Title

Attempts in generalising the description of a local environment, used for off-lattice atom-resolved kMC

Author

Miha GUNDE (PhD student)

Collaborations

Normand MOUSSEAU, Université de Montréal, Canada
Layla MARTIN-SAMOS, CNR-IOM Trieste, Italy

Keywords:

kinetic Monte Carlo, local environment descriptors, geometry, graph theory

Context

Simulation of phenomena that naturally occur in longer time scales, e.g. surface oxidation, with an accuracy of atomic-scale methods, e.g. Density Functional Theory (DFT), is much-desired goal in several scientific communities. The problem is that the atomic-scale methods work with time scales which are several orders of magnitude lower than the time scales of the phenomena desired to study. There are methods that have been designed to simulate phenomena at any desired scale, but if one wants to precisely simulate higher-scale phenomena, then all lower-scale phenomena that can happen must be taken into account. Meaning that one should start simulations at the lowest possible scale, find all possible phenomena that can reasonably occur, and then simulate them at an accordingly higher scale.

Due to the several orders of magnitude of difference of scales, any phenomena occurring at the higher-scale is a rare-event at a lower scale. Attempting to simulate higher scale phenomena with any lower scale method generally means a huge computational load and human effort, with no guaranteed outcome. Therefore one can think of a higher scale phenomenon as series of lower-scale ones. Then,

one could generate the involved lower-scale phenomena using a low-scale computational method, and use the results as inputs to a higher-scale computational method.

A method employing this philosophy is kinetic Monte Carlo (kMC). It propagates given phenomena (events) according to their statistical probability, which results in some evolution of a higher-scale phenomenon.

A mechanism at the core of kMC philosophy is that of applying a known, pre-described transformation of configuration (an event) to a site configuration found in a system under study, that is recognized as equivalent to the initial state of that event. If one can assume that the system under study can be imposed on a grid (lattice), the way of solving this problem is greatly simplified. However this assumption is quite limiting the choice of systems one could study. The goal is then to find a way of solving this problem in a slightly more general way.

Methodology

The prescription of a kMC move is given as a transformation T from an initial configuration $\mathbf{R}_{ini}^{ref}$ to a final configuration $\mathbf{R}_{fin}^{ref}$, which together with a probability associated to this transformation form an entry in a kMC event catalogue. For the system under study, any local site configuration $\mathbf{R}^{sys}$ that is considered equivalent to the initial configuration of some event $\mathbf{R}_{ini}^{ref}$ is deemed as a possible site for eventual execution of that event. But since the local site configuration $\mathbf{R}^{sys}$ will in general be translated and rotated with respect to $\mathbf{R}_{ini}^{ref}$, some manipulation is required to properly execute the transformation T. An event is written in some basis set β as:

$$\mathbf{R}_{ini}^{ref} \xrightarrow{\beta T \beta^{-1}} \mathbf{R}_{fin}^{ref} \tag{1}$$

Or rewritten as:

$$\beta^{-1}\mathbf{R}_{ini}^{ref} \xrightarrow{T} \beta^{-1}\mathbf{R}_{fin}^{ref} \tag{2}$$

If this transformation T is to be applied on configuration $\mathbf{R}^{sys}$ that is recognized as appropriate, then a basis γ for this site needs to be found such that:

$$\beta^{-1}\mathbf{R}_{ini}^{ref} = \gamma^{-1}\mathbf{R}^{sys} \tag{3}$$

So that the transformation T can be executed to obtain a final state configuration in the system $\mathbf{R}_{fin}^{sys}$ as:

$$\mathbf{R}_{fin}^{sys} = \gamma T \beta^{-1}\mathbf{R}_{ini}^{ref} = \gamma \beta^{-1}\mathbf{R}_{fin}^{ref} \tag{4}$$

However, the Eq. (3) needs a proper interpretation. In general, the configurations $\mathbf{R}_{ini}^{ref}$ and $\mathbf{R}^{sys}$ even when written in basis which give equivalent descriptions

can have permutations in vector indices. This should be taken into account when evaluating the equivalence relation. Another thing is the "flexibility" of the equivalence relation in Eq. (3), as the two configurations might not be exactly equal, but can still be considered *close enough* to execute the transformation T.

The Eq. (3) is a central point of the algorithm, and is thus turned into a condition. If a configuration $\mathbf{R}^{sys}$ written in a basis set γ satisfies a properly interpreted equivalence relation within some tolerance criterion to a configuration $\mathbf{R}^{ref}_{ini}$ written in a basis set β, then a known transformation can be executed on $\mathbf{R}^{sys}$ to obtain $\mathbf{R}^{sys}_{fin}$ as given by Eq. (4).

The task is then to set up a way of evaluating the equivalence relation, satisfying given conditions. Evidently, it needs to be sensitive to descriptions of the same configuration in different basis sets. That is, it needs to be variant under rotations.

The algorithm works in two stages. In the first stage a local configuration $\mathbf{R}^{sys}$ is set up according to a criterion (spherical cutoff radius R_{cut}) around a central atom. It is rewritten such that the central atom is at the origin of the configuration. Then a simple colored graph is set up according to the connectivity of the local configuration. The graph is then compared to a set of graphs generated for the initial state configurations of all possible events $\{\mathbf{R}^{ref}_{ini}\}$, as graph isomorphism problem. This is done using the canonical graph hashing in the dense form of NAUTY package [2], which assigns to a graph an integer number based on the connectivity of graph vertices and their colors, in a permutationally invariant way. Since the connectivity itself is a rotationally invariant property, this description is not sensitive to the specific basis choice. The equivalence of two integer numbers obtained in this way signifies isomorphism of underlying graphs, and thus equivalent connectivity. However, as the topological space of graphs has more symmetries than the Euclidean 3D space of coordinates, isomorphism of simple colored graphs is not a sufficient condition to univocally declare geometric equivalence of two configurations and directly launch execution of an event on that site. Although, isomorphism of graphs of configurations $\mathbf{R}^{sys}$ and $\mathbf{R}^{ref}_{ini}$ is a necessary condition for two configurations that might satisfy an equivalence condition of Eq. (3).

The second stage of the algorithm is done for configurations where graph isomorphism has been found. In this stage a search for basis sets is performed. To reduce the search space of this action, a rule for generating vectors of the basis set of any configuration $\mathbf{R}$ is imposed. The rule is that the first basis vector $\hat{\mathbf{e}}_1$ is the normalised vector of any atom that is connected to the central atom, the second basis vector $\hat{\mathbf{e}}_2$ is the normalised perpendicular part of a vector to another atom that is connected to the central atom and not collinear with the first one, and the third basis vector $\hat{\mathbf{e}}_3$ is a cross product of the first two basis vectors. The basis set is then orthonormal, and a minimum number of non-collinear atoms needed in a configuration to be able to form a basis is 3. Notice that with 3 non-collinear

atoms there are 2 ways to form a basis depending on which atom is taken as first or second. To make the choice of β and γ unambiguous, the set β is defined for each configuration in the set $\{\mathbf{R}_{ini}^{ref}\}$ in the initial phase of the algorithm, before the graph isomorphism stage. In this way the search for a basis is effectively performed only over the configuration $\mathbf{R}^{sys}$ in order to find the basis set γ. As there are multiple possibilities to form a basis set depending on the number of connected atoms, each candidate basis set is used to test the condition in Eq. (3). If a basis set satisfies that condition, further action can be taken as desired. For example, all basis sets that satisfy the condition of their related event, for all sites in the system, can be written to memory. Such that when an event is chosen for execution it can be traced back to specific site and basis set (direction of execution).

Since the condition given in Eq. (3) needs to be evaluated in a way that is sensitive to basis change, and invariant to index permutation, we define the following. A configuration $\mathbf{R}$ is written in terms of a related basis set $\gamma = \{\hat{\mathbf{e}}_{\alpha=1,2,3}\}$, to obtain the term $\gamma^{-1}\mathbf{R}$. For each such term $3k$ functions are generated, where k is the number of different colors (atomic types) present, along each axis α of the basis set in the range of R_{cut}. Each atom in the configuration is projected to each of the basis vectors, the projection value defines the center of a Gaussian peak on the corresponding axis α, whose total area is related to the distance to that atom. Written more precisely,

$$f_\alpha^k(q_\alpha) = \sum_i^N \frac{1}{\sigma_k\sqrt{2\pi}} exp\left(\frac{(\Delta q_\alpha - \langle \mathbf{R}_i^k, \hat{\mathbf{e}}_\alpha\rangle)^2}{2\sigma_k^2}\right) exp\left(-\frac{|\mathbf{R}_i^k|}{\mu_k}\right) \tag{5}$$

where q_α is a dummy variable going along the axis α, $\langle \mathbf{R}_i^k, \hat{\mathbf{e}}_\alpha\rangle$ is the projection value of a vector of atom index i of type k on a basis vector $\hat{\mathbf{e}}_\alpha$, $|\mathbf{R}_i^k|$ is the distance to atom index i, σ_k is the Gaussian width parameter for type k, and μ_k is a parameter modifying the "penetration depth" of type k. The functions are normalized such that the total integral over all 3 axes and types is 1, the normalization factor is the following.

$$\sum_{\alpha,k} \int_{-R_{cut}}^{R_{cut}} f_\alpha^k \, \mathrm{d}q_\alpha = \sum_k \sum_{i=1}^N exp\left(-\frac{|\mathbf{R}_i^k|}{\mu_k}\right) \tag{6}$$

For the evaluation of condition in Eq. (3), the terms on both sides of the equivalence relation to be tested get a set of descriptor functions as given by Eq. (5). The term on the left gets a set call it $f = \{f_\alpha^k\}$, and the term on the right gets a set call it $g = \{g_\alpha^k\}$. The sets f and g are then compared by means of computing the overlap between the corresponding functions as Eq. (7), the final overlap value $\mathcal{O}$

being the sum of overlaps over all axes α and types k.

$$\mathcal{O} = \sum_{\alpha,k} \int_{-R_{cut}}^{R_{cut}} \min\left(f_\alpha^k(q_\alpha),\ g_\alpha^k(q_\alpha) \right) \mathrm{d}q_\alpha \tag{7}$$

Since the functions f and g are normalized to 1, the maximum value the overlap $\mathcal{O}$ in Eq. (7) can obtain is 1.0, which is the case when the two configurations are exactly identical. The minimum value is 0.0, which is the case when the two configurations are completely different. The parameters σ_k and μ_k can be used to tune the "flexibility" of the equivalence relation, the width σ_k to account for small-scale distortions in positions, and depth μ_k to tune relative importance of atomic types. Then a threshold can be set for the overlap $\mathcal{O}$ value, for example any configuration together with a basis that gives $\mathcal{O} \geq 0.9$ is considered *close enough* for an event of which initial state it is compared to.

Another way that is currently being tested for evaluation of the condition in Eq. (3), is a philosophy borrowed from methods of 3D shape registration [1]. Having in mind that at this point of the algorithm the configuration sets $\mathbf{R}^{sys}$ and $\mathbf{R}_{ini}^{ref}$ are written in bases that could potentially give the same description of geometry, one can assume that the Euclidean distance between a point from the set $\mathbf{R}^{sys} = S$ and the corresponding point in the set $\mathbf{R}_{ini}^{ref} = M$ should be within some very small threshold. The task is to find a pair (s_i, m_j), such that each point s_i in $S = \mathbf{R}^{sys}$ has a corresponding point m_j in $M = \mathbf{R}_{ini}^{ref}$ which gives the smallest (zero) Euclidean distance r from s_i to m_j.

$$\min_{m_j \in M} r(s_i, m_j),\ \forall i\ :\ s_i \in S\ \Rightarrow\ (s_i, m_j) \tag{8}$$

Effectively it means finding a matching for indices i and j in which the set elements correspond index-by-index. The result of the operation is a set of distances corresponding to pairs (s_i, m_j), which can be used to evaluate another threshold, as follows. If any of these distances is larger than some highest allowed threshold, the current basis is rejected as candidate. This corresponds to the situation where one (or more) atoms are displaced from its proper position by more than a threshold. An analogous definition is that of Hausdorff distance $H(A, B)$, which is defined as distance between sets A and B as follows

$$H(A, B) = \max(\ h(A, B),\ h(B, A)\ ) \tag{9}$$

where

$$h(A, B) = \sup_{\alpha \in A} \inf_{\beta \in B} \mathrm{d}(\alpha, \beta) \tag{10}$$

is the *maximal shortest distance* between points α and β.

Once the condition of Eq. (3) is evaluated in a satisfactory manner, the algorithm makes a decision based on this value wether a move can possibly be realised on this site or not. The details regarding event and site are kept in memory until all possible sites are checked for all possible events. Finally a decision for a move is made by the kMC kernel. The move is executed following Eq. (4).

References

[1] P. J. BESL AND N. D. MCKAY, *A method for registration of 3-d shapes*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 14 (1992), pp. 239–256.

[2] B. D. MCKAY AND A. PIPERNO, *Practical graph isomorphism, {II}*, Journal of Symbolic Computation, 60 (2014), pp. 94 – 112.