ij is the product of 3 sigmoid functions (CNN sig), however it's exact expression is beyond the scope required to operate LoDiS. The parameters r_λ, c_λ, n_λ and m_λ are used to tune the shape of the window function. The number of nearest neighbour considered is given as d_λ.

Squared distance of the centre of mass (d^2 COM)

CNA and pdf calculations intrinsically provide information on the centre of mass for a heterogeneous cluster. As such the squared distance of the centre of mass can be used to generate bias potentials specific to the cluster.

As previously mentioned, the parameters serve either to tune the the shape of the CV function (m and n) or are distances (d_0, r_0 and c_0) in proportion to the bulk lattice constant which is input into the system via the **.pot** file. Naturally the cut-off distance for the pdf related window and sigmoid functions are also system dependent (the bi-metallic case may not be trivial). After many rigorous mono-metallic based simulations, the following parameter values have provided the best results and should be used as a reference when carrying out similar tests.

Parameter	CN	$2^n d$ NN	SFN	CNN win	CNN sig	d^2 COM
r_0	0.147	0.1	0.05	0.3	0.12	_
σ	2.0	50.0	25.0	0.5	0.5	2.0
d_0	$\frac{1}{\sqrt{2}}$	3.4	1.354	4.9	0.71	_
m	12	12	12	6	2	_
n	6	6	6	2	2	_
c_0	_	_	_	0.5	0.24	_

TABLE II A table of best values for parameters for CVs for a homogeneous metal nanocluster. Please note that aside from n and m, all parameters are double precision variables in LoDiS. For CN and C2N, the NN distance d_0 is obtained via the .pot file instead of directly via the input.in

V. TUTORIAL

This section is dedicated to an in-depth explanation of the capabilities of LoDiS, providing details and pointers to enable ease of input. Paradigmatic case studies are presented to give a walkthrough of the entire process and descriptions of the physics beyond the obtained data. Before configuration of the simulation with the choice of 2 input schemes, several files must be prepared beforehand by the user, the first of which is the **.xyz** file containing the initial atom positions:

```
       147
Ag Ag
 Ag    0.00000    0.00000    0.00000    12
 Ag    2.46014    1.52045    0.00000    12
 Ag    2.46014   -1.52045    0.00000    12
 Ag   -2.46014    1.52045    0.00000    12
 Ag   -2.46014   -1.52045    0.00000    12
 Ag    1.52045    0.00000    2.46014    12
 Ag    1.52045    0.00000   -2.46014    12
 Ag   -1.52045    0.00000    2.46014    12
 Ag   -1.52045    0.00000   -2.46014    12
 Ag    0.00000    2.46014    1.52045    12
 Ag    0.00000    2.46014   -1.52045    12
 Ag    0.00000   -2.46014    1.52045    12
 Ag    0.00000   -2.46014   -1.52045    12
 Ag    4.92028    3.04090    0.00000    12
 Ag    4.92028   -3.04090    0.00000    12
 Ag   -4.92028    3.04090    0.00000    12
```

FIG. 5 First few lines of an ***.xyz** file for an Ag 147 cluster. The first line indicates the number of atoms in the cluster whilst the second shows the two elements present. The following rows contain 5 columns, each row being a specific atom in the cluster. The first column tells the element of atom i and the following 3 rows are its (x,y,z) coordinates. The last column gives the initial CNi. Therefore, an Ag 147 cluster will have 149 row altogether in the files.

In the case of a Coalescence run, a second **.xyz** file must be prepared as well for the second cluster. Note that the same file cannot be used for both clusters, doing so will retrun an error and halt the program. In the case of using identical clusters prepare a copy file instead. The second file is the ***.pot** files containing the intracluster interaction parameters. The monometallic and bimetallic pot files are near identical :

```
# INPUT PARAMETRIZATION FOR MolDyn CODE
# METALS
Ag      Co
#####################################################################
# PARAMETRIZATION OF Ag-Ag, Co-Co, Ag-Co INTERACTION
10.79d0                 9.210d0                 10.001d0        p
3.19d0                  2.975d0                 3.085d0         q
0.104331912d0           0.17570d0               0.14440d0       a
1.194019029d0           1.84300d0               1.47760d0       qsi
#####################################################################
# OTHER METAL PROPERTIES
2.95d0          4.45d0                  Cohesion energy [eV]
1.445d0         1.25d0                  Atomic radius [Angstrom]
108.d0          59.d0                   Mass [amu]
#####################################################################
# CUTOFF_START AND CUTOFF_END [Angstrom]
4.087077141d0          4.330126762d0
```

FIG. 6 RGL parameter file for Ag-Co cluster interactions between Ag-Ag, Co-Co and Ag-Co pairs. The corresponding pair parameters are listed under each pair type in the parametrization section. The following sections provide element specific properties and parameters. For a monometallic Ag cluster, the 3 columns within the parametrization section are for Ag-Ag, Ag-Ag and 'N/A' i.e the first 2 columns are identical and the final is a set of 0.000d0.

In the case of simulating a supported cluster, a ***.MgO.pot** file, containing the MgO specific interactions, i.e. the 27+ fitting parameters that code the a_{ij} functions must also be available:

```
# INPUT PARAMETRIZATION FOR BASIN_HOPPING CODE
# METALS
Au      Au
################################################
# PARAMETRIZATION OF Au-MgO INTERACTION
  0.05514599d0  0.02779034d0  0.00211568d0
  1.47270073d0  0.04732589d0  0.02759875d0
  4.82593720d0 -0.25499144d0 -0.22440063d0
  0.62260401d0  0.07095966d0  0.00325932d0
  0.03011466d0 -0.03011500d0  0.01505767d0
 -2.38130416d0  0.16206616d0  0.25536062d0
  4.34957262d0 -0.10813126d0 -0.05043234d0
 -5.55886944d0  2.51045831d0 -0.44518730d0
 34.63980485d0 -1.15728009d0 -1.69702334d0
################################################
# PARAMETRIZATION OF Au-MgO INTERACTION
  0.05514599d0  0.02779034d0  0.00211568d0
  1.47270073d0  0.04732589d0  0.02759875d0
  4.82593720d0 -0.25499144d0 -0.22440063d0
  0.62260401d0  0.07095966d0  0.00325932d0
  0.03011466d0 -0.03011500d0  0.01505767d0
 -2.38130416d0  0.16206616d0  0.25536062d0
  4.34957262d0 -0.10813126d0 -0.05043234d0
 -5.55886944d0  2.51045831d0 -0.44518730d0
 34.63980485d0 -1.15728009d0 -1.69702334d0
################################################
# O-O DISTANCE IN THE OXIDE [Angstrom]
4.21d0          4.21d0
################################################
# alpha          r_c [Angstrom]     ############
5.000d0            3.468d0
```

FIG. 7 The MgO parameters for an Ag cluster. For a bimetallic cluster, the two parametrization sections will be for each element.

Note: Due to the inherent nature of Fortran 90, there is a character length limit for file names including the path of about 31.

A. The interactive approach: Pythonic GUI

The first input scheme available is a simplistic Tkinter general user interface (GUI), which is recommended for new users unfamiliar with LoDiS's structure and various quirks. Note Tkinter was chosen as it is packaged along with the Python programming language, thus reducing the LoDiS suite requirements to simply a base Python and Fortran environment. The GUI code is contained in the **py_interface** folder with a Frame Class structure, each frame coded by a separate file. **All frames are generated at initialisation of the program and thus Frames with dependencies (on a prior parameter for instance) must make use of functions to update them when necessary**. The interface was designed in mind to provide a visual aid and prevent errors generated by incorrect inputs by the user. Running the python program **lodis_gui.py** will start the GUI:

$$python \quad /PATH/LoDiS_{G}IT/py\_interface/lodis\_gui.py$$

/PATH can be either is the relative path between **LODIS_all** and the current working directory or the absolute path. Please note that the current working directory is where the outputs of the program are generated.

The interface comes equipped to automatically provide an image of the cluster the user has selected via **Matplotlib 3D** as well as its chemical composition so as to provide further clarification as to which files have been chosen. In addition, the code will automatically shift said cluster, in the presence of a MgO support, to an optimal distance away to allow for soft-landings. Making use of the benefits afforded by Tkinter, the interface will systematically lock certain parameters given the user's previous inputs or halt moving past the current window given incorrect allocations, so as to prevent error generation during LoDiS runtime, causing the program to exit and waste the user's time. An additional feature is option to run various post analysis calculations unique to this input scheme by clicking on the **run post-process** check-button in the system parameters window. After allocating values for all the parameters and clicking 'confirm' on the final window, the python code will generate a **input.in** file which contains namelists of parameters and their respective values. The file is to be read by LoDiS which is initiated immediately after via the **subprocess check_call** method. The current situation throughout LoDiS's run-time is updated onto the terminal for the user to observe. Post-analysis code will begin soon after LoDiS is finished given the check-button is ticked before simulation in the system parameters window. By ticking the checkbox, a window for the post-analysis parameter will appear before the final confirmation page asking for values for the following parameters:

- snapshots t(ps)/T(K) - a list of approximate times/temperatures to run calculations around.
- min distance - the minimum cut-off distance in Angstrom for calculations.
- cut-off distance - the maximum cut-off distance in Angstrom.
- bin width multiplier - the multiplication factor to the (larger) atomic radius of the element(s) to determine the bin width in calculations.

Within the same window, there will be a checkbox labelled **ID = T (K)**, which will set the units from (ps) to (K) when clicked for **itMD** run. In the case of other processes, the box will be greyed out and calculation will automatically be run in in regards to picoseconds. By setting snapshots = 'auto', the program will select 4 times/temperature spread throughout the simulated range to carry out calculations. Otherwise the user can manually input select values they wish to observe and analyse. Make sure to separate the time/temperature stamps by commas or spaces in the entry so that values are read correctly. Inherently, the code does not have a set limit to the number of stamps the user can select (something that should be added soon), it should be limited to between 4-6 to prevent a clutter of created graphs within a figure. The python code will read the **energy.out**, **movie.out** and (if available) **fort.56** files generated by LoDiS to calculate:

- pair-distance distribution (PDF)
- radial-distance distribution (RDF)
- analytic coordination number (CN)
- analytic general coordination number (GCN)
- velocity density of states (VDOS) (not fully implemented)

For the PDF and RDF, graphs will be generated at each time/temperature stamp within the lower and upper bounds set by the user in the post-processing page. The bin-width multiplier function will determine the width of the histogram bars as well as the smoothness of the curve. To create smooth profiles for the PDF and RDF whilst avoiding extensive loss of information, the optimal value of the bin width multiplier = 0.04 is set by default. These graphs are saved into the output directory. The CN and GCN are calculated for the smallest and largest times/temperatures and are plotted into radial-distance vs CN (2D) and radial-distance vs CN vs GCN (3D) graphs. Due to the nature of the Matplotlib 3D, the initial viewing angle can be problematic. Consequently the 3D graphs are visually plotted rather than saved so that the user can rotate to the ideal angle to view the figure and then save. For a slightly different reason, the VDOS graph is plotted so that the user may select the appropriate range before saving.

Please note that the current post-analysis code was designed with itMD in mind and thus currently isn't equipped to process simulations in which atom populations change i.e.

Growth and Coalescence. Due to this reason, the choice to run further calculations is greyed out for these two processes. Further development will be to extend the code to these processes and improve the VDOS calculations.

Furthermore, due to not being fully tested, the CNN CV for the Metadynamics isn't presented as an option within this input scheme. To run use this process, make use of the second scheme available.

B. The direct approach: input.in file

The second scheme is the use of a **input.in** file mentioned in the previous scheme. Within the **LoDiS_GIT/base** folder, such a file is already present and can be copied over to an another directory. This particular file within the base directory comes with all the namelists. Note that the **input.in** files generated through the use of the interface are specific to the intended process that was simulated and thus will not have all the namelists written into the file. If rerunning the simulation or running the same process with different values, using the interface generated will be no issue, otherwise the other namelist must be written into the file. To save time, simply use the formerly mentioned **input.in**. This scheme is useful when running many simulations, simulations on remote computers and rerunning simulations with little hassle. As it simply makes use of the base Fortran code, it allows modification of the core LoDiS program without need to also update the python interface (useful for development and testing). Naturally incorrect inputs will only be flagged during LoDiS's run-time and thus this method is recommended for those familiar with the program or at least this manual.

```
&simul
  irand  = 4001,                    ! Odd 4 digits number lower than 4095
  tstep  = 5.d-15,                  ! Time step smaller than phonon frequencies, usually 1-7 fs
  npas   = 200000,                  ! Total number of steps (2000000000)
  scrivo = 2000,                   ! After how many steps the program writes
  npast  = 1000,                    ! Thermalization time
  tinit  = 200,                     ! Initial temperature
  vnu    = 5.d11,                   ! Andersen thermostat frequency (5.d11)
  type_process = 'canon',          ! Process to simulate: 'quenching', 'canon', 'melting',
                                    !'freezing', 'coalescence' or 'metadynamics'
                                    !
  output_xyz = .true.,              ! Output movie .xyz format (with hdf5 format)
  mgo_substrate = .false.,          ! Run process with the substrate force-field?
  geometry_type = 1
  metal_on_top  = .false.,          ! For the MgO force-field with correction for the metal-on-top effect
  sticky_atoms  = 0,                ! Number of atoms bound to their starting position by a spring
  sticky_k      = 10,               ! Elastic constant (F=-k*x) [eV/A^2]
  cn_cal        = .false.,          ! Use Polynomial formalisation to calculate the coordination number?
                                    !(Otherwise uses the Fermi distribution formalisation)
                                    !
  filepos      = 'Ag147.xyz',            ! Initial atom positions file, ONLY .xyz format
  filepot      = 'Ag_Ag.pot',            ! Potential parameters file, ONLY .pot format
  mgo_pot      = 'Ag_Ag.MgO.pot',           ! File of parameters for MgO substrate, ONLY .pot format
  impl_env = .true.
  pot_a = 1.5d0
  pot_b = 1.5d0
  eta_a = 0.02d0
  eta_b = 0.02d0
/

&system
  type_potential  = 'rgl',          ! Potential type between 'lj' (for noble gases), 'rgl' (for metallic systems)
                                    ! and Girifalco (for carbon systems)
                                    !
  natom           = 147,            ! Initial number of atoms in the cluster
  fattor          = 1.d0,           ! LEAVE IT, stupid units conversion
  elem1           = 'Ag',           ! Chemical species one
  elem2           = 'Ag',           ! Chemical species two
  sys             = 'mon',          ! Monometallic cluster 'mon' or bimetallic cluster 'bim'?
/

&canon
  vel_af = .true.
```

FIG. 8 Example of a LoDiS input for an NVT run. The variable unique to NVT is listed under the &canon namelist. The simulation is marked to run with an implicit environment and unsupported ($mgo\_substrate$ = '.FALSE.')

After copying and modifying the input file, LoDiS can be directly started by running the executable created by compiling the Makefile, the name of which is determined via the Makefile, the default being **LODIS_all**:

$$/PATH/LoDiS\_GIT/base/LODIS\_all \quad < /PATH2/input.in(> \quad output.out)$$

Both **/PATH** and **/PATH2** are either the absolute paths or the relative paths from the current working directory to their respective files. For simplicity, it is best to set the current working directory as the same as that of the **input.in** file so that **/PATH2** is simply " and the outputs are contained within the same directory with the input file to acting as a reference. The bracketed part is an optional addition to write the **PRINT** statements into a **output.out** file instead of onto the terminal (useful for long simulations on remote systems for instance).

C. Implicit environment

Note: An implicit environment cannot be used in a Growth run in this current version of LoDiS and will return errors if attempted.

By setting the parameter *impl_env* = '.true.' a run will be simulated with an environment, the nature of the interaction, ρ, determined by *pot_a* and *pot_b* for each chemical species. In the same fashion, the interaction strength ϵ is given by *eta_a* and *eta_b*. In the case of running a mono-metallic simulation, the parameter **cn_cal** can be set to either '.TRUE.' or '.FALSE.' to calculate the CN_i via Fermi distribution or polynomial formalism respectively. In the bimetallic case, it is generally better to run with '.False.'.

D. MgO substrate

Note: the use of a substrate is still in the testing stage and therefore the base algorithm doesn't work for growth and coalescence. In addition, the metal_on_top variation is unfinished and will return errors if run. For these reasons, the parameter *metal_on_top* **should be set to '.false.' (no mention in the GUI)**

The presence of a double square MgO substrate can be turned on by simply inputting *mgo_substrate* = '.true.'. Be aware that the substrate is placed into the system at z = 0 and therefore the cluster must be either constructed at an appropriate distance away from the origin or repositioned for the simulation to succeed. The python interface scheme automatically carries out the repositioning of the cluster, however when using the **input.in** method, it must be done manually. Running the python script **/base/CCP7.py** will create a substitute ***_repos.xyz** containing the reposition cluster much alike the interface method.

Process	Implicit environment	Double square substrate
NVE	y	y
NVT	y	y
Quench	y	y
itMD	y	y
Growth	n	n
Coalescence	y	n
Metadynamics	y	y

TABLE III A table listing which processes can be run with an implicit environment and substrate for both monometallic and bimetallic clusters, where y = yes and n = no. **Note: simulations that can run with either can also be run with both!**

E. Taxing simulations

As one would expect, investigating events over extensive virtual time periods and/or with large clusters (1000+) will diminish processing speeds and tax memory rapidly. To reduce the burden and speed up simulations, $*$**_light.f90** files are incorporated which forgo differential simulations carried out by their 'twin' files. The files that have a 'light' version are:

- **bim_cn.f90**
- **comneighfun.f90**
- **coord_no.f90**
- **d_com.f90**
- **secneigh_no.f90**
- **stackfault_no.f90**

To utilise the cheaper 'light' versions, change the subroutine names called within the code and recompile the code (an option to easily switch between 'lighter' and 'heavier' calculations is another direction of future development).

F. Case studies

The following simulations were simulated in the gas phase for means of simplicity as some processes cannot be run with a substrate for instance. Aside from the NVT case study **input.in**, process specific namlists will only be shown in figure to avoid retreading ground.

NVT

Running the above **input.in** in figure 8 will simulate a cononical NVT with an strong interacting implicit environment ($impl\_env$ = '.TRUE.', $pot\_a > 1$). The Verlet algorithm timestep $tstep$ is set to 5fs, whilst the andersen thermostat regulates the temperature initially set to 200K. The simulation lasts a total amount of npas = 200000 steps which translates by the following relation:

$$virtual\ time\ simulated\ =\ tstep\ *\ npas$$

into 1ns of canonical evolution. We consider a mono-metallic silver cluster of 147 atoms, the file **Ag_147.xyz** specifies the coordinates of each atom in the nanocluster. Interaction between atoms in the cluster is described by the Rosato-Guillope-Legrande potential, parametrized according to what reported in the Ag-Ag.pot. The parameter vel_af = '.TRUE.', which tells the system to call the velocity auto-correlation function.

Six output files, including the caloric **energy.out** and trajectory **movie.xyz** files, are generated within the same directory as the **input.in**. In the energy.out file the following data are stored: number of elapsed timesteps, amount of elapsed time in picoseconds, total potential and kinetic energy, averaged energies and temperature of the sistem in the previous range. The .xyz file contains the number of atoms and coordinates of the cluster during the relaxation process. The final pr.out file contains the fully relaxed structure.

The subsequent examples will only provide the additional process specific namelists to avoid retreading. Descriptions on all the parameters in the **input.in** file can be found in the Github LoDiS Documentation page,
https://github.com/kcl-tscm/LoDiS/wiki/LoDiS-Documentation.

Quenching

As a paradigmatic example of a quenching simulation we use as an initial structure a perfectly icosahedral Ag_{147} nanocluster which has been constructed via a geometrical algorithm. The input reported in Figure 8 is used to run the simulation, albeit with 'quenching' for type_process and an additional namelist (Figure 9). The initial temperature is set at 600K, 50 ps of thermalization are allowed, after which the quenching procedure starts until a final temperature of 10^{-5} is registered. Structures and energies of the system are written in the output every 10 ps.

```
&quench
  itremp          = 5000,        ! after itremp steps the quench starts
  tmin            = 1.d-5,       ! minimum temperature
/
```

FIG. 9 Example of a LoDiS input quenching namelist for an Quenched run.

The potential energy of the system decreases as the system relaxes towards the minimum configuration while the kinetic energy, and thus the temperature, of the system became null. By plotting the pair distance distribution function of the atoms of the geometric and relaxed nanocluster we observe how relaxation leads to changes in the neighbouring peaks and occurrences.

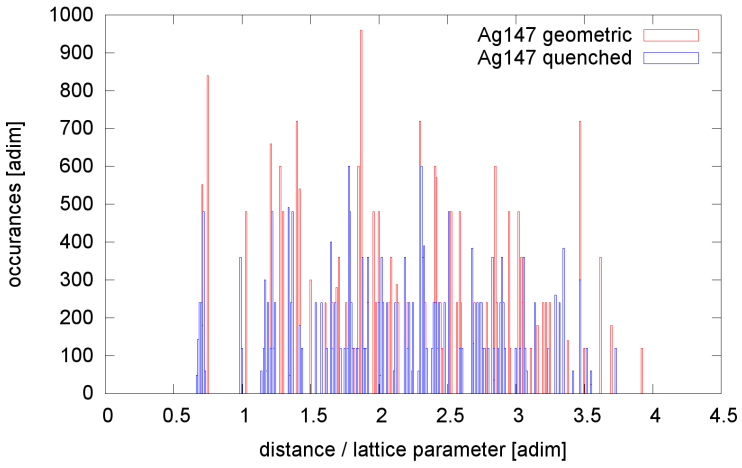

FIG. 10 Pair distance distribution function for the perfectly geometric and relaxed Ag_{147}.

Microcanonical NVE

A microcanonical NVE run can be initiated either through a canonical NVT or via quenching and therefore doesn't have a specific namelist. As the coupling to the heat bath dictates whether a run is NVT or NVE, by simply setting the Anderson thermostat frequency

$$vnu => 0$$

will uncouple system to the bath causing an NVT => NVE. A quenching run can be converted to an NVE by setting

$$itremp > npas$$

to prevent the rapid annealing from occurring (9).

Melting

Molecular dynamics runs at increasing temperatures can be automatically concatenated, allowing for thermalization, to investigate melting. In this section, we present an example of these calculations for Ag_{147}.

```
&calor
  deltat          = 50.d0,            ! temperature step
  tcaloric        = 800.d0,          ! final temperature
```

FIG. 11 Example of a LoDiS input melt namelist for an melted run.

The initial temperature of the system is set at 400K, 10 ps of thermalization, temperature increases by ΔT=50K every $\Delta\tau$=1ns:

$$melting\ rate\ =\ deltat/(npas\ *\ tstep)$$

until a temperature of 800K is reached. Structure and energies of the system are written in the output every 10 ps.

After running the simulation the previously mentioned energy.out file and a movie.xyz file have been produced. By plotting the potential energy of the system as a function of temperature we observe a sharp jump around 700K signalling the onset of a melting transition, which is confirmed by visual inspection of the registered nanocluster architectures, Figure 12 .

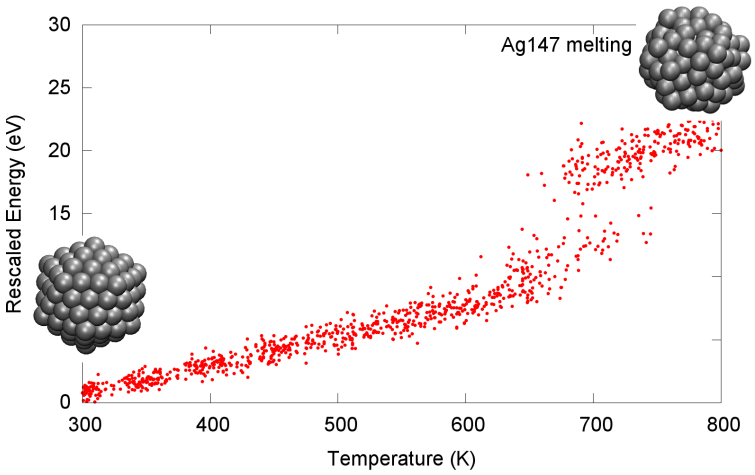

FIG. 12 Caloric plot for the melting of an Ag_{147}. Total energy of the system is rescaled with respect to the initial Ih structure.

Freezing

Freezing processes may be investigated in the same fashion of melting. Here we discuss this process in the case of a melted Ag_{147}. The initial temperature of the system is set at 800K, 10 ps of thermalization, temperature decreases by ΔT=50K every $\Delta\tau$=1ns until a temperature of 400K is reached. Structure and energies of the system are written in the output every 10 ps.

```
&calor
  deltat          = 50.d0,            ! temperature step
  tcaloric        = 400.d0,          ! final temperature
```

FIG. 13 Example of a LoDiS input melt namelist for a freezing run.

After running the simulation the previously mentioned energy.out file and a movie.xyz file have been produced. By plotting the potential energy of the system as a function of temperature we observe a sharp drop at 650K signalling freezing of the nanocluster. We remark that freezing corresponds to the transition of the free energy landscape from a well-connected and equiprobable number of basin to a single local minimum. Thus an average of these process is necessary to tackle polydispersivity of sample and characterize and compare with experimental population distributions.

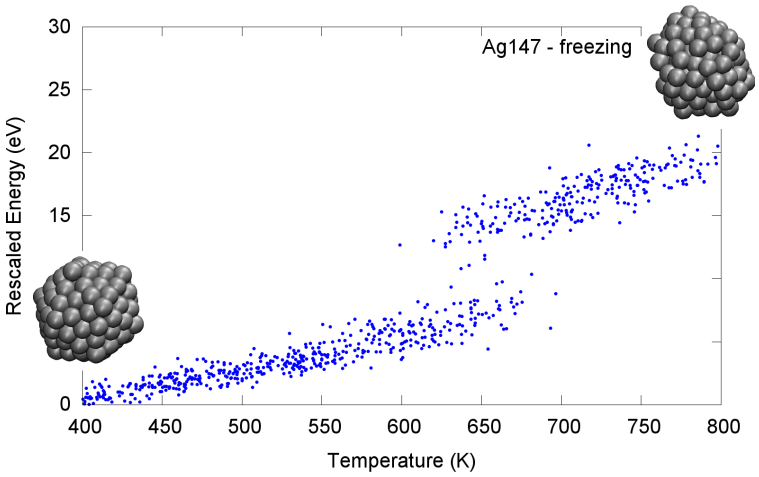

FIG. 14 Caloric plot for the freezing of an Ag_{147}. Total energy of the system is rescaled with respect to the final Ih structure.

Atom by Atom growth

Another characteristic process of interest to understand how a nanocluster is formed in vitro is atom-by-atom deposition or growth. To simulate such a phenomenon an original seed is set as the initial structure while the number of atoms to be added is indicated in the relative namelist together with the amount of time between two depositions. The rest of the input parameters are set up as in the case of a classical Molecular Dynamics run. In essence, growth is an iterative MD with overarching deposition loop.

$$virtual\ time\ =\ ndepmax\ *\ npas\ *\ tstep$$

Three variations of growth are coded into the program and are chosen by fixing the parameter lcs:

- $lcs = 1$, monometallic growth
- $lcs = 2$, mixed-shell growth
- $lcs = 3$, core-shell growth

As the name implies, monometallic growth is the deposition of atoms of the same chemical species as the atoms within the cluster.The system automatically sets the element of the deposited atoms as $elem1$ and only reads the parameters $ndepmax$, $tsorg$ and rad from $\&growth$ when $lcs = 1$. However for ease of post analysis, it is ideal to write

$elemd = elem1$. The example namelist for a monometallic growth of Ag 147 is displayed below. Mixed-shell offers slightly more flexibility in regards to input and procedure. A

```
&growth
ndepmax = 25          !number of deposited atoms
lcs = 1               !1 (A+A), 2 (A+B), 3 (A/B + A/B)
at_tipo2 = 0          !number of deposited atoms of species 2
elemd=elem1           !chemical species of deposited atoms
prob=1.0d0            !probability of deposition chem spec 1
tsorg=1500.d0         !temperature of the source
rad=6.d0              !radius of the source
/
```

FIG. 15 Example of a LoDiS input growth namelist for a growth run.

mixed-shell growth can begin with either a mono- or bimetallic seed (monocore/bicore) which undergoes subsequent depositions of atoms whose chemical species is governed by probability centred around the parameter *prob*. When $randn < prob$ the deposited atom will be of the secondary chemical species specified by *elemd*, *randn* being a random number called by the program. In the other case the atoms will be of the primary element. The total number of *elemd* atoms deposited is given by at_tipo2, upon reaching this threshold, *prob* is set to zero and the remaining depositions are of the primary element. For a mixed-shell growth with either a monocore or bicore, setting $elem1 \neq elem2$ and $elemd = elem2$ is the easiest approach. In both cases, the .pot file must be representative of interactions between the two chemical species and $sys =$ 'bim'.

Much like the mixed-shell growth, core shell growth can initiate with either a monometallic or bimetallic core, regardless of which *sys* must be set to 'bim'. Core-shell growth is simply the deposition of a selected element to produce an outer shell which differs in chemical composition to the core. In the monocore case, the core and shell are monometallic but with different elements. For the bicore, the shell will be composed solely from one of the two species in the core. To achieve monocore growth, the best practice is to set $elem1 = elem2$ and $elemd \neq elem1$, choosing a monometallic cluster .xyz file and a .pot file with parameters for interactions between the two chemical species. In a similar way, $elem1 \neq elem2$ and $elemd = elem2$ with a bimetallic .xyz file and a bimetallic .pot file is the easiest approach to setup for a bicore growth.

The file resulting from the example simulation are energy.out file and out-number.xyz.

An interesting phenomenon can be observed by plotting the total number of atoms in the cluster versus the total energy of the system and the energy of the system rescaled with respect to the excess energy Δ:

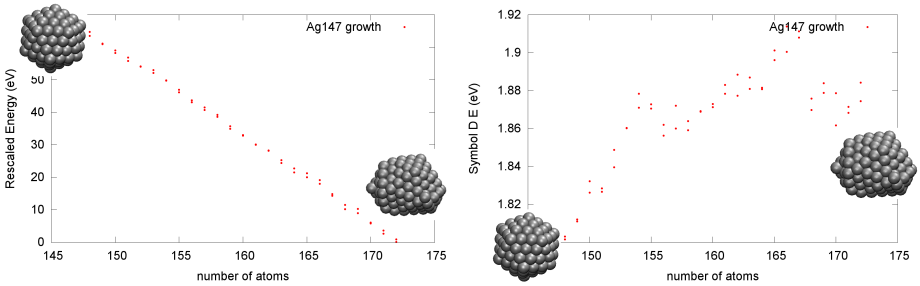

FIG. 16 Evolution of a)the total energy and b)the excess energy of the system with respect to the deposition of new atoms. Total energy of the system is rescaled with respect to the final structure.

The total energy of the system decreases because new bonds are formed by the deposited atoms, yet, the excess energy of the nanocluster is increasing because of anisotropy of the newly formed structure. The initial icosahedron is one of the so called closed-shell polyhedra and is an energetically favourable structure, especially at small sizes, because it maximizes the number of bonds between atoms in the nanocluster, thus its Δ is lower with respect to the one of the other observed structures.

Coalescence

LoDiS has the capability to coalesce clusters that are of similar chemical composition. In essence, it is possible to simulate a monometallic and bimetallic cluster together provided the chemical species in the first are present within the second. **Be aware, in this procedure, the parameter** *natom* **refers to the shared total number of atoms between the 2 clusters!** Due to the structure of the algorithm, LoDiS will keep first .xyz file open whilst reading the second meaning using the same file for both clusters will return an error and prevent initialization. To coalesce identical clusters, there must also be two .xyz files to read.

In this particular example, an Ag 309 and 146 atom clusters were simulated with an initial starting distance of 100 angstroms.

By plotting the time against the number of intercluster bonds and the excess energy, a jump in the excess energy is clearly visible. This is likely due to an increase of the anisotropy

```
&coal
  filepos2 = 'Ag_146.xyz'            ! Initial atom positions file for coalescence cluster, ONLY .xyz format
  natom2 = 146                       ! Initial number of atoms in the coalescence cluster
  somedist = 100.0d0                 !
```

FIG. 17 Namelist for the coalescence procedure. The parameter *somedist* is the initial separation between the two clusters

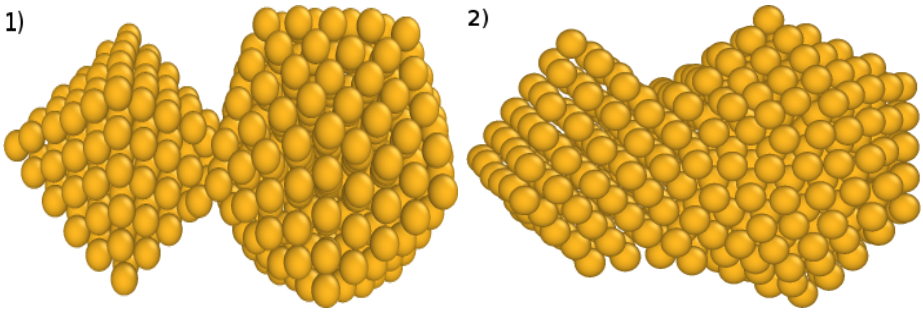

FIG. 18 Visual representations of the clusters at 1) the beginning and 2)end of a coalescence run between Ag 309 and 146.

of the aggregate in comparison to the individual clusters presented in figure 18. Much like the growth case study, the increase in interatomic bonds formed accounts for the initial and subsequent drop after the peak in the second plot as well as an overall decrease in the total energy of the system.

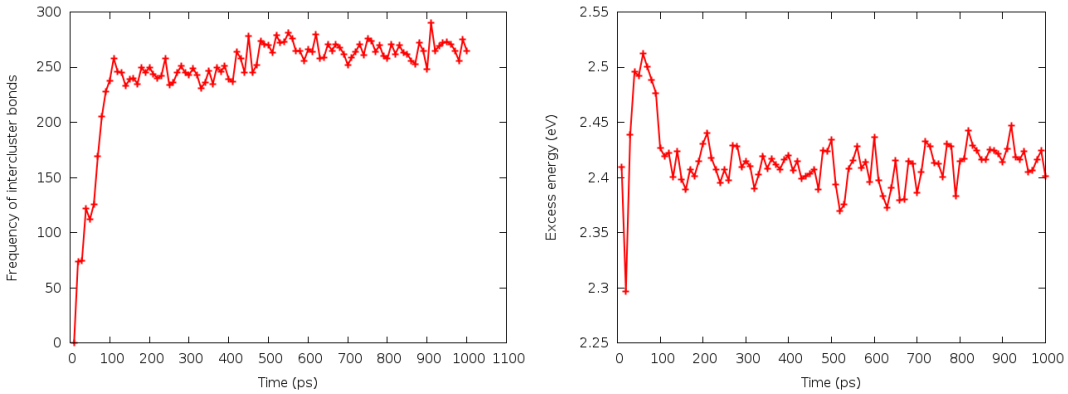

FIG. 19 A graph plotting the change in frequency of interatomic bonds over time during a coalescence run between Ag 309 and Ag 146 clusters.

Metadynamics

For a metadynamics run, there is the choice between 5 different CVs; 3 for monometallic (CN, C2N and SFN) and 2 for bimetallic (CN bimetallic and d^2 COM). Due to its lesser

accuracy compared to CN, C2N and SFN, CNN has seen little use and testing. Consequently, the current algorithm related to CNN is unlikely to work without modification. As previously mentioned, a maximum of 2 CVs can be used in a run, but in the case of using only 1, the parameter *collvar_name*(2) must be the one listed as 'none', the reverse case will prevent the procedure from starting (21). Furthermore, a monometallic cluster cannot undergo metadynamics with a bimetallic CV and vice versa.

In the example namelist, the system is set to run a single CV metadynamics of Ag 147 in a stronly interacting environment ($\rho = 1.5$, $\epsilon = 0.02$) with CN as the choice. The experiment is to last 100ns at a temperature of 400K with an output written every 0.1ns. A Guassian of height=0.75eV and width=2eV would be produced every half increment of output generation.

```
&metalist
  collvar_wanted  = .false.,         ! to write CV values and average forces (total and MT)
                                     ! at each time step during whatever procedure
                                     !
  metaframe       = 2000,            ! after how many steps a frame of movie.xyz is generated
  gheight         = 0.75d0,          ! [eV] Gaussian height
  metaperiod      = 1000,            ! after how many steps a new gaussian is added
  num_cv          = 1,               ! number of CVs used (1 or 2)
  collvar_name(1) = 'coord_number',  ! 'coord_number', 'C2N', 'SFN', 'CNN', 'none'
  collvar_name(2) = 'none',           ! 'CN_bim', 'd_com'
  !-----------------------------------------------------------------------------------------
  ! coord_number (CN)          - Sigmoid Function
  n_pwr           = 6,               ! n
  m_pwr           = 12,              ! m
  rzero           = 0.147d0,         ! r0 [latt bulk ref]
  gwidth          = 2.d0,            ! Gaussian width
  !-----------------------------------------------------------------------------------------
  ! C2N (2nd Neighbour No.)    - Window Function 1            >>> d3.4N <<<
  n2n_pwr         = 6,               ! n
  m2n_pwr         = 12,              ! m
  d2n             = 3.4d0,           ! d0 [latt bulk ref]
  r2n             = 0.1d0,           ! r0 [latt bulk ref]
  g2nwidth        = 50.d0,           ! Gaussian width
  !-----------------------------------------------------------------------------------------
  ! SFN (Stacking Fault No.)   - Window Function 2
  nsf_pwr         = 6,               ! n
  msf_pwr         = 12,              ! m
  dsf             = 1.354d0,         ! d0 [latt bulk ref]
  rsf             = 0.05d0,          ! r0 [latt bulk ref]
  gsfwidth        = 25.d0,           ! Gaussian width
  !-----------------------------------------------------------------------------------------
  ! CNN (Common Neighbour No.) - Common Neighbour Function
  cnf_in(1)       = 0.71d0,          ! d0 : 1st coord sphere [latt bulk ref]
  cnf_in(2)       = 0.12d0,          ! r0 : 1st coord sphere [latt bulk ref] from d0
  cnf_in(3)       = 0.24d0,          ! c0 : 1st coord sphere [latt bulk ref] from d0
  cnf_in(4)       = 2,               ! m  : 1st coord sphere
  cnf_in(5)       = 2,               ! n  : 1st coord sphere
  !-----------------------------------
  win_in(1)       = 4.9d0,           ! d0 : window function
  win_in(2)       = 0.3d0,           ! r0 : window function from d0
  win_in(3)       = 0.5d0,           ! c0 : window function from d0
  win_in(4)       = 6,               ! m  : window function
  win_in(5)       = 2,               ! n  : window function
  !-----------------------------------
  gcnnwidth       = 0.5d0,           ! Gaussian width
  !=========================================================================================
  ! CN_bim (CN bimetallic)     - Sigmoid Function bimetallic
  cn_aa           = .false.,         ! pairs elem1-elem1
  cn_bb           = .false.,         ! pairs elem2-elem2
  cn_ab           = .true.,          ! mixed pairs
  cn_n_pwr        = 6,               ! n
  cn_m_pwr        = 12,              ! m
  cn_rzero        = 0.147d0,         ! r0 [latt bulk ref]
  cn_gwidth       = 2.d0,            ! Gaussian width
  !-----------------------------------------------------------------------------------------
  ! d_com (Squared distance of CoM) [latt bulk ref element A ^2]
  d_com_gwidth    = 2.d0,            ! Gaussian width
```

FIG. 20 Namelist for the metadynamics procedure. The CV vales listed are the same as those written in II. Only the CN parameters are used in this particular run.

Running the experiment will generate two file of interest: **meta.out** and **metahistory.out**. The first contains the all the caloric and CV data whilst the second file has information on the all the Gaussians generated throughout the run. Figure 21 is the result of plotting the CV and excess energy data. Both graphs display no significant change in the plot shape although there is noticeable increase in energy barriers between local minima as time goes on. This suggests that no new structural basins were explored within the entire lengthy period, likely due to influence of the implicit environment modifying the energy landscape.

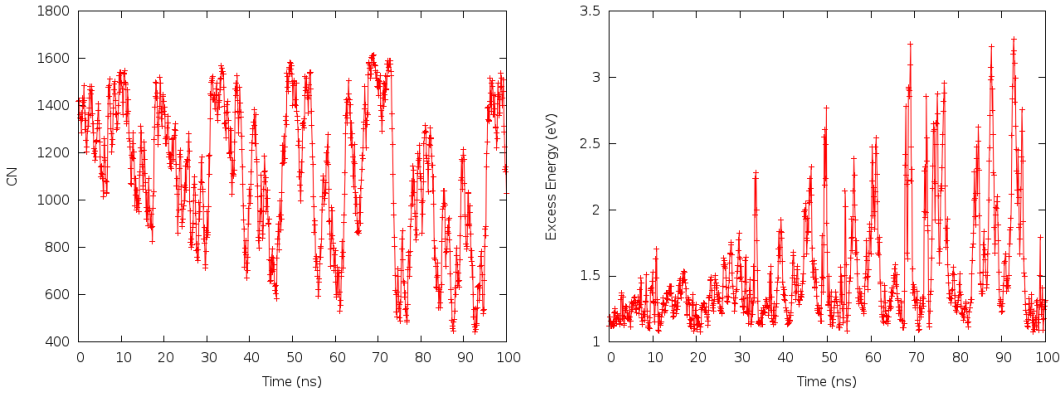

FIG. 21 Two graphs plotting the time against 1)the CN and 2)the excess energy.

References

L. Verlet, Phys. Rev. **159**, 98 (1967).

H. C. Andersen, The Journal of Chemical Physics **72**, 2384 (1980), https://doi.org/10.1063/1.439486 .

I. Atanasov, G. Barcaro, F. Negreiros, A. Fortunelli, and R. Johnston, The Journal of chemical physics **138**, 224703 (2013).

V. Rosato, M. Guillope, and B. Legrand, Philosophical Magazine A **59**, 321 (1989), https://www.tandfonline.com/doi/pdf/10.1080/01418618908205062 .

F. Cleri and V. Rosato, Phys. Rev. B **48**, 22 (1993).

F. Baletto, R. Ferrando, A. Fortunelli, F. Montalenti, and C. Mottet, The Journal of Chemical Physics **116**, 3856 (2002), https://doi.org/10.1063/1.1448484 .

J. E. Jones and S. Chapman, Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character **106**, 463 (1924), https://royalsocietypublishing.org/doi/pdf/10.1098/rspa.1924.0082 .

L. A. Girifalco, The Journal of Physical Chemistry **95**, 5370 (1991), https://doi.org/10.1021/j100167a002 .

R. Cortes-Huerto, J. Goniakowski, and C. Noguera, The Journal of Chemical Physics **138**, 244706 (2013), https://doi.org/10.1063/1.4811670 .

W. Vervisch, C. Mottet, and J. Goniakowski, Phys. Rev. B **65**, 245411 (2002).

F. Baletto, C. Mottet, and R. Ferrando, Phys. Rev. Lett. **84**, 5544 (2000).

F. Baletto, Journal of Physics: Condensed Matter **31**, 113001 (2019).

A. Laio and M. Parrinello, Proceedings of the National Academy of Sciences **99**, 12562 (2002), https://www.pnas.org/content/99/20/12562.full.pdf .

G. A. Tribello, J. Cuny, H. Eshet, and M. Parrinello, The Journal of Chemical Physics **135**, 114109 (2011), https://doi.org/10.1063/1.3628676 .

G. Santarossa, A. Vargas, M. Iannuzzi, and A. Baiker, Phys. Rev. B **81**, 174205 (2010).