PSL UNIVERSITÉ PARIS

THÈSE DE DOCTORAT

DE L'UNIVERSITÉ PSL

Préparée à Chimie ParisTech

Développements méthodologiques pour la simulation de l'adsorption des gaz simples dans les matériaux nanoporeux

Methodology developments for the simulation of simple gas adsorption in nanoporous materials

Soutenue par

Lionel ZOUBRITZKY

Le 15/10/2024

École doctorale n°388

Chimie Physique et Chimie Analytique de Paris Centre

Spécialité

Chimie Physique

Composition du jury :

Caroline MELLOT-DRAZNIEKS CNRS, Collège de France	*Présidente du jury*
Emeric BOURASSEAU CEA Cadarache	*Rapporteur*
Tina DÜREN University of Bath	*Rapporteuse*
Pluton PULLUMBI Air Liquide	*Membre du jury*
François-Xavier COUDERT CNRS, Chimie ParisTech – PSL	*Directeur de thèse*

Chimie Paris ParisTech | PSL

REMERCIEMENTS

TABLE OF CONTENTS

Notations and abbreviations

$\boldsymbol{r}$	triplet of coordinates in real space.
$\boldsymbol{k}$	triplet of coordinates in Fourier space.
$\boldsymbol{p}$	configuration of a system.
$\mathcal{X}$	system or sub-system.
$\|\boldsymbol{x}\|$	Euclidean norm of vector $\boldsymbol{x}$.
$\|\boldsymbol{x}\|_\mathrm{p}$	periodic distance, *i.e.* the smallest Euclidean norm of a periodic image of vector $\boldsymbol{x}$.
pcu	topology from the RCSR.
FAU	topology from IZA-SC.
MC	Monte-Carlo
GCMC	Grand-Canonical Monte-Carlo

1

TOPOLOGY OF CRYSTALLINE MATERIALS

1.1 Notion of topology

1.1.1 Definition

In order to describe the average structure of crystalline materials at the nano-scale, reticular chemistry[1,2] proposes to classify crystals, and in particular framework materials, in light of their topology. This approach consists in abstractly regrouping the atoms of a crystal into clusters, called *vertices*, bonded together by an abstract kind of linkers, called *edges* — the choice of grouping being informed by chemical sense. In simple compounds, such as dense oxides, the mapping can be trivial, with each atom as a separate vertex and each chemical bond as an edge. For zeolites and aluminosilicates, the tetrahedral atoms T are considered the vertices (T = Al, Si, P, …) and the T–O–T linkages are the edges. In the case of metal–organic frameworks (MOFs), the vertices would be chosen among the *secondary building units* (SBUs)[3] with more than 2 points of extension, and the edges would correspond to bonds between those as well as ditopic SBUs, for example. Any crystal can then be simplified into a topological structure, called a *net*, consisting of the sets of vertices and edges (see section 1.1).

Like the atomic structure it represents, the net is 3-dimensional and periodic. It contains all the information on the connectivity of the vertices, *i.e.* which pairs of vertices, called *neighbours*, are linked by an edge. Thus, it provides a convenient mathematical abstraction to classify crystal structures. However, it is important to note that this representation is dependent on the choice of vertices and therefore a crystal may be represented by different nets.[8] Moreover, reconstructing the full crystal from the net necessitates additional data, such as the unit cell parameters and the nature and exact geometric placement of the vertices and the edges.[5] A crystal may also be composed of multiple interpenetrating nets,[9] which is common in ultraporous framework materials.

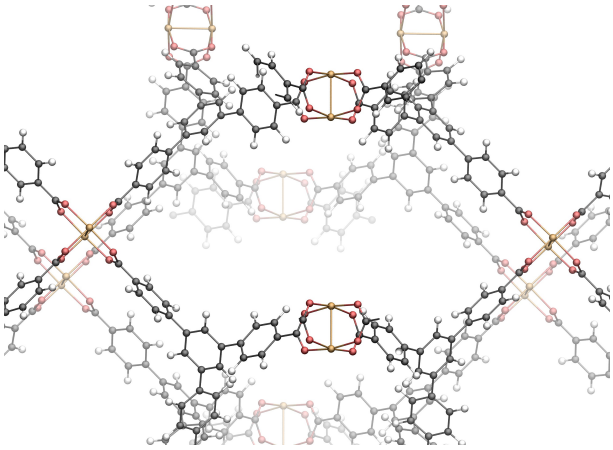
1.1.2 Existing resources

Several existing projects, such as the RCSR[7] or Epinet,[10,11] aim at collecting the different nets underlying the structure of real or hypothetical crystals. This is key to allowing researchers to create new materials with a target topology, which is useful to obtain specific physical properties, given the strong link between materials topology and properties for a given chemical composition.[12] However, in order to check the accuracy of the collected nets with regard to existing crystals, one needs a way to determine the net from a given crystal structure.

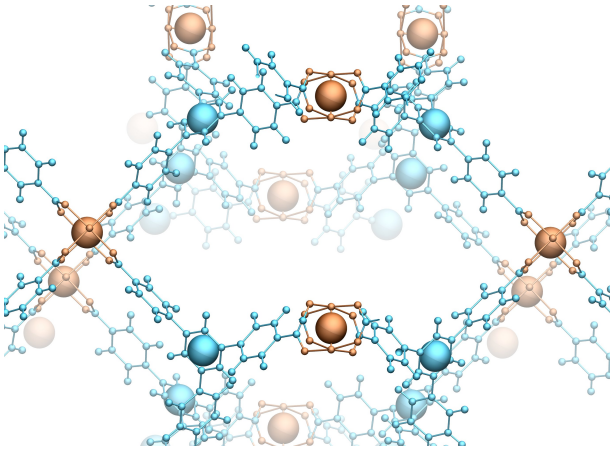
There are two different approaches to solve this problem, which correspond to the two main softwares used in the field: ToposPro[13,14] and Systre.[15] ToposPro is a crystallographic tool that allows to retrieve an extensive set of topological properties from crystals. It is a proprietary software, but its topology detection tool can be accessed for free online[16][1] for crystallographic files in CIF format.[17] Its strategy to identify the topology of a net consists in computing three specific properties for each vertex u:

- The coordination sequence up to distance 10. A vertex v is at distance k of u is there is a chain of $k+1$ vertices $u = u_0, u_1, \ldots, u_k = v$ such that for all i in $[\![1, k]\!]$ (where $[\![a, b]\!]$ designates the set of integers between a and b), u_{i-1} and u_i are neighbours, and there is

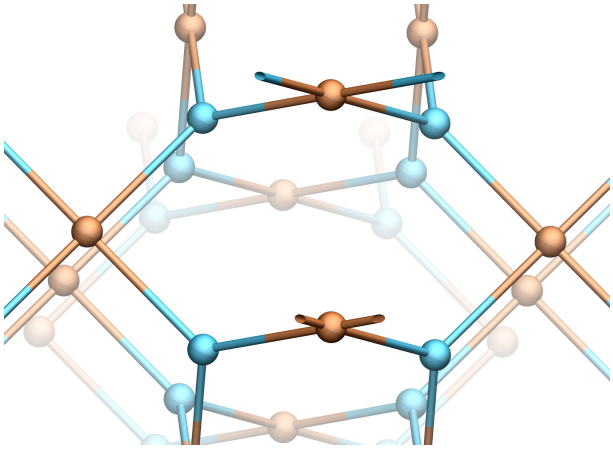
[1] available online at https://topcryst.com/



(a)



(b)



(c)

Figure 1.1: Decomposition of a crystalline framework into its underlying net:
(a) Crystalline structure of MOF-143.[4]
(b) Identification of the SBUs: Cu-based paddle-wheel[5] *and benzene-tribenzoate (BTB).*[6]
(c) The underlying net, called ***pto*** *in the database of the RCSR.*[7]

no smaller such chain. The k-th term of the coordination sequence of u is then defined as the number of vertices at distance k of u.

- The point symbol.
- The vertex symbol.

The latter two account for the shortest cycles and rings starting at vertex u.[18,19] It is found empirically that the combination of these three properties for all vertices forms an almost unique identifier of the topology. However, this approach is not completely sound as there can be different nets that do share these same three properties.

Systre is an open-source software[2]; its inputs are files representing the graph underlying the crystal structure, rather than the crystallographic files directly. It implements an algorithm that yields a so-called *topological genome*[20] of the net, which is a finite series of numbers. This genome is provably unique for each net, only depends on the net itself and not its representation, and can be computed in polynomial time of the size of the net. To do so, Systre starts by computing an *equilibrium placement*[21] of the vertices, which attributes to each vertex a position which is the barycentre of that of its neighbours. The net is then minimised, which amounts to finding a unit cell. Afterwards, Systre collects a number of *candidates*, which are frames of reference that do not depend on the current representation of the net. For each candidate, the vertices of the net are ordered according to their position in the frame of reference, and a key is determined which uniquely identifies the net with this ordering in this frame. The genome is then chosen as the lexicographically smallest key. Since the candidates do not depend on the initial representation of the graph, the keys do not either, hence the genome is indeed uniquely determined and only depends on the net itself. In graph theory, the problem Systre solves is called *graph canonization*. It incidentally allows determining whether two input graphs are actually two representations of the same graph (the *graph isomorphism* problem) by comparing their canonical forms (here, the genome), but it can also be used to give a canonical name to graphs for example, by accessing a database of names with the canonical form of the graphs.

Apart from ToposPro and Systre, a few alternative tools have been developed to simplify structures into topological descriptors, but none are suitable for topology identification of chemical structures at large. Indeed, they either only work on specific structures (usually zeolites) like TOTOPOL,[22] or rely on a non-unique representation of the structure (such as a tiling[23]) like the T-ring representation,[24] or because their output is not unique for each input structure, like the Single Repeating Unit representation[25] for example. For finite molecules, the two most common representations are the SMILES[26] file format, which is proprietary and comes with several conflicting implementations, or the IUPAC's International Chemical Identifier (InChi)[27] key, which is open-source and more standardized, but less human-readable. Both include additional chemical information as well.

In this context, we propose a new tool to efficiently determine the topology of crystal structures: the Julia package CrystalNets.jl[3]. Julia[28] is a modern general-purpose and scientific programming language that offers both great performance and ease of programmability.[29] This, as well

[2]available online at http://gavrog.org/
[3]available online at https://github.com/coudertlab/CrystalNets.jl

as carefully chosen optimisations, allows CrystalNets.jl to outperform Systre while leveraging a very similar core algorithm to ensure soundness. Moreover, the package uses the Julia port of chemfiles[4] which allows it to read common chemical file formats like CIF, making it easy to use for the analysis of single cases as well as large databases of crystal structures.

Delgado-Friedrichs and O'Keeffe[15] explained the overall architecture of the Systre algorithm which CrystalNets.jl largely reuses: we will follow its structure, only briefly summarising the key concepts that are already explained elsewhere along the way, while focusing on the implementation choices and some proofs of correctness that do not appear in other publications. We will then focus on a series of new optimisations of the algorithm that allow to greatly improve its performance for most nets. Finally, we will demonstrate the applicability of the method by analysing several existing crystalline databases using CrystalNets.jl.

1.2 Algorithms and implementation

1.2.1 From the chemical structure to the net

Bonding algorithm

Since the topological information is derived from the bond network, using correct bonds is a necessary prerequisite to any topology recognition tool. Yet, identifying bonds in a crystal is not a clear-cut science. Following the chemical meaning of a bond, the map of electronic density in the system should theoretically be necessary to establish the bond network of the crystal. A bond would then be detected when the density reaches a certain threshold along the interatomic axis, and the nature of the bond (simple, double, triple) can be derived from the overlap of atomic orbitals for instance. However, using this minute vision of the chemical structure, the value of the threshold results from an arbitrary decision.

The fundamental reason why attributing bonds is cannot be made absolute results from the fact that chemical bonds are no more than a useful convention, rather than a measurable observable. Yet, this convention is not without meaning, otherwise any notion built from it, like the topology in our case, would lack sense. Therefore, several bond attributing strategies have been devised, none being proven as the ultimate one.

In the case of CrystalNets.jl, whenever the bonds are not explicitly given, the automatic bonding algorithm starts from the heuristic strategy used by Chemfiles and VMD. First, the smallest distance between all pairs of atoms is computed, "smallest" being taken among the translated images of each atom across unit cells. Any distance lower than 0.5 Å is considered an input error. Otherwise, two atoms are bonded if their distance is below the minimum of two threshold: the first one is proportional to the maximum Van der Waals radius of all atoms in the structure, and the second is proportional to the sum of Van der Waals radii of the two atoms. The Van der Waals radii were taken from the Blue Obelisk's data repository and slightly modified for a few atoms as in Chemfiles.

From this point, our algorithm departs from that of Chemfiles and VMD by using multiple additional heuristics. First, the two cutoff values mentioned before are controlled by an option

[4] available online at https://chemfiles.org/

that can be tweaked by the user, to remove some spurious bonds or detect some which are just past the threshold. Additionally, if the structure is a MOF or a zeolite (which is set by the user), then the Van der Waals radius of all metallic atoms is multiplied by 1.5, to account for the larger bonds used in the description of inorganic SBUs.

A number of spurious bonds are then removed from the initial guess resulting from these rules. Some common atoms have fixed maximum valence : hydrogen is limited to 1 neighbour, carbon and nitrogen to a maximum of 4, oxygen to 2 unless it has metallic neighbours, in which case it can have up to 4 neighbours (a typical use case is for metal-oxo clusters). Bonds shorter than 0.65 Å or longer than 4 Å are removed, as well as those longer than 3 Å that are significantly out-of-plane with respect to other bonds around any of the two neighbours. Metal-metal bonds are conditionally removed according to user options. Finally, in a configuration where three atoms are pairwise bonded in a triangle, the longest of the three bonds is removed if it is either too long or too close to the opposite atom.

All these heuristics were derived from manual inspection of the result of the bonding algorithm over databases of crystal structures. As with any strategy relying on heuristics, it cannot be made perfect, but it appears to be reasonable enough for all cases encountered so far. Overall, this strategy works well with respect to the time vs precision trade-off required by CrystalNets.jl ; any input without explicit bond which requires this algorithm will still result in a warning though, to let the user know that the automatically extracted bond network should be carefully checked.

It should be mentioned that some bonding strategies are out of scope of this algorithm, such as the detection of hydrogen bonds, typically. Studying the topology resulting from such a bond network thus requires passing explicit bonds in the input file.

1.2.2 Clustering algorithms

The last transformation that the bond network undergoes before yielding a net is clustering, which consists in grouping together sets of atoms: each such cluster will then be represented as a single vertex in the net. This transformation is useful in order to simplify the exact atomistic topology into a higher-level topology that can expose the overall organisation of the crystal, without reproducing the low-level details.

We provide five main clustering options: “all nodes”, “single nodes”, “standard”, “PE” and “PEM”. Except for the “standard” representation which is uniquely defined by its implementation in ToposPro,[30,31] the other clustering options are only described but not uniquely determined by an explicit algorithm.[8,30,32–35] We also chose not to follow the original implementation of the “standard” representation,[31] primarily because of the required computational cost of determining the minimal ring around each bond. Instead we use a custom algorithm whose main steps are as follows:

- First, organic cycles are replaced by a new virtual carbon atom. Afterwards, all metallic atoms are assigned to the “inorganic” class and all carbon atoms to the “organic” class. Then, in a loop, each remaining atom that is bonded to an atom in the “inorganic” class is itself classified as “inorganic”, until no such atom remains. All remaining unassigned atoms are then classified as “organic”. P and S atoms are treated differently and considered

organic if directly bonded to a carbon, metallic if directly bonded to a metal, and part of a third class otherwise.

- Afterwards, SBUs are formed by regrouping together all adjacent atoms belonging to the same class. Paddle-wheel patterns are identified separately and regrouped into a single inorganic SBU even when the two opposite metal centres are not bonded together. The *points of extension*, which are organic atoms bonded to an inorganic atom, form their own SBU each. The algorithm stops here if there is no periodic SBU, *i.e.* if there is no infinite set of adjacent atoms belonging to the same class.

  Otherwise, if there are such periodic SBUs, they are split by virtually removing the atoms with the highest connectivity and regrouping the other atoms of the periodic SBU into the remaining connected components. A few other heuristics are used to ensure that all final SBUs are finite, and to tackle the various possible configurations encountered in framework structures.

This algorithm yields our "all nodes" representation.[32] It can be converted to "PE" (standing for "points of extension")[36,37] by removing all metallic atoms and bonding their corresponding points of extension to describe the coordination polyhedra around the removed metals. The "all nodes" representation is also convertible to "single nodes"[32] simply by merging the organic points of extension with their neighbouring organic SBU. It can also be transformed to yield the "PEM" (for "points of extension and metals") representation[35,36] by splitting inorganic clusters to keep only one metal atom per SBU. Combining the algorithm for "PEM" and "single nodes" yields our "standard" representation.[30] For simple cases, we also provide an "each vertex" clustering option which bypasses the entire clustering step and keeps each atom as its own SBU, as well as an "input" option which determines clusters directly from the `Residues` of the chemfiles-parsed input file, so that the user may provide their own SBUs.

Once the SBUs are determined, they are converted to vertices and bonded together according to the atomic bonds. Finally, as with all nets in CrystalNets.jl, if a vertex has one or zero neighbour, it is removed, and if it has exactly two neighbours it is converted into an edge between its neighbours, and this iteratively until convergence. Once vertices and edges are defined, interpenetrating nets are separated, each of them accounting for a single structure.

All decompositions start by separating metals, on the one hand, and organic carbons, on the other hand. Aromatic cycles are merged into a single pseudo-atom. The "Single Node" strategy consists in growing the metallic clusters, then the organic clusters, with adjacent non-attributed atoms until all have an attributed group. The "All Nodes" additionally puts the points-of-extension, which are the organic atoms bonded to an inorganic cluster, into their own separate group. Alternatively, each metal atom can be put into its own separate group, yielding the "Standard" strategy. These three methods have been standardized by IUPAC.

In addition to these three main options, the "Points-of-Extension and Metals" (PE&M) algorithm proposed by [REF] consists in retaining the points-of-extension from the initial clustering, but discarding those clusters to only keep one per metal atom and one per point-of-extension. Finally, the "Points-of-Extension" (PE) strategy only retains one cluster per point-of-extension.

CrystalNets.jl also offers the "Each Vertex" option to take one cluster per atom, and the "Input" option to make one cluster per residue of the input file, as can be defined in PDB files for instance, to give the user maximum control on the clustering.

Overall, the net which will be identified by CrystalNets.jl heavily depends on a series of choices made when deciding on the bond attribution, as well as the clustering step. Automated tools can only guess so far as to the expected decomposition of the crystal, hence the importance of checking the alignment of the detected net with the expected outcome.

1.2.3 Periodic graph structure

Crystal nets are special instances of 3-periodic graphs. In general, a *graph* is a mathematical construct that contains objects, called *vertices*, some of which are linked together by *edges*. Two linked vertices are said to be *neighbours*. In a crystal, the pattern representing the atoms and their chemical bonds periodically repeats itself across three dimensions: for this reason, the graph representing a crystal is called 3-periodic.

More generally an N-periodic graph (with $\mathrm{N} \in \mathbb{N}^*$) can be represented using the vector model.[38] This model conceptually follows the description of a crystal as a lattice and a pattern. Any point in the lattice is denoted by an offset $o \in \mathbb{Z}^\mathrm{N}$, whose i-th coordinate is the offset along the i-th axis of the lattice compared to a fixed origin. An N-periodic graph is then defined by a pair (V, E) where V is the set of vertex identifiers and $\mathrm{E} \subset \mathrm{V} \times \mathrm{V} \times \mathbb{Z}^\mathrm{N}$ is the set of edges. Any vertex in space is uniquely identified by a pair $(v, o) \in \mathrm{V} \times \mathbb{Z}^\mathrm{N}$ where $v \in \mathrm{V}$ is its vertex identifier and $o \in \mathbb{Z}^\mathrm{N}$ is the offset of the cell that contains the vertex. Let's note $n = |\mathrm{V}|$ the number of vertices per cell and identify V with the integer interval $[\![1, n]\!]$ so that each vertex identifier can be considered to be a number in $[\![1, n]\!]$. When working in a fixed cell, the vertices will simply be referred to by their vertex identifier. Finally, for most of the rest of this article N will be equal to 3 since nets are 3-dimensional, but the algorithms exposed in the rest of this section actually work for any dimension and CrystalNets.jl can identify 2-periodic and 1-periodic graphs as well.

An edge $e = (u, v, o)$ can be broken down into a source $u \in \mathrm{V}$, and a destination $(v, o) \in \mathrm{V} \times \mathbb{Z}^3$. The source can be seen as the vertex of identifier u and positioned in the cell at the origin of the lattice, and the destination is the vertex to which it is linked. When the source and the destination are in the same cell, the offset o is zero and noted $0_{\mathbb{Z}^3} = (0, 0, 0)$. Edges of the form $(u, u, 0_{\mathbb{Z}^3})$, called *loops*, are absent in nets because they do not convey any structural information.

For any edge $e = (u, v, o) \in \mathrm{E}$, the reverse edge $(v, u, -o)$ is also in E: this property makes the graph *undirected*. All graphs used to represent crystals are undirected because chemical bonds are symmetric: if atom A is bonded to atom B, then atom B is bonded to atom A, and likewise for clusters of atoms.

As opposed to periodic graphs, *simple* graphs do not have a concept of offset: an edge of a simple graph is only a pair of vertices. Numerous computational representations of simple graphs exist in the literature, but they cannot be directly transposed to periodic graphs. In order to find a computational representation for crystals relevant to topological analyses, a useful observation is that nets share two distinctive features that come from chemistry:

- The number of edges is at most proportional to the number of vertices, because of the limited valence of atoms. Such graphs are generally called *sparse*. This remains true when vertices represent clusters instead of single atoms, although in that case the number of neighbours can be higher than the maximum valence of an atom — for instance, SBUs with up to 66 points of extension have been built..[3]
- The set of edges is small to medium-sized because the number of vertices per unit cell is, in practice, between 1 and 10,000 in most real-world crystalline structures. This is small compared to the trillion of edges of some graphs in social media modelling for instance.[39]

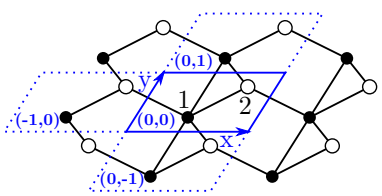
Finally, the most important operation on graphs that will subsequently be used is finding the set of neighbours of a given vertex. Driven by this, we chose to computationally represent a periodic graph by a list of size n whose u-th element is the set of neighbours of u: $\mathrm{N}_u = \left\{(v, o) \in \mathrm{V} \times \mathbb{Z}^3 \;\middle|\; (u, v, o) \in \mathrm{E}\right\}$. N_u is stored as a lexicographically sorted list.

Because of the previously mentioned undirectedness of the graph, if $(v, o) \in \mathrm{N}_u$, then $(u, -o) \in \mathrm{N}_v$. However, it is often convenient to be able to enumerate the edges of the graph without double counting an edge and its reverse. For this reason, the representation comprises another list of size n whose u-th element is the first index i in N_u such that $\mathrm{N}_u[i] = (v, o)$ with either $v > u$ or $\left[v = u \text{ and } o > 0_{\mathbb{Z}^3}\right]$. Such an edge (u, v, o) will be called a *direct* edge. Since loops are forbidden, all edges are either a direct edge, or the reverse of a direct edge. Thus, since each list of neighbours is lexicographically ordered, reading the list of neighbours starting at the given indices i is guaranteed to yield all direct edges of the graph, hence each edge exactly once without its reverse. The lexicographically ordered sequence of direct edges is a unique identifier of a representation (V, E) of a periodic graph in the vector model: such a sequence will be called the *key* associated with the representation.

We implemented the graph structure detailed above in a companion Julia package, PeriodicGraphs.jl, which can be used independently of CrystalNets.jl.

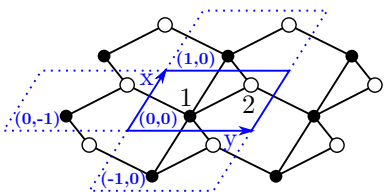
section 1.2 shows several representations of the same 2-periodic graph. The graph is made of two kinds of vertices (represented by open and closed circles), but the choice of the vertex identifiers and of the cell influences the representation. Each choice of cell is characterised by its origin and its axes x and y; some repeated cells are represented as well, along with their offset in the bottom-left corner. For each representation, the key is written below, each line referring to a direct edge. Recall that the Systre algorithm works by choosing the lexicographically smallest key from a set of representations, extracted from candidates that do not depend on the representation of the graph. Hence, if the set of representations was those shown in the figure, the topological genome of the graph would be the smallest key, *i.e.* that under representation (a).

Having a working representation for nets, let's now discuss our reimplementation of the Systre algorithm[15] to compute a unique identifier for each net topology.



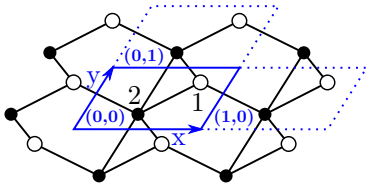
(a) Reference unit cell

1	1	0	1
1	2	-1	0
1	2	0	-1
1	2	0	0



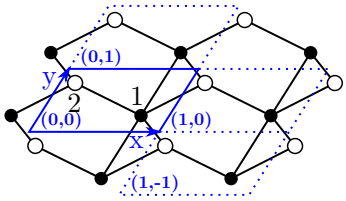
(b) Inversion of axes

1	1	1	0
1	2	-1	0
1	2	0	-1
1	2	0	0



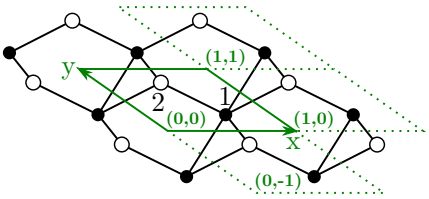
(c) Renumbering of vertices

1	2	0	0
1	2	0	1
1	2	1	0
2	2	0	1



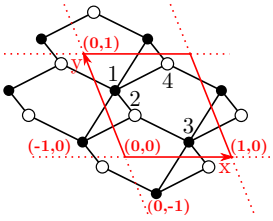
(d) Change of origin

1	1	0	1
1	2	0	0
1	2	1	-1
1	2	1	0



(e) Other unit cell

1	1	1	1
1	2	0	-1
1	2	0	0
1	2	1	0



(f) Non-unit cell

1	2	0	0
1	3	-1	0
1	3	0	1
1	4	-1	0
1	4	0	0
2	3	-1	0
2	3	0	0
3	4	0	-1

Figure 1.2: Six different representations of the same 2-periodic graph. The first two columns are the source and destination vertices, the last two are the edge offset.

1.2.4 Equilibrium placement

Definition

Given a periodic graph, the first step is to compute, for each vertex $(u, o') \in \mathrm{V} \times \mathbb{Z}^3$, a position $p(u, o') \in \mathbb{R}^3$ such that

$$\forall u \in \mathrm{V}, o' \in \mathbb{Z}^3,\ p(u, o') = \frac{1}{|\mathrm{N}_u|} \sum_{(v,o)\in \mathrm{N}_u} p(v, o' + o) \tag{1.1}$$

The set of positions p, called an *equilibrium placement*, is such that each vertex is at the barycentre of all its neighbours. p must also satisfy the additional constraint that

$$\forall u \in \mathrm{V},\ o \in \mathbb{Z}^3,\ p(u, o) = p(u, 0_{\mathbb{Z}^3}) + o \tag{1.2}$$

to enforce the periodicity of the equilibrium placement.

The equilibrium placement has been proven to be unique modulo isometries, because it is the solution of a full-rank linear problem.[21] Equation 1.1 can be rewritten with Equation 1.2 in matrix form, $\mathrm{A} \times \mathrm{P} = \mathrm{O}$, where $\mathrm{A} \in \mathcal{M}_{n,n}(\mathbb{Z})$, $\mathrm{O} \in \mathcal{M}_{n,3}(\mathbb{Z})$ and unknown $\mathrm{P} \in \mathcal{M}_{n,3}(\mathbb{R})$ such that, noting $n(u, v)$ the number of neighbors of u whose identifier is v:

- $\forall (u, v) \in \mathrm{V}^2,\ \mathrm{A}_{u,v} = \begin{cases} |\mathrm{N}_u| - n(u, u) & \text{if } u = v \\ -n(u, v) & \text{otherwise} \end{cases}$

- $\forall u \in \mathrm{V},\ \mathrm{O}_u = \sum_{(v,o)\in \mathrm{N}_u} o$

and from which we determine for all $u \in \mathrm{V}$, $p\left(u, 0_{\mathbb{Z}^3}\right) = \mathrm{P}_u$.

Since the coefficient of A and O are integers, the computed solution P is always in $\mathrm{M}_{n,3}(\mathbb{Q})$ with the additional constraint that one of the vertices u be placed at the origin. The exact solution with coefficients in $\mathbb{Q}$ is actually necessary for the subsequent computations and not an approximation, so the resolution of the system must be exact. This turns out to be a very costly operation: it requires storing integers of arbitrary size, which are computationally far more expensive than fixed-width machine integers.

Computational aspects

The GNU Multiple Precision Arithmetic Library (GMP)[40] provides access to heavily optimised structures that allow to efficiently perform exact integer and rational computations. Julia has native support for the arbitrarily-sized integer structure from GMP, through the `BigInt` type, but does not provide a full interface for GMP's rational numbers (although it is internally used for Julia's native `Rational{BigInt}` type). As part of this work, we provided this special support in a separate package, BigRationals.jl, which exports the `BigRational` type wrapping GMP's rational type.

In addition to the data structure used, a choice has to be made about the algorithm for solving this linear system. The most straightforward idea would be a general linear algebra technique such as LU decomposition followed by forward and backward substitution. Yet, matrix A is structured since it is both symmetric and sparse, calling for the use of more efficient algorithms.

Several algorithms have been discussed in the literature for the exact resolution of sparse and integer linear systems.[41–44] Following the analysis by Eberly *et al.*,[41] we decided to implement Dixon's algorithm[43] which relies on a modulo-prime inversion of the matrix. In accordance with their conclusion, this algorithm largely outperforms a sparse LU decomposition algorithm (which we implemented as a reference and for benchmarking), both algorithms leveraging GMP instructions specific to the `BigRational` type for improved performance. Yet, Dixon's algorithm can fail if matrix A happens not to be invertible modulo the chosen prime: hence, we chose to successively invert it modulo three hard-coded large primes and stop at the first succeeding one, and finally resort to LU decomposition if all three fail. This implementation is up to orders of magnitude faster than that used in the original Systre program written in Java. This extra performance is crucial for the use of the topology detection algorithm as part of large-scale screening methodologies.

The equilibrium placement naturally embeds the graph in a Euclidean space, namely $\mathbb{Q}^3$. The first vertex of the graph is placed in position $(0,0,0) = 0_{\mathbb{Q}^3}$, which is allowed because any translation of the equilibrium placement still results in an equilibrium placement. Moreover, because of Equation 1.2, for any vertex identifier $u \in \mathrm{V}$ there is an offset o such that $p(u,o) \in [0,1[^3$, where $[a,b[$ designates the set of reals between a (included) and b (excluded). In order to simplify the representation, this offset is set to $0_{\mathbb{Z}^3}$ by adjusting the offsets of the edges of the graph accordingly. In this new representation, equations 1.1 and 1.2 still hold, as well as the new property:

$$\forall u \in \mathrm{V},\ p(u) \in \left(\mathbb{Q} \cap [0,1[\right)^3 \quad \text{where } p(u) = p(u, 0_{\mathbb{Z}^3}) \tag{1.3}$$

In order to accelerate the subsequent algorithms, the vertices of the graph are lexicographically sorted according to their new position. The first vertex, being at $0_{\mathbb{Q}^3}$, remains first.

Nets where at least two vertices have the same position are called *unstable* and, similarly to Systre, are generally not handled by our implementation. This is usually not problematic since their probability of occurrence is low.[45] Our implementation can however handle unstable nets where each collision site comprises at most four vertices if there is no edge between two different collision sites and no collision site is bonded to two representatives of the same vertex. This accounts for roughly half of the encountered unstable nets in the databases we explored. We will not detail the case of these unstable nets here but focus on stable ones. We can thus assume that no two vertices have the same position.

section 1.3 represents the equilibrium placement for the net of diamond (called **dia** in the RCSR). Note that the natural embedding in $\mathbb{Q}^3$ imposes a cubic unit cell for all nets, which is not representative of the actual unit cell of diamond; however, the geometry of the unit cell is irrelevant here because only the topology of the net matters. As a consequence, the bond lengths are not representative of that of diamond either, which explains why not all have the same lengths in this representation. Nonetheless, each vertex is indeed at the barycentre of the positions of its neighbours.

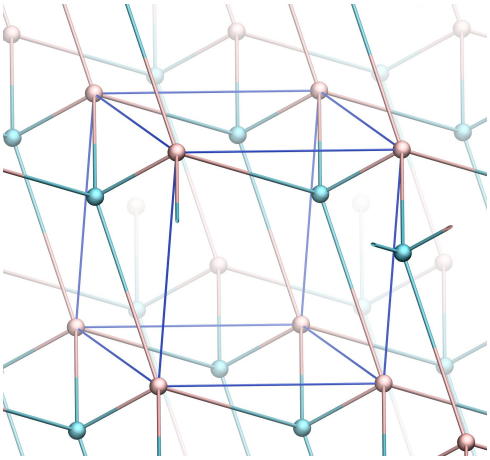


Figure 1.3: Natural Euclidean embedding of the equilibrium placement for the **dia** *net.*

1.2.5 Minimisation of the graph

Once the equilibrium placement is found, the next step of the algorithm is to find a unit cell. In general, a *cell* is a parallelepiped whose three axes $(\vec{a}, \vec{b}, \vec{c})$ are linearly independent translations that constitute symmetries of the graph. Such translations will be called *valid translations*. Another representation for this cell is the matrix A whose columns are respectively $\vec{a}$, $\vec{b}$ and $\vec{c}$. Its volume is $\det(A) = (\vec{a} \wedge \vec{b}) \cdot \vec{c}$. It is a *unit cell* if it contains the smallest possible number of vertices, which is equivalent to having a minimal volume.

At this point, it is important to recall that the Euclidean space in which the graph is embedded through the equilibrium placement depends on its initial representation, so there is no simple way of finding a unique unit cell independently of the original representation. In particular, common cell reduction techniques such as Delaunay[46] or Niggli,[47,48] which yield a uniquely defined cell, do not solve this problem. Unfortunately, existing libraries implementing these

techniques (like spglib[49] or cctbx[50]) cannot be used either for the simpler problem of finding any unit cell because they do not handle arbitrary precision.

First, note that, by construction of the equilibrium placement, the initial cell is the canonical basis of $\mathbb{Q}^3$, so its matrix is the identity I. Let's now assume that I is not a unit cell and there is a unit cell $A \in \mathbb{Q}^{3\times3}$ of volume $\det(A) \in]0,1[$.

Let's define $f : \mathbb{Q} \to \mathbb{Q} \cap [0,1[\,, \frac{p}{q} \mapsto \frac{p\%q}{q}$ where $p\%q$ is the remainder of the euclidean division of p by q (hence $p\%q \in [\![0, q-1]\!]$). Let $B = \left(f(A_{i,j})\right)_{1\leq i,j\leq 3} \in \mathbb{Q}^{3\times3}$. By definition of f, the columns of B are axes within the initial cell; moreover, since $\forall x \in \mathbb{Q},\ x - f(x) \in \mathbb{Z}$, each axis of B can be seen as the corresponding axis of A plus an offset $o \in \mathbb{Z}^3$. But since I is a repeating unit, such an offset is a valid translation, so each axis of B is a valid translation. Finally, since $\det(A) \notin \mathbb{Z}$, not all coefficients of A can be integers, so A has at least one coefficient x such that $f(x) \neq 0$, so at least one axis of B is not $0_{\mathbb{Q}^3}$.

To summarise, the previous observations show that if the initial cell was not a unit cell, then there exists a non-zero valid translation $t \in \left(\mathbb{Q} \cap [0,1[\right)^3 \setminus \{0_{\mathbb{Q}^3}\}$.

Let's use this fact to find a unit cell by enumerating all such translations. Symmetries map vertices to vertices, so the first vertex (lexicographically sorted by position) is mapped to another. But since the first vertex was put in position $0_{\mathbb{Q}^3}$, the translation is directly given by the position of the other vertex. Hence, only $n-1$ translations need to be checked, and if none is valid then the initial cell is a unit cell.

Checking whether a translation is valid can be done with $O(n \log n)$ time complexity. Vertices are already sorted by position, which allows to efficiently check that all the vertices are mapped to vertices. Doing so first allows to collect the map from initial vertices to their new counterparts and return early if the translation is invalid. Afterwards, it remains to be checked that all edges are also mapped to edges, which can also be done efficiently since the lists of neighbours N_u where $u \in V$ are also lexicographically sorted.

If I is not a unit cell, then a valid translation $t = (t_1, t_2, t_3) \in \left(\mathbb{Q} \cap [0,1[\right)^3 \setminus \{0_{\mathbb{Q}^3}\}$ must have been found. Let $k \in [\![1,3]\!]$ be such that t_k is the smallest non-zero coefficient of t. Then a new smaller cell is given by matrix $A^{(t)}$ where

$$A^{(t)}_{i,j} = \begin{cases} t_i & \text{if } j = k \\ 1 & \text{if } j \neq k \text{ and } i = j \\ 0 & \text{otherwise} \end{cases} \tag{1.4}$$

The volume of $A^{(t)}$ is $\det(A^{(t)}) = t_k \in]0,1[$ and by constructions its axes are all valid translations, so A is indeed a new smaller cell. This process can then be iterated in the new Euclidean space created by mapping each point $x \in \mathbb{Q}^3$ to $(A^{(t)})^{-1}x$, until no valid translation is left.

There are several ways to improve the computational complexity of this procedure. The first one consists in observing that if $t \in \mathbb{Q}^3$ is a valid translation, then so is $k \times t + o$ where $k \in \mathbb{Z}$ and $o \in \mathbb{Z}^3$. This allows to minimise the volume of the new cell $A^{(t)}$ thanks to algebraic

considerations. Let's illustrate this point in two dimensions for simplicity, although it can be generalised to any number of dimensions.

Let $t = \left(\frac{p_1}{q_1}, \frac{p_2}{q_2}\right)$ be a valid translation. Let's assume that $p_1 \neq 0$ and $q_1 \geq q_2$ (the opposite reasonning can be done in the other case). Let $k \in \mathbb{Z}$ and $k' \in \mathbb{Z}$ be Bézout coefficients for p_1 and q_1, *i.e.* such that $kp_1 + k'q_1 = 1$, and let $o = (k', 0) \in \mathbb{Z}^2$. Then $k \times t + o = \left(\frac{1}{q_1}, \frac{kp_2}{q_2}\right)$.

Hence, matrix $\mathrm{A}^{(kt+o)}$ can replace $\mathrm{A}^{(t)}$ to reduce the cell to a volume $\frac{1}{q_1}$ instead of $\frac{p_1}{q_1}$.

Another potential improvement consists in using all the valid translations instead of stopping at the first one encountered. This way, the translation that minimises the volume of the new cell can be chosen instead of any valid translation. However the cost of collecting all valid translations can be up to n times higher than finding the first valid translation.

Furthermore, $\mathrm{A}^{(t)}$ is just a special case of the set of possible new cells. In general, any invertible matrix whose columns are valid translations (including columns of the identity matrix) is the matrix of a valid cell. The best new cell is the one with the smallest volume, but combinatorial exploration of all possible such matrices that lead to smaller cells has a worst-case complexity of $\mathrm{O}(n^3)$. In practice, many candidate matrices can be eliminated before computing their determinant so using this method is actually quite efficient.

1.2.6 Search space exploration

At this stage in the workflow, the graph has a unit cell and a corresponding equilibrium placement. Let's now look for the topological genome. The strategy follows that implemented in Systre, which consists in taking the lexicographically smallest key among those computed for a number of candidates. Both the computation of the key from a given candidate and the selection of the candidates should not depend on the current representation of the graph.

A candidate is defined as a triplet of linearly independent edges, and represented by a pair {vertex, basis of $\mathbb{Q}^3$} where the vertex is the source of the first edge and the columns of the basis are the translations induced by the edges. The process of selecting candidates is explained in section 1.3. This subsection focuses on the computation of a key from a given candidate.

This precise algorithm is explained in[15] and analysed as Algorithm 5 in.[21] It is detailed in section 1.4. The main point is that the algorithm creates a new representation of the graph which is independent of the previous one by redefining both the unit cell and the numbering of vertices thanks to a unique euclidean space derived from the candidate. The first element of the candidate, the vertex, is the origin of the new space and the columns of the basis are its three axes.

Step **(b)** is a simple affine transformation of the positions and steps **(c)** to **(h)** are straightforward to implement, they only require keeping track of the positions of the numbered vertices. Step **(j)** is also straightforward, it consists in multiplying each translation by the inverse of the matrix of the new unit cell to obtain an integer offset. This allows to retrieve the set of edges E of the graph, and since $\mathrm{V} = [\![1, n]\!]$, this directly yields the representation of a periodic graph in the vector model.

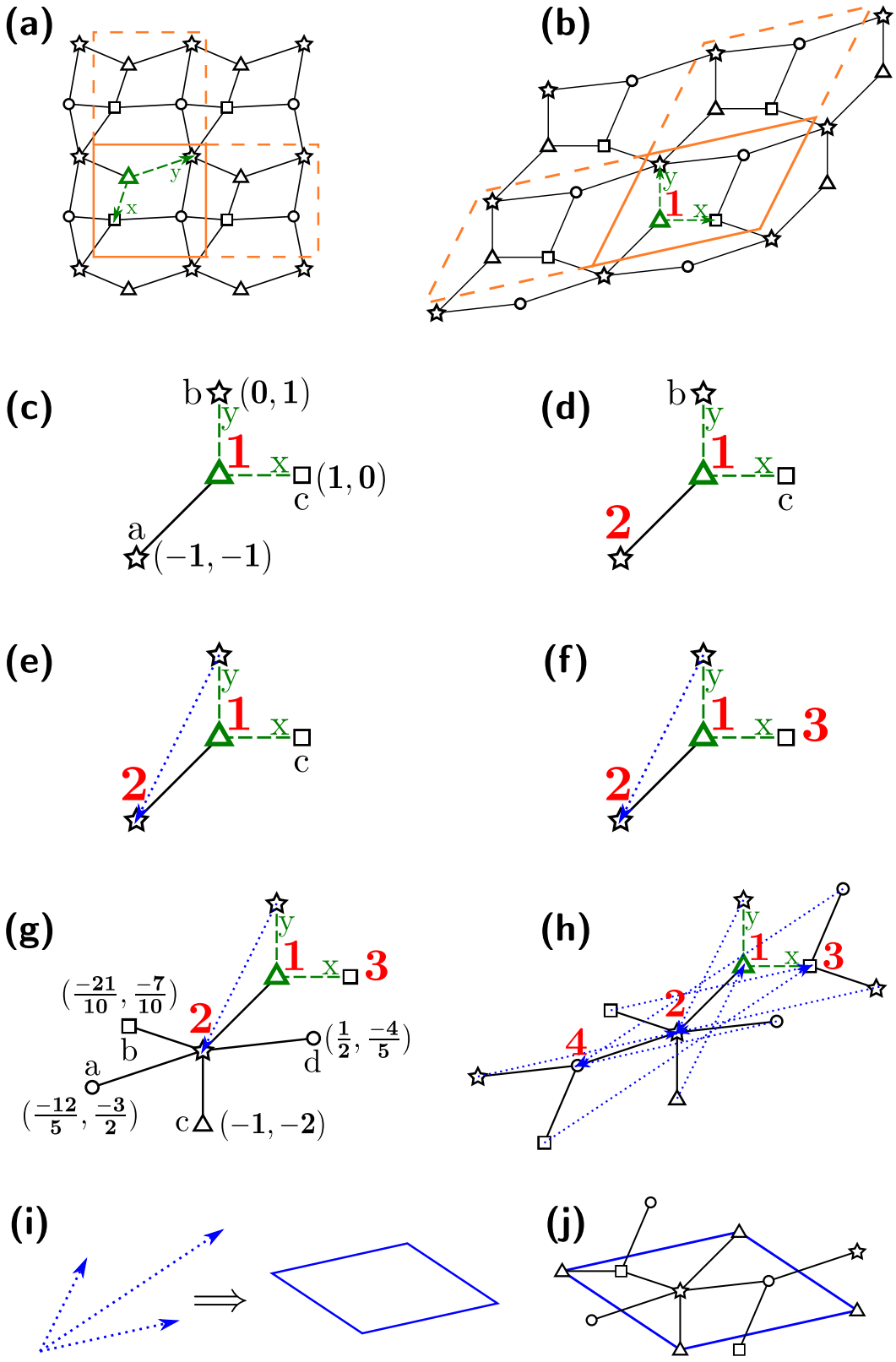


Figure 1.4: Illustration of the algorithm used to extract a key from a candidate on a 2-dimensional graph

(a) Start from a candidate (in green), whose origin is △ and axes are x and y. Initialise an empty queue of vertices.
(b) Represent the graph in the basis given by the candidate (the former unit cell and two of its neighbours are represented in orange). Note that axes x and y are now orthogonal. Assign number 1 to the origin △ and push it to the queue.
(c) Take the first vertex u of the queue and sort its neighbours according to the lexicographical order of their position. Here, $u = 1$ and the sorted neighbours are a, b and c.
(d) For each neighbour v', in order, if it is not the representative of an already encountered vertex, assign it the first available number between 1 and n and push it to the queue. In this case, neighbour a is assigned number 2.
(e) Otherwise, store the translation between the newly encountered representative v' and that which has been recorded v (the blue arrow). Here, v' is neighbour b and v is vertex 2: both are representatives of the same vertex.
(f) Continue until all the neighbours have been processed. In this case, only neighbour c remains and it is assigned number 3.
(g) Repeat the steps from **(c)** until the queue is empty. Here, the neighbours of 2 are being processed.
(h) When the queue is empty, all numbers between 1 and n have been assigned and a set of translations has been recorded.
(i) Triangulate the set of positive translations to obtain a new unit cell (see section 1.2.6).
(j) Convert the translations to integer offsets using the unit cell to find a new representation of the graph, and extract the key.

Figure 1.4: Continued: *Illustration of the algorithm used to extract a key from a candidate on a 2-dimensional graph*

Step **(i)** deserves special consideration. It consists in building a new unit cell from the set of positive translations t of the edges, positive in the sense of being lexicographically greater than $0_{\mathbb{Q}^3}$. Negative translations are negated to only keep positive ones. By construction, this set does not depend on the previous representation of the graph, it only depends on the candidate. As a consequence, any deterministic algorithm that transforms a set of translations into a unit cell can be used for this purpose. Delgado-Friedrichs and O'Keeffe[15] mention this point and suggest triangulating the translations, without detailing their algorithm. Before presenting the one used here, let's sketch the general idea and prove a lemma.

Given their origin, translations of the set should correspond to integer offsets between vertices and their representatives in the new unit cell to be found. Thus, the matrix $\mathrm{B} \in \mathbb{Q}^{3\times3}$ representing the unit cell must be such that each positive translation $t \in \mathbb{Q}^3$ of the set verifies:

$$\mathrm{B}^{-1}t = o \in \mathbb{Z}^3 \tag{1.5}$$

Not all such basis B will do, since it should also represent a unit cell, hence the columns of B must be valid translations. Since all positive translations of the set map some vertex to one of its representatives, they are all valid translations, so it is tempting to look among the integer

combinations of these translations to find the columns of B. However, it remains to be proven that such combinations indeed span the entire group of valid translations.

To show this, let's take the Euclidean space of step **(b)** and consider a path between the origin and any of its representatives. This path consists in a series of edges $(u, v, t) \in \mathrm{V} \times \mathrm{V} \times \mathbb{Q}^3$ where t is the translation between vertex v and the vertex v' which is the actual neighbour of u. After any number of steps, let's note $(u, \tau) \in \mathrm{V} \times \mathbb{Q}^3$ the pair where u is identifier of the vertex on which the path has just landed and τ is the sum of the translations t encountered so far. The initial pair thus is $(1, 0_{\mathbb{Q}^3})$, since 1 is the number of the origin. An induction on the number of steps shows that the position of the exact vertex on which the path lands when reaching pair (u, τ) is that of vertex u translated by τ. As a consequence, when reaching the final vertex of the path, the pair is $(u_{\text{end}}, \tau_{\text{end}}) = (1, \tau_{\text{end}})$. Thus, by construction τ_{end} is the sum of translations t stored in the edges during the algorithm, so it is an integer combination of the positive translations of the set. But since this reasoning was valid for the translation between the origin and any of its representatives, this proves that any valid translation can be written as an integer combination of positive translations of the set.

Let's now proceed to the algorithm that converts the set of positive translations into a matrix B satisfying Equation 1.5.

First, note that most positive offsets present in the usual representations of periodic graphs are one of either $(1, 0, 0)$, $(0, 1, 0)$ or $(0, 0, 1)$. As a consequence, in the most common cases, there are only three elements in the set of positive translations, which can simply be mapped to the three canonical offsets. This fact serves as a heuristic optimisation for the more general algorithm, by representing all the positive translations in the basis $\mathrm{A} \in \mathbb{Q}^{3\times3}$ whose columns are the three first linearly independent positive translations. The reverse transformation will be applied at the end of the algorithm. Overall, this operation is not crucial for the algorithm to work and the input remains a set of translations, but it tends to reduce the denominators of the coefficients of the translations, which is computationally helpful.

Given a set of translations, the goal now is to find three combinations such that they form the columns of a matrix B satisfying Equation 1.5. In other words, these combinations must span the group of valid translations. This problem on rationals is lifted to integers by storing the least common multiple of the denominators of the coefficients of the translations columnwise, and pointwise multiplying the translations by this triplet of integers in order to obtain a set of integer translations.

From that point, several algorithms exist in order to solve the problem on this integer form.[41–44] The standard procedure is to build the rectangular matrix whose lines are the triplets representing integer translations and put it in Hermite normal form,[51] which is precisely such that it is built from an integer combination of the initial triplets and still spans the same group as that of the triplets.

Determining the exact Hermite normal form is useless in this case however, since there is no need for the complexity brought by its other properties, and the existing solvers are specialised to have their best performance on big square matrices, not thin rectangular matrices one of its

dimensions being only 3. Thus, we implemented a custom solver for this particular problem which performs the following steps, for each column j in a fixed order:

- Find Bézout coefficients of the j-th element of the integer translations, as well as their greatest common divisor $d \in \mathbb{N}^*$. To do so, we implemented an efficient algorithm from Havas *et al.*.[51]
- Set the j-th column of B as the sum of the integer translations weighted by their Bézout coefficient.
- For each integer translation $t \in \mathbb{Z}^3$, update it to zero its j-th element by subtracting the new column of B times the quotient of the j-th element of t by d. By construction, after this step all translations have at most $(j-1)$ non-zero elements.

Finally, columnwise divide B by the triplet of least common multiples that was found before to get back to rationals, and left-multiply it by A to go back to the Euclidean space specified by the candidate.

The resulting matrix B has all the required properties for being a unit cell. Thus, each translation $t \in \mathbb{Q}^3$ of an edge now corresponds to an offset $o = \mathrm{B}^{-1}t \in \mathbb{Z}^3$ with respect to the unit cell, so the lexicographically ordered set of edges $(u, v, o) \in \mathrm{V} \times \mathrm{V} \times \mathbb{Z}^3$ is a representation of the initial graph that does not depend on its initial representation, but only on the candidate.

Let's note that this method transforms a set of translations into a unit cell, and this is precisely the problem encountered when minimising the cell from the set of all valid translations in section 1.2.5. However, in the current case, the procedure cannot rely on a starting cell to minimise, which is why it is more complex.

To conclude, let's observe that this representation might be impossible to embed, in the sense it may be impossible to put all the vertices of the graph, as they were determined by the algorithm, within the same unit cell given the constraints on their edges. This can be circumvented by taking appropriate representatives for each vertex given a fixed unit cell, but even that is unnecessary since the goal is only to extract a key, and it does not matter whether the key itself can be embedded.

1.3 Reduction of the search space

The only aspect of the algorithm that remains to be explored is the selection of the candidates, which is a crucial point for the overall complexity of the algorithm. After explaining how the implementation of Systre does this, we will present a series of novel reductions of the search space, *i.e.* of the number of candidates, that yield a notable performance gain.

1.3.1 The initial search space

Let's recall that a candidate is defined as a triplet of edges, which is used to create a new representation for the graph as explained in section 1.4. The topological genome is chosen as the lexicographically smallest key extracted from the candidates, so the set of candidates must be chosen in a fashion that is independent of the initial representation of the graph, lest the selection biases the search space. As a consequence, Delgado-Friedrichs and O'Keeffe[15] proposed the following selection pattern:

- If there exists at least one vertex such that its outgoing edges are not all coplanar, then the set of candidates is the set of triplets of linearly independent edges that all start at the same vertex.
- Otherwise, the candidates are all triplets of linearly independent edges such that the two first start from a common vertex and the third starts from another vertex.

In both cases, the set of valid candidates is not empty (otherwise it would mean that the graph is either not connected or not 3-periodic) and independent of the initial representation of the graph. It can be further reduced by removing duplicate candidates that yield the same key, using symmetry considerations as will be shown in section 1.3.3.

For many graphs, this search space remains rather large compared to the set of all possible triplets of edges. Our goal in this project was for our software to be able to tackle very big structures very efficiently, for two different types of applications: high-throughput screening of hypothetical materials, where the topological detection is performed on hundreds of thousands (and sometimes several millions) of structures,[52] and application to very large periodic structures, such as models of amorphous framework materials.[53] Therefore, reducing this search space is crucial for performance since it directly tackles one of the bottlenecks in the original implementation of Systre.

1.3.2 Categorisation of vertices and edges

Since the search space must remain independent of the representation of the graph, it can only depend on some properties that are common to all its representations. Such a property is called a *topological invariant*. The only invariant used so far is whether there exists a vertex whose outgoing edges are not all coplanar, but many other invariants exist, such as the three used by ToposPro.[13] These properties can be used to considerably reduce the search space by separating vertices according to them.

The core idea of vertex categorisation consists in trying to separate vertices into as many categories as possible, each of these categories being identifiable independently of the representation of the graph. For instance, in this work, vertices are separated according to their coordination sequence (the number of vertices at distance k from the vertex of interest with $k = 1, 2, \ldots$, see Introduction), so that two vertices are in the same category if and only if they share the same coordination sequence up to distance 10.

Vertex categories are then sorted by the total number of outgoing edges from vertices in that category and, in case of equality, by the invariant they were defined from. For example, in our case, ties are broken by lexicographically sorting according to the coordination sequences.

With these sorted categories in mind, let's look for candidates and try to minimise their number. The following strategy is used in our improved algorithm:

- Let c be the first vertex category such that there is a triplet of non-coplanar edges starting from the same vertex in c (if no such category exists, skip to the alternative below). Then, for all such triplets, let c_1, c_2 and c_3 be the sorted categories of the vertices to which the initial vertex is bonded with these edges. There are four subsets of such triplets, characterised by whether they satisfy:
  1. $c_1 = c_2 = c_3$.

2. $c_1 < c_2 = c_3$.
3. $c_1 = c_2 < c_3$.
4. $c_1 < c_2 < c_3$.

Then the candidates are all the triplets of edges whose categories are in the last non-empty such subset with (c_1, c_2, c_3) lexicographically minimal.
This strategy illustrated for the 2D case in section 1.5.

- Alternatively, if all vertices have only coplanar outgoing edges, let c be the first category such that there exists another category c' such that there is a non-coplanar triplet of edges, the two first of which start from a vertex of c and the last from c'. c' is chosen as the first such category. Let c_1, and c_2 be the sorted categories of the destinations of the two first vertices, and c_3 be the destination of the third. Then the candidates are all the triplets of edges, the two first of which start in c and the last from c' such that (c_1, c_2, c_3) is lexicographically minimal.

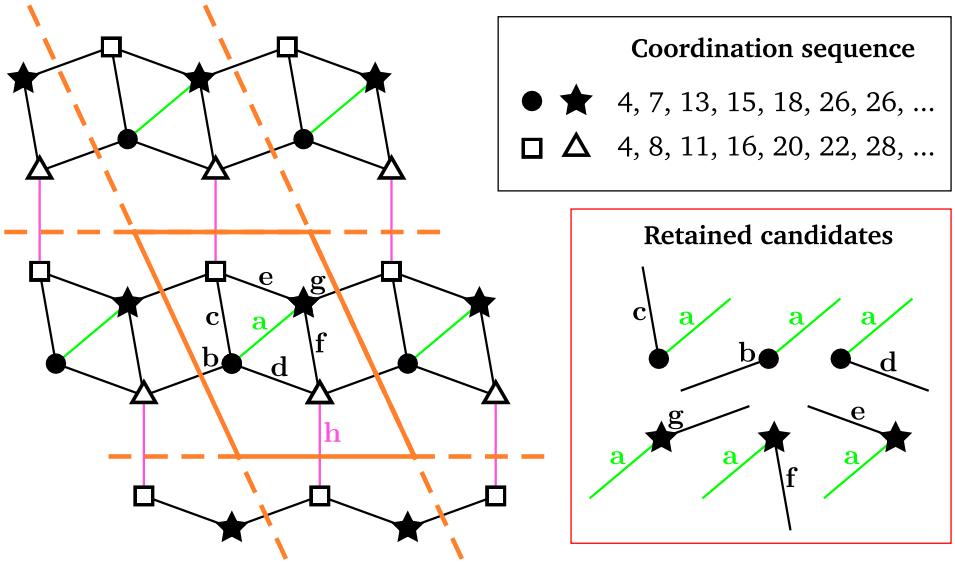
The main difference with the strategy of Systre is that the categorisation of vertices bounds the number of candidates with respect to the maximal size of a category, hence the goal of having as many small categories as possible. However, the two strategies are exactly the same in the worst-case scenario where there is only one category. Note that, although rather complex, the constraints on the retained candidates described above can actually be implemented in an efficient way, which is necessary to prevent the search space reduction from taking more time than the actual search space exploration.

1.3.3 Reduction through symmetry

In addition to the individual properties of vertices which are used to separate them into categories, symmetry considerations on the equilibrium placement can be useful in reducing the search space. Indeed, if two candidates are symmetry-related, then the representations extracted from both will be exactly identical. This optimisation is already used in Systre, and in the current work, it is useful to perform a symmetry analysis on the equilibrium placement of the unit cell before even categorising vertices.

Unfortunately, we do not know of any library that can find symmetries for an input given in arbitrary precision; all existing libraries take approximate positions, and thus return only approximate symmetries. We use the *spglib* library[49] to this purpose. *spglib* is a C library, which makes it easy to interface with Julia without sacrificing speed. Since all its inputs and outputs are approximate, the returned symmetries need to be checked against the exact equilibrium placement of the graph to rule out false positives. Because of the imprecision of the input, there might also be some false negatives – that is, some symmetries may be missing – but they do not matter because symmetries can only help to eliminate candidates that are symmetric images of other ones already encountered. Hence, the use of symmetries is purely an optimisation, but it has no influence on the result of the algorithm.

Symmetries are useful to avoid needlessly recomputing coordination sequences when dividing the vertices into categories. Indeed, the search space is reduced by singling out one vertex per symmetry orbit, and using the convention that only these can serve as the source of the first edge of a candidate. No such constraint can be placed on the destinations of the edges unfortunately, as that would be unsound.



1. The coordination sequence of all vertices is computed. In this particular case, there are two vertex categories, one represented with filled symbols (circle and star) and the other with empty symbols (square and triangle). Edges are colored in green when they connect two filled symbols, in magenta for two empty symbols and black otherwise.
2. Both categories have the same number of outgoing edges, so they are sorted in lexicographical order of the coordination sequences: the "filled" category comes first.
3. There exists a filled vertex with two linearly independent outgoing edges (here, all vertices actually match the criterion because no two neighboring edges are aligned). Hence, c is the "filled" category.
4. Consider all ordered pairs of linearly independent edges starting from a filled symbol u. There are 24 such pre-candidates, represented as the two first lines of section 1.6. Among those, only 6 bond u to vertices of category c_1 and c_2 respectively, such that $c_1 < c_2$. Keep those as candidates.
   If there was no such pre-candidate, the candidates would have been all the pre-candidates where $c_1 = c_2$ with c_1 minimal, *i.e.* "filled" if possible, "empty" otherwise.

Figure 1.5: Illustration of the candidate selection on the **cqe** *net.*

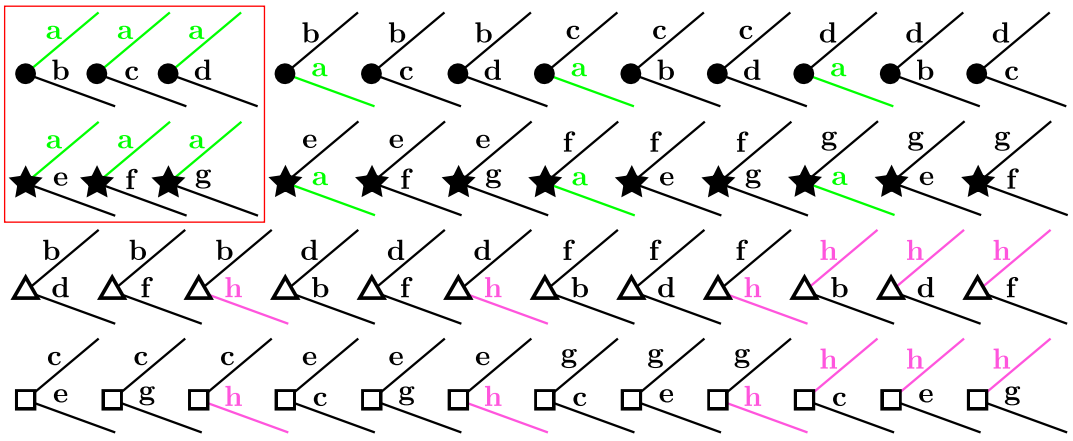


Figure 1.6: The 6 candidates of the **cqe** *net selected by CrystalNets.jl, among the 48 possible candidates.*

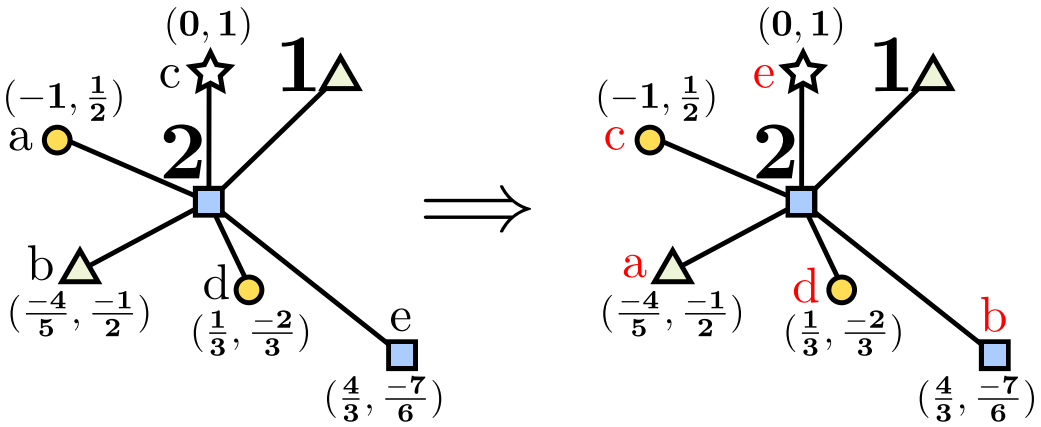


Figure 1.7: Difference between previous and new ordering of neighbours at step **(c)** *on an example. Representatives of the same vertex have the same symbol.*

On the particular example of the **cqe** net shown in section 1.5, there is only one symmetry and it maps each symbol of a category on the other of the same category. As a consequence, only one of the two first lines and one of the two last lines of candidates shown in section 1.6 need to be kept. Systre thus examines 24 candidates, and only 3 for CrystalNets.jl.

1.3.4 Ordering of keys

Finally, the algorithm can be made faster by changing the mode of comparison of the candidates. In Systre, the topological genome was defined as the smallest key extracted from a candidate. However, we observe that, arrived at step **(h)** of the algorithm in section 1.4, both the embedding of the graph as well as the ordering of its vertices only depend on the candidate. Hence, we can define at that point a pseudo-key consisting in the sorted list of triplets $(s, d, t) \in \mathrm{V} \times \mathrm{V} \times \mathbb{Q}^3$ if there is an edge between the vertex defined by its number s and the representative of vertex number d which is translated from the latter by t. This pseudo-key only depends on the intermediate representation of the graph obtained in step **(h)**, which means it is exclusively determined by the candidate, so we can define the genome as the key extracted from the candidate with the smallest pseudo-key.

Instead of comparing the keys extracted after a costly lifting of the positive translations to a proper representation of the graph, it is thus sufficient to compare the pseudo-keys. Moreover, this pseudo-key can be computed incrementally during the main loop of the algorithm. To do so, at step **(c)**, the neighbours of vertex u should be sorted in a different manner (represented on section 1.7): all representatives of the same vertex should be contiguous and lexicographically sorted according to their position within each category; the categories corresponding to vertices which have a number (assigned in step **(d)**) are placed first and sorted according to this number, while the others are placed afterwards and sorted according to the position of the first vertex in the category. This makes it so that each edge of the graph is added to the pseudo-key in the order in which it is studied (in steps **(d)** – **(f)**), while preserving the invariant that the pseudo-key is lexicographically sorted.

Since the pseudo-key is constructed on-the-fly, it can thus be compared at each step with the previous smallest pseudo-key. If a triplet of the current pseudo-key is lexicographically greater than the triplet at the same position in the previous best pseudo-key, the computation for the candidate is aborted; if it is lower then this comparison can be skipped for the rest of the

computation and the pseudo-key will be the new smallest. Finally, once all the candidates have been checked, the minimal pseudo-key is converted to an actual key through steps **(i)** and **(j)**, yielding the topological genome of the graph.

1.4 Application to materials databases

CrystalNets.jl is a Julia package which can be intuitively used from an interactive Julia shell or as an executable to determine the structure of single crystallographic files in the variety of file formats recognised by chemfiles[5]. Since chemical bonds are not systematically present in such files, it uses a custom bond detection algorithm inspired from that of chemfiles, and can export both the input with the detected bonds as well as the extracted underlying net to allow visually checking the correspondence with the input. The bonding algorithm reuses available information on the structure to accurately account for the larger bonds surrounding metals in MOFs for instance, but the user can also override the cutoff distance for bonding if necessary.

By combining the bond detection heuristic with topology determination, CrystalNets.jl can also be used to survey large databases of crystalline structures, without the need for user input or relying on other chemical information (such as choice of SBUs) on the structures themselves. In this section, we demonstrate the code by exploring a series of available materials databases.

1.4.1 RCSR, EPINET and IZA-SC database

The Reticular Chemistry Structure Resource (RCSR)[7] [6] is a repository of 3,132 structures with 2 or 3-periodicity (excluding weaving and 0- and 1-periodic structures), each given a canonical name. It offers a Systre archive containing 2,928 nets that we reprocessed to obtain the 2,928 topological keys corresponding to each structure. These keys are not the same as those of the archive because of our algorithmic differences explained before.

We evaluated the performance of CrystalNets.jl by comparing its time to compute the topological keys of 2,053 structures of the RCSR chosen at random, compared to both Java and JavaScript (JS) implementations of Systre[7]. All three programs were executed 6 times with one input and 6 times with another, each input being a textual file containing a different representation of the same 2,053 nets. All ran on a single thread with Intel Xeon Silver 4210R CPUs (2.4 GHz). The average timing over the 12 executions is represented on section 1.8. On average, CrystalNets.jl is 56 times faster than the Java version of Systre, and 117 times faster than the its stripped-down JavaScript implementation. Two outliers are marked with a black arrow: these correspond to nets **sas** and **dhe**, the two first nets in the input file of dimension respectively 3 and 2. Since Julia is compiled Just-In-Time,[29] it must compile the entire algorithm the first time the program computes the topological genome of a net, hence a significant overhead. Rerunning the computation on the same two nets after compilation yields an average timing of 3.9 ms for **sas** and 5.9 ms for **dhe**, compared to 63 s and 18 s respectively when including compilation, as represented on the figure.

[5] full list available at https://chemfiles.org/chemfiles/latest/formats.html#list-of-supported-formats
[6] available online at http://rcsr.anu.edu.au/nets (3D) and http://rcsr.anu.edu.au/layers (2D)
[7] available online at https://github.com/odf/gavrog and https://github.com/odf/systreKey respectively

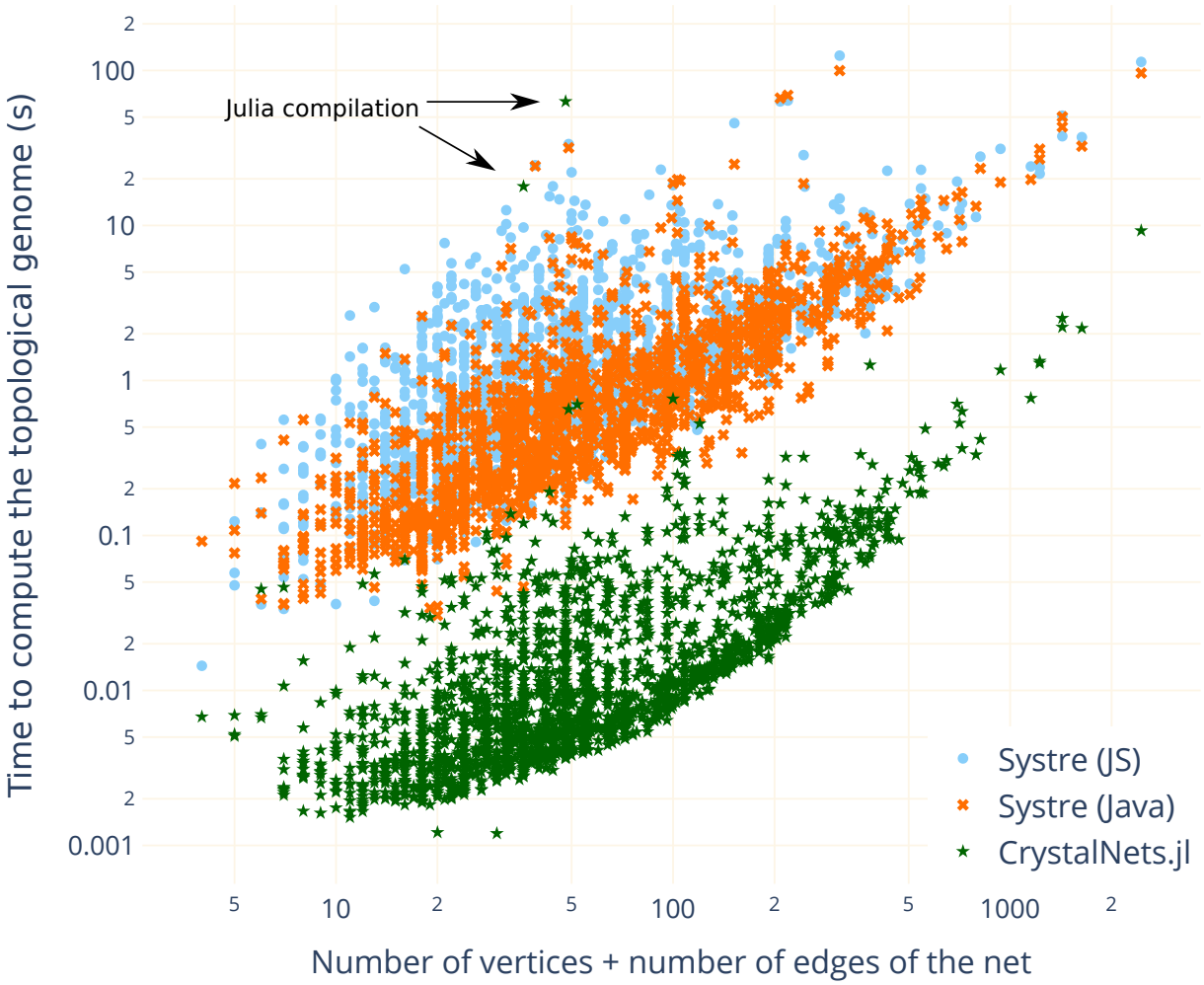


Figure 1.8: Comparison of processing times for 2,053 nets with Systre and CrystalNets.jl.

This performance is consistent across the nets. On the **tep** net, which is the largest of the RCSR (920 vertices and 1840 edges per unit cell), CrystalNets.jl computes a topological genome in 8.6 s, compared to 229 s for the Java implementation of Systre, and 193 s for the JavaScript version. For the smallest net **pcu**, CrystalNets.jl takes 6.5 ms, which is twice as fast as the JavaScript implementation of Systre, and 16 times faster than the Java version. The maximum observed speedup is for the **tsl** net where CrystalNets.jl takes 7.1 ms on average compared to 5.2 s for the Java version of Systre and 16 s for the JavaScript version.

The RCSR partially overlaps with the larger EPINET database[10] [8] which contains, among other data, 14,532 stable 3D nets, which we processed similarly. Finally, both databases partially overlap that of Structure Commission of the International Zeolite Association (IZA-SC),[54] which provides names for 255 zeolitic framework structures and corresponding CIF files.[9] These structures have unambiguous vertices, defined by the tetrahedral atoms, and edges between two such atoms bonded by an oxygen, thus a uniquely-defined underlying net.

[8] available online at http://epinet.anu.edu.au
[9] available online at http://www.iza-structure.org/databases/

The results for all three databases are stored within CrystalNets.jl, so that they may be used to recognise the topology of structures processed by the code. In the following, we call *unknown* nets those that are not part of any of these databases — although it should be noted that other complementary databases of nets do exist (in particular the proprietary databases used by ToposPro).

Finally, we note that this processing of the databases exposed a few mismatches between the names of the structures given in the three databases, which were all reported. Namely, the RCSR nets **ucn**, **rad**, **jst**, **fuv** and **fvb** have as IZA-SC names SBN, ATS, JST, NPT and PTT, which did not appear on the RCSR. Additionally, we noticed that 137 structures from EPINET were missing their RCSR name, while 6 RCSR structures mentioned an incorrect EPINET counterpart. This highlights the need for systematic and automated checks of material and topology databases for consistency.

1.4.2 MOF structures

In order to test CrystalNets.jl on a realistic workload, we surveyed the CoRE MOF 2019[55] database of metal–organic frameworks (version 1.1.3). To do so, we implemented a custom clustering algorithm to regroup atoms into SBUs. Since the definition of MOF SBUs is not universal[3] and IUPAC recommends providing complementary topological descriptions when necessary,[32] we implemented several clustering options to account for different strategies used in the literature, detailed in **??**.

To simplify the presentation, we will only discuss the results when using the "single nodes" approach, as well as the "MOF" structure option which widens metallic bond radii. CrystalNets.jl recognises 61% of all 21,692 resulting structures of CoRE MOF 2019 as one of the nets in either the RCSR, the IZA-SC or the EPINET databases (only 33% structures are recognised with "PEM" and between 45% and 55% with the other clustering options). Among the unrecognised structures, less than 2% correspond to unstable nets which CrystalNets.jl does not handle. These proportions are similar for each of the four subcategories of the database, divided whether only free solvent was removed (FSR) or all solvent molecules (ASR), and within each, whether the structure is disordered or not. The closeness of these numbers is expected since most structures from FSR also belong to ASR with only molecules of solvent as difference, which should not affect the topology. This highlights the robustness of the algorithm.

Next, we focus on the 11,190 ordered structures of the ASR subset to avoid duplicates. By studying the distribution of the number of structures per corresponding net, we observe that this distribution drops faster for unknown nets than for known ones. This is not surprising since we expect known nets to correspond to the most common occurring in nature, whereas unknown nets should only correspond to at most a few corresponding structures. Indeed, among the latter, 1,488 unknown nets only correspond to one MOF structure each, and 302 to two structures. Yet, 5 of them still correspond to thirty MOF structures or more each: we manually checked through the TopCryst interface[16] that these 5 nets were part of the ToposPro database, but we could not do so for all the unknown nets since these databases are not open. In comparison, for known nets, 110 of them only correspond to a single MOF structure, 54 to two and 38 to thirty or more each. These most represented nets are detailed in section 1.9: **pcu** (primitive cubic lattice), **dia** (diamond) are by far the most common, together accounting for

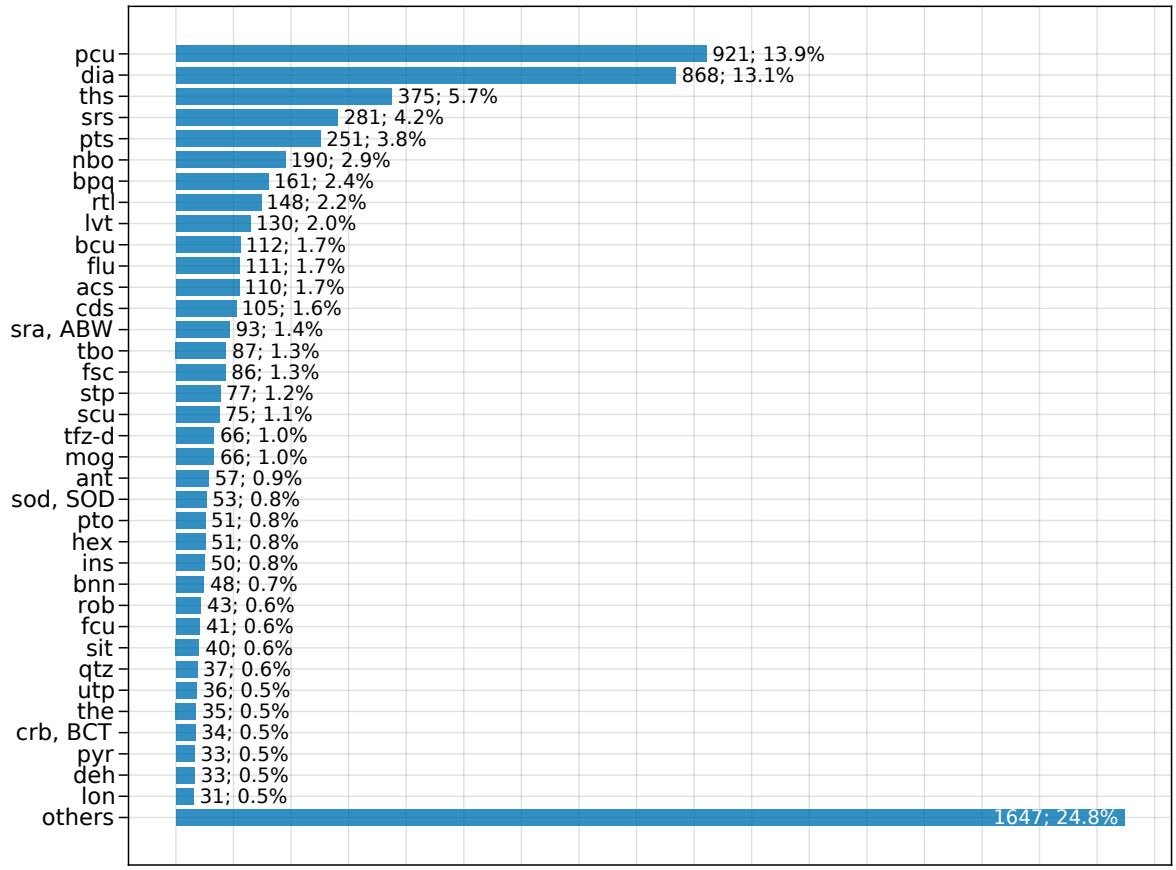


Figure 1.9: Number of structures per known net in the ordered structures of ASR.[55]

27% of know nets, followed by **ths**, **srs** and **pts**. For zeolitic nets, ABW, SOD (sodalite) and BCT are the most commonly encountered.

The high number of nets with known topologies is a good indicator of the quality of our clustering algorithm. Yet, since it partly relies on heuristics, it should not be considered fail-proof. To assess its validity, CrystalNets.jl exports both the input and the detected net in VTF format, which can be superposed in a molecular visualisation software to check the correspondence between the vertices of the nets and the expected clusters. We manually checked this correspondence for hundreds of structures with both known and unknown nets, to assert the validity of our implementation. Comparison with TopCryst on many instances comforts our belief that the vast majority of unknown nets encountered in CoRE MOF are not the result of a wrong clustering by CrystalNets.jl, but simply missing from the databases we use, probably because they occur on only a handful of actual structures. On the other hand, the proprietary databases of ToposPro appear to include many such nets.

We compared CrystalNets.jl and ToposPro, accessed through its online TopCryst interface, on a selection of MOFs common in the literature. The result, presented on section 1.10, shows that both usually return the same topology if it is in the RCSR when using the “single nodes” or the ‘all nodes‘ clustering options. On the other hand, the “standard” clustering leads to different results in several cases. This comes from the difference in clustering heuristics: our implementation regroups each paddle-wheel into a single SBU and removes SBUs made of a lone

Name (CSD identifier)	CrystalNets.jl			ToposPro	
	Single/All Nodes	Standard	PE	Single/All Nodes	Standard (MOF)
HKUST-1 (LUDLED)	**tbo**	**tbo**	☐	**tbo**	5,6T2
JXUST-1 (SAXVEW)	**pcu**	☐	☐	**pcu**	3,5,5,5T11
MIL-53 (MINVUA02)	**bpq/rna**	**bpq**	**sra**, ABW	∗	**bpq**
MIL-100 (BUSPIP)	**moo**	☐	☐	*Error: timeout*	
MIL-101 (OCUNAC)	**mtn-e**	☐	**mtn-e-a**	*Error: timeout*	
MOF-5 (FIQCEN)	**tbo**	**tbo**	☐	**tbo**	5,6T2
MOF-14 (QOWRAV)	**pto**	**pto**	sqc11259	**pto**	5,6T46
MOF-177 (BABRII)	☐	☐	☐	3,3,5,6T36	4,4,4,4,4,6,8,8T1
MOF-801 (BOHKAM)	**fcu**	**xbi**	**ubt**	**fcu**	3,4,8T15
PCN-700 (RUBLAD)	**bcu**	☐	**pcb**, ACO	**bcu**	3,3,4,6,8T9
UiO-66 (SURHEU)	**fcu**	**xbi**	**ubt**	**fcu**	3,4,8T15
ZIF-8 (FAWCEN)	**sod**, SOD	**sod**, SOD	**sod-e**	**sod**, SOD	**sod**, SOD
ZIF-67 (GITTOT02)	**sod**, SOD	**sod**, SOD	**sod-e**	**sod**, SOD	**sod**, SOD

Figure 1.10: Comparison of CrystalNets.jl and ToposPro[13] (through TopCryst[16]) for the identification of topologies on a selection of MOFs.
Topologies in bold come from the RCSR,[7] in capitals from IZA-SC,[54] sqc11259 comes from EPINET[11] and the rest are part of ToposPro databases. ☐ indicates unknown nets.
*∗: ToposPro does not provide Single/All Nodes results for MIL-53 but returns **rna** as "Standard representation of covalent and ionic compounds" in addition to **bpq** as "Standard representation of coordination compounds and valence-bonded MOFs".*

oxygen (while bonding their neighbors together) while ToposPro does not systematically do so. These two heuristics can be individually toggled off in CrystalNets.jl by the user if necessary. In practise, we observe that keeping these heuristics usually leads to less clusters without loss of chemical meaning, which can in turn help identify relevant underlying topologies, like the **xbi** net for MOF-801 and UiO-66 which contains the information on connectivity between individual metal atoms and with organic linkers. Having access to other clustering options, like "PE" also allows considering alternative topological descriptions of the topology.

ToposPro may fail for big nets because of the 3-minutes timeout of TopCryst: this happens for MIL-100 and MIL-101 among the tested structures. By comparison, CrystalNets.jl takes less than 10 s to handle them, but these timings are not comparable because we do not know the performance of the server on which the TopCryst webservice depends.

Since our analysis relies solely on CrystalNets.jl, which is independent from the tools used to build and clean the CoRE MOF database, it allowed us to identify 28 improper structures in the database, which were reported.

1.4.3 Database of aluminophosphates

Additionally, we surveyed the topologies of Zheng, Li and Yu's database of aluminophosphates.[56] These 385 experimental structures are given as CIF files with little to no cleaning from the publications they stem from. In particular, solvent is included and the structures can contain multiple overlapping atoms with partial occupancy, while atoms of the main structure are

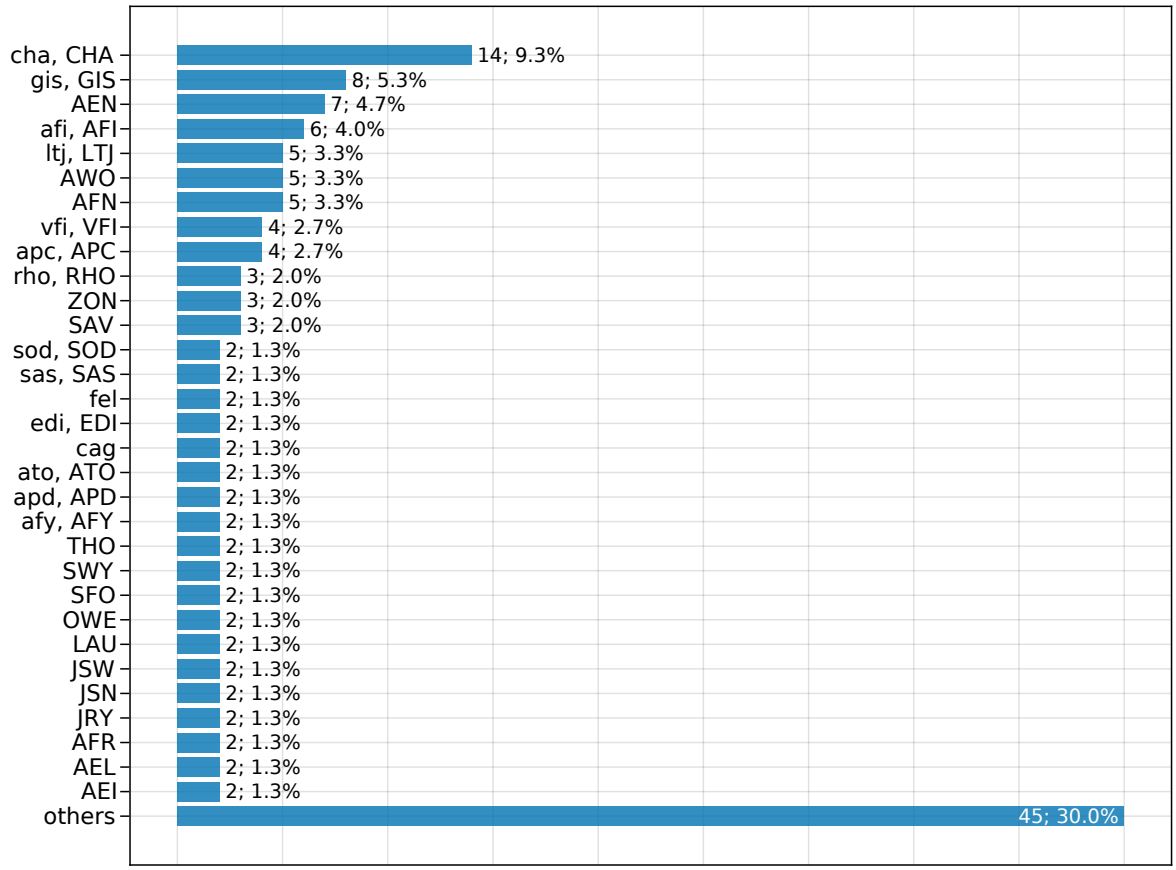


Figure 1.11: Number of structures per known net in the ALPO database.[56]

often partially substituted. CrystalNets.jl automatically removes 0-dimensional residues like solvent and keeps for each site only the atom with the highest occupancy. Using the "Zeolite" structure option of CrystalNets.jl, we recognise 64% of the 205 3D nets in the database. This number rises to 73% by adding a heuristic to remove P–O–P and Al–O–Al linkages, which are usually not accounted for when computing the topologies of AlPOs.

The 31 known nets represented on more than one structure in the database are represented on section 1.11. With the exception of **fel** (originating from feldspars) and **cag** (occuring in variscite), they are all referenced in the database of zeolite frameworks of IZA-SC. This is expected since AlPOs are structurally close to zeolites, with the replacement of Si atoms by Al and P. Conversely, it shows that **fel** and **cag** may be nets underlying the structure of unreported zeolites, especially since both only have 4-coordinates vertices, corresponding to the T-atoms, and can be embedded to exhibit pores.

Most known structures have a net represented only once (45 nets *i.e.* 30%) or twice (19 nets *i.e.* 25%). In comparison, among the unknown nets, 1 occurs on three structures, 5 on two and the 32 others appear only once each. Both distribution tails and comparison of known and unknown nets follow a pattern similar to the CoRE MOF database.

For the structures for which the underlying net has been recognised as one of the IZA-SC, there is little doubt on the validity of the detected net. For the others however, there is no systematic

way of determining whether the detected net is indeed that representing the structure or not except from visually checking the adequate superposition of the detected net and the initial framework for each file. This limitation comes from the lack of cleaning of the input CIF files, which can lead CrystalNets.jl to interpret solvent residues as part of the framework for instance, or duplicate atoms as two different atoms instead of one. This issue can be circumvented by either cleaning the CIF inputs, as in the CoRE MOF database, or by manually checking the correspondence if working on individual files. Obviously, this limitation is inherent to the quality of the input files and affects all topology determination codes.

1.4.4 Hypothetical zeolites

The study of framework topology has historically been extensively applied to zeolites.[57] One reason is that the structure of zeolites, made only from tetrahedral (T = Si, Al, ...) atoms bridged by T–O–T linkages, is simple enough to show an almost one-to-one correspondence between each framework and its underlying net. Using this fact, large databases of hypothetical zeolites have been devised: the core idea consists in embedding 4-connected nets in real space, either by enumeration of the nets[58] or by Monte-Carlo sampling of vertex positions.[59] The vertices correspond to Si atoms while O atoms are placed along the edges. Each obtained structure is subsequently refined by minimizing its energy through molecular dynamics and with simulated annealing.

Since these structures are built from their underlying net, we tried to perform the opposite operation on the Predicted Crystallography Open Database (PCOD)[59] to assess our implementation. This database contains 331,172 CIF files representing zeolites with energies within 30 kJ/mol (per SiO_2 unit) of α-quartz, which were analyzed by CrystalNets.jl in 45 minutes using 20 cores (Intel Xeon Silver 4210R CPUs, 2.4 GHz). Bond guessing forced O atoms to have at most 2 neighbours, which is the default heuristic for O atoms not bound to metals: this incidentally ensures that all vertices of the net correspond to Si atoms.

As the SiO_4 tetrahedra are somewhat distorted during the refinement steps, the bonds guessed from the resulting structure do not always correspond to those from which it was built. As a result, we found 11 frameworks whose T-atoms were not all 4-connected, and 75 other frameworks whose coordination sequences did not match that recorded in the database. Two structures were found to have the same underlying net: 8013876 and 8035701, the latter having different detected coordination sequences than indicated. Barring these 86 mismatched structures, 18 nets only have a 2D periodicity and 31 are unstable nets not handled by CrystalNets.jl. All those "problematic" nets represent less than 0.04% of all nets in the database. For the rest, no two structures had the same underlying net.

Indeed, by construction of the database, structures whose topology is already part of the database are removed. The invariants used to identify the topology are the coordination sequences and the ring sizes: this strategy does ensure that no two structures with the same net are present, but it is too restrictive since two distinct nets could share these invariants. In the future, the use of a unique topological identifier through CrystalNets.jl could thus potentially lead to the generation of more structures.

1.5 Web interface

We present CrystalNets, our web interface to the CrystalNets.jl library. The focus of this online tool is to provide easier handling than its programmatic back-end, to make it maximally accessible. Its interface, presented on [FIG] simply consists in an input box where the user can drag-and-drop or browse the input file. The main options are visible below, with pre-selected default values. These main options include the type of material, which may guide the automatic choice of the subsequent options, the bonding algorithm (whether automatic as discussed above, or using the input), the clustering strategy, and some visualization export choices. A panel of advanced options, folded by default, can also be accessed to tweak the cutoff radii of the bonding algorithm or toggle the aromatic cycle detection, among other choices.

Once the input and options have been selected, typing Enter or clicking the "Submit" sends the request to a server, which internally calls the CrystalNets.jl library with the correct input, formats the answer and transfers it to the user. The resulting page contains a green box with the identified topologies, as well as a visualization panel on the right. Visualization files can be downloaded by clicking on the icon next to the required exports: they are in VTF or PDB format, which can be used by VMD. A yellow box, folded by default, may also appear in case CrystalNets.jl emitted some warnings or information statements. If an error occurred, or if no net can be extracted (as may happen with a non-periodic input), a red box detailing the error appears. In any case, the HTML result page contains all the information; it can thus be saved locally and reopened at a later time if need be.

The visualization panel on the right of the result page relies on the 3Dmol.js library. This choice was mainly driven by its reactivity when displaying even large crystals, as well as its ease of manipulation. As with any such visualization tool, dragging the canvas rotates the crystal while zooming in or out is done with the mousewheel. The home button on the left resets the viewer position and zoom. Clicking on the right-most button, or double-clicking the canvas, displays it across the entire webpage, which can be convenient to focus on visualization.

The middle buttons control what is currently being displayed. Each button either directly controls a visible element, *e.g.* the button showing the input crystal or its trimmed version, or it reveals a menu with other buttons. For instance, each substructure of an interpenetrated crystal has its own menu. Navigating these menus thus allows interactively choosing the visible elements on the canvas, to enable, for example, focusing on the superposition of the input crystal and the "All Nodes" representation of its second substructure only.

The server handling the requests is written with the Bottle library in Python. Upon receiving a user query for CrystalNets, it calls a custom code written in Julia, passing on the input and related options.

This code relies on the DaemonMode.jl library to minimize the experienced latency. Indeed, Julia works with a Just-In-Time (JIT) compilation paradigm, which enables highly specialized compilation of code leading to very fast execution. The counterpart of this strategy is the compilation time incurred each time a function is called with a new set of argument types. Several mitigation strategies have been implemented in Julia, to reduce this "time to first

execution", the three main ones being writing type-ground code to reduce the amount of total compilation, hunting invalidation to prevent code from being re-compiled after loading and using precompilation to effectively compile part of the code ahead of time. The practical gains due to these strategies is considerable, especially with the advent of native code caching as part of precompilation in Julia v1.9.

However, the latency requirements of a webserver are particularly stringent, since this waiting time is immediately felt by the user. DaemonMode.jl is a Julia library offering an alternative latency-reduction scheme, well adapted to this use-case. Instead of launching a new Julia process with each request, it captures each query and forwards it to a single Julia process that operates all of them. JIT compilation can then be performed on the unique process at startup, and any subsequent Julia call will simply be executed by the same process, without the need to compile anything at all, effectively resulting in an ahead-of-time compilation of the necessary codebase.

Depending on the load state of the server, we observe that most requests can be served within a few seconds at most, the only cases exceeding ten seconds being very large nets for which the latency time is negligible with respect to the actual processing time of the net identification algorithm.

1.6 Conclusion and perspectives

We have presented CrystalNets.jl, a Julia library implementing topology identification on crystalline materials. The core algorithm for topology detection is strongly inspired from that used in Systre[15] which computes a topological genome, that is a sequence of numbers uniquely identifying the net underlying the given structure. The algorithmic and implementation improvements of CrystalNets.jl still preserve the soundness and polynomial time complexity of the initial algorithm, while offering a much better performance across all kinds of nets: it is on average 56 times faster than Systre and up to thousands of times faster in some cases. Moreover, our software directly takes crystallographic files as input and can automatically detect bonds if they are absent; it also includes a general clustering algorithm with different target options for framework materials, and heuristics adapted to metal–organic frameworks specifically.

Overall, CrystalNets.jl was successfully tested against a large diversity of crystalline structures: MOFs, aluminophosphates, zeolites and many other crystals. Even on large databases, our implementation shows great performance and can reliably recognise any structure in the Systre archive of the RCSR, in the IZA-SC database of zeolite structures and in EPINET. It successfully attributes known topologies to 60% of the structures in CoRE MOF after automatically clustering the vertices according to the "single nodes" policy. It was also able to recognise 73% aluminophosphate structures from Zheng *et al.*'s database,[56] automatically removing solvent residues and colliding atoms when possible.

We believe CrystalNets.jl can be used as a faster and more convenient alternative to Systre and ToposPro for the identification of nets underlying crystalline structures. Its programmatic interface can be integrated to any computational pipeline written in Julia, as well as some other languages through bindings. Looking forward, we plan on providing a website interface similar

to TopCryst,[16] which could be useful to compare any structure to one of those surveyed in the previously exposed databases. We also plan on including results from the topological survey of the COD[60] and CSD[61] databases. The latter could then serve as comparison for our bonding and clustering algorithms against those of ToposPro.[31]

2

CATION PLACEMENT IN ZEOLITES

Among the different families of adsorbents, zeolites are among the most used in industry. Small gases tend to adsorb preferentially on cations when they are present, which makes the study of cationic zeolites particularly relevant to the understanding of industrial adsorption processes. This chapter presents the chemical nature of cationic zeolites and explain the challenges and methods used to obtain representative models of them for subsequent adsorption simulations. The central part of the chapter focuses on the newly developed "shooting star" methodology, which significantly reduces the amount of resource needed to compute the repartition of cations in zeolites.

2.1 Cationic zeolites

In this section, we detail the general structure of cationic zeolites and highlight the challenges associated with the simulation of such systems.

2.1.1 Structure and properties

Zeolites are a family of aluminosilicates, first discovered by Axel Fredrik Cronstedt in 1756. Similar to glass, their atomic structure is made of tetrahedra where the central atom, called a T-atom, is usually silicon, bridged together by oxygens. They are crystalline however, with typical unit cell lengths varying between 10 Å and 100 Å, and have nanopores.

The different kinds of zeolites are separated by topology: each known zeolite topology is given a capital three-letter code by the International Zeolite Association Structure Committee (IZA-SC). As explained in **??**, the topology designates the information extracted from the network of chemical bonds: in the case of zeolites, this network can be simplified by taking one vertex per T-atom, and abstracting each oxygen bridge as a simple edge between the two corresponding T-atoms. This simple protocol makes it clear that the net of a zeolite is very close to the structure it represents – in other words, simplifying the crystal into its net does not lose much information on the initial structure – which explains why a classification by topology is the primary way to distinguish different zeolite structures. It is noteworthy that such a classification is vastly insufficient in the case of MOFs, where the chemistry of metal nodes and the nature of the ligands is thus used first; while for other kinds of materials such as glasses, it is downright impossible to establish because of the absence of cell periodicity.

While millions of hypothetical zeolite topologies have been identified by numerical methods [10.1039/C0CP02255A], only 256 have been experimentally observed[1] as of 2024. This discrepancy may stem from the metastability of zeolites, whose porosity make them thermodynamically less favourable than glass [10.1021/acs.cgd.3c00893]. On the other hand, new experimental zeolite topologies have been discovered each year in the last twenty years, which indicate that accessing new structures is also limited by the current synthetic methods used. Moreover, new synthetic routes can allow new chemistry on existing topologies [10.1039/D2SC06010H, REF].

Indeed, each given topology may give rise to several zeolites that differ by their exact chemical composition. Isoelectronic metal substitution consists in the replacement of silicon by germanium or, in some cases, titanium, zinc or tin [REF]. A much more common kind of substitution consists in replacing some T-atoms by aluminium, or, more anecdotally, boron,

[1] plus 9 intergrowth families, which are not strictly crystalline

gallium [REF]: each such substitution introduces one extra electron in the system, hence these zeolites contain cations to maintain electroneutrality. The nature of the cation itself leads to some more variability: sodium cations are usually the ones used during synthesis [REF], and can then be substituted by other cations [REF]. For simplicity, all zeolites discussed will be considered having only either Si or Al as T-atoms.

Ion exchange is actually the most prevalent industrial use case for cationic zeolites, especially as water softener for laundry agents [REF]. Their considerable internal surface and the accessibility of active sites such as the cations or the substituted metals make them also useful for catalysis [REF]. Other applications include nuclear waste storage, building material additives, water absorbents, soil treatment [REF], oxygen and contrast agent carrier for cancer treatment [10.1039/D3QI00169E] and others. Finally, they can be used as adsorbents for gas separation or storage, which is also of capital industrial interest.

Zeolite structures are mostly considered rigid structures in simulations, although some notable exceptions exist [10.1021/acs.chemmater.5b02103, 10.1126/science.abn7667], and most actually posess a flexibility window [10.1039/C003977B] which is exploited, especially during adsorption [10.1016/j.micromeso.2016.10.005]. In addition, T-atom substitution tends to modify only slightly the T-O bond length (Si-O: REF while Al-O: REF) and does not affect the O-T-O angles [REF]. As a consequence, the structure of a substituted zeolite can usually be well approximated by taking the reference structure of the topology (provided as a pure silicate by the IZA-SC) and replacing the relevant silicon atoms. However, not all placements of the substituents may correspond to existing zeolites

2.1.2 Aluminium placement

Cationic zeolites have some aluminiums as part of their structure, however their repartition among the T-atoms raises many questions.

Rules

The first rule, which is obeyed by almost all known zeolites, was discovered by Löwenstein [REF]. It states that no two aluminiums may be neighbours, in the topological sense of being bridged by an oxygen. This hard rule limits the maximum number of aluminiums to half the number of T-atoms in general, and less for topologies that contain odd cycles: for instance, if there is a cycle of 5 T-atoms, only two among them can be Al.

The amount of aluminiums in a zeolite is usually discussed in terms of Si/Al ratio. Löwenstein's rule thus limits the minimal Si/Al ratio to 1 in general, which corresponds to the maximum number of Al and thus of cations. Zeolites with high Si/Al ratios, usually called high-silica zeolites, contain few cations and tend to be hydrophobic [REF]; in general, they also tend to adsorb fewer molecules since cations are privileged adsorption sites.

While Löwenstein's rule is widely accepted as an axiom, and is mainly sustained by the absence of experimental synthesis of any zeolite with Si/Al ratio lower than 1, it is not currently proven. This comes from the cost of the few spectroscopic methods that allow identifying the precise locations of Al atoms: ^{27}Al and ^{29}Si magic-angle spinning NMR can be used to provide some limited information, while [10.1039/B203966B, 10.1039/B301634J] used UV-Visible-NIR spectroscopy on Co^{2+}-exchanged ZSM-5 with high Si/Al ratio to retrieve their Al distribution. But this method cannot be generalized to low-silica zeolites. DFT computation confirms the

rule in general, but also show potential violations for some zeolite topologies such as HEU [10.1016/j.mtcomm.2021.102028].

The second rule, proposed by [REF Dempsey et al, 10.1021/j100722a020] on the basis of FAU zeolites with varying Si/Al ratios, states that when possible, the number of next-neighbours Al atoms should be minimized to decrease the number of unfavourable interactions between close aluminiums. This rule is actually known to be violated in many real zeolites, including dealuminated FAU. Speculatively, this may come from the fact that the [REF Dempsey et al] made their observations on zeolites whose synthesis lead them to reach their global energy minimum, while many other synthetic routes can create structures trapped in a local energy minimum with respect to their Si/Al ordering.

Simulation methodologies

Overall, the lack of experimental evidence attesting to the precise position of aluminiums in zeolites make the theoretical prediction of such placement delicate, since it lacks verifiable comparison points. In particular, a crucial question is whether Si/Al ordering is truly guided by thermodynamical consideration or not [10.1007/s11814-021-0796-2]. The latter case would appear for zeolites whose structure is in a metastable state, which is likely to be the case for many real zeolites as evidenced by [10.1039/B301634J] in one of the few experimental studies tackling that very issue.

Even if the aluminium placement can be obtained as that which minimizes the global energy of the structure, this does not necessarily mean that the placement itself is periodic, nor that it follows the same periodicity as the unit cell. Yet, as it is difficult to provide a concrete result on the localisation of aluminiums without making such hypotheses, the studies in the literature only focus on the study of the placement of aluminiums in a single unit cell, supposing that the rest is periodic.

In these conditions, the last remaining question is the simulation methodology to use to find the global energy minimum of the structure. The most basic method consists in generating all possible configurations, computing their energy, and choosing that which minimizes the energy. This approach has actually been widely followed to find the preferred location of aluminiums when there are only one or two aluminiums per unit cell, using Density Functional Theory (DFT) to compute the energy [10.1007/s11814-021-0796-2, REF]. The involved computational cost makes it impractical for lower Si/Al ratios however, which are of major interest for adsorption purposes.

[REF Marie Jeffroy] proposed a more refined Monte-Carlo (MC) protocol for this problem, based on classical force fields for energy computation. Starting from a configuration with the target Si/Al ratio under Löwenstein's rule, each MC step consists in exchanging the positions of a Si and an Al while maintaining Löwenstein's rule, and accepting or refusing the step according to the Metropolis-Hastings criterion (explained in Section 2.2.1). For low Si/Al ratios, this algorithm requires adding some "restart" option that entirely repopulates the aluminiums at random under Löwenstein's rule, because that very rule can lead to blocked situation where no single Si/Al exchange is possible, as illustrated on Figure 2.1.

[REF Findley et al., 10.1021/acs.jpcc.8b03475] proposed an innovative method to assess the validity of theoretically-obtained aluminium placements, by studying CO_2 adsorption isotherms. They found that different Si/Al orderings lead to different isotherms, which, conversely, means

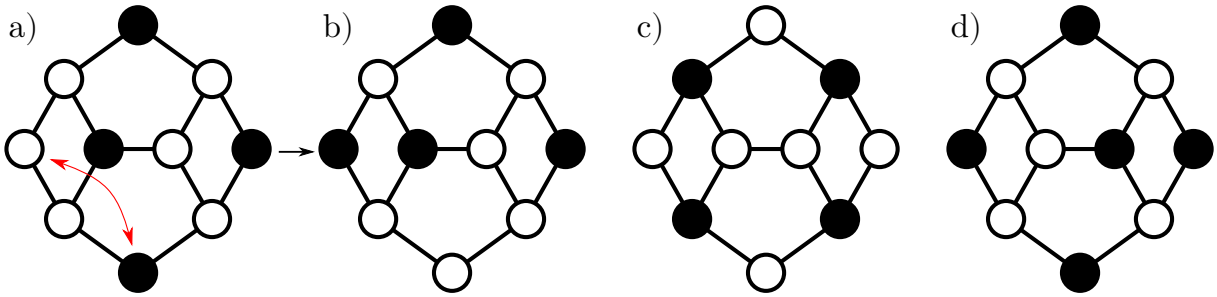


Figure 2.1: Schematic representation of aluminium placement in a zeolite fragment. Empty circles represent silicium, filled circles represent aluminium. Black edges represent -O- bridges. Löwenstein's rule forbids two aluminiums from being neighbors.
a) An exchange move between a silicium and an aluminium.
b) The result of the exchange.
c) A blocked configuration where no move is possible.
d) A configuration with one more aluminium, which cannot be reached from c).

that obtaining a precise experimental isotherm can allow retrieving the degree of Al ordering. In accordance to other results of Marie Jeffroy [REF], they conclude that some topologies (REF) are more likely to have their aluminiums placed at random compared to others (REF).

Hence, given the uncertainty that any aluminium placements obtained by thermodynamical simulation actually reflects those observed in real zeolites, we choose to consider that the aluminium placement in our zeolites is taken at random. In practise, each zeolite with a given Si /Al ratio will be modelled as the average of six zeolites of that topology with the same Si /Al ratio, all following Löwenstein's rule but having different Al repartition (taking symmetry into account as well). The results will be computed on each of these six models, and averaged to obtain the result for the zeolite.

To obtain the zeolite structures with random aluminium placements obeying Löwenstein's rule and the target Si /Al ratio, we use the following algorithm:

- The starting point is either the zeolite filled with Si if the required Si /Al ratio is above 3, else a zeolite filled with as many Al as possible using a greedy algorithm. That greedy technique consists in doing a graph traversal of the T-atoms, transforming each atom into an Al if it is not the neighbour of an Al, or making it a Si otherwise: if all cycles in the graph are even, this is guaranteed to provide an aluminium placement with the minimal Si /Al of 1.
- The main algorithm then alternates between adjusting the number of Al and changing their positions, for a few steps each, back and forth.
  - The position update follows the principle of Marie Jeffroy's MC scheme but disregarding the energy change: one Si and one Al are exchanged at random if the resulting structure still follows Löwenstein's rule.
  - The number of Al atoms is modified if the current Si /Al ratio does not correspond to the target: if it is too high, an Al is converted into a Si, and otherwise a Si that has no Al neighbour is converted into an Al.

- Since the situation can be blocked (see Figure 2.1), the algorithm sometimes restart from scratch, or sometimes does a partial restart by converting multiple Al atoms into Si.
- Configurations where the correct Si/Al ratio is reached are kept in a set. Each configuration is assigned a hash, which is simply a sequence of 0 if the T-atom is a Si or 1 otherwise, in a fixed order of the T-atom. To avoid considering symmetry-equivalent configurations as different, the signature of a configuration is computed as the lexicographically minimum of all the hashes corresponding to the application of a symmetry operation on the configuration. For each unique signature, a corresponding configuration is then retained, and the six models are taken at random among these configurations.

One last consideration is the choice of the supercell. It appears that, for a large number of zeolites topologies, the smallest unit cell cannot be filled with aluminiums so that it both obeys Löwenstein's rule and reaches the minimal Si/Al ratio for that topology. In the case of LTA for instance, the smallest compatible cell is a $2 \times 2 \times 2$ supercell of the smallest unit cell. As a consequence, the greedy algorithm used before is actually attempted on several supercells, and the one leading to the smallest Si/Al ratio is retained.

The full list of minimum Si/Al ratio found for each zeolite topology (except interrupted ones) and the respective supercell is presented in Table 2.1. The supercell is considered with respect to the idealized framework available for each topology on the IZA-SC database[2]. This table presents the lowest Si/Al ratio found, but apart for the cases where it is 1, the values here are not proven to be minimal, since the algorithm relies on random attempts. Similarly, the supercell used could be not minimal.

It is worth mentioning that a previous screening study of adsorption in zeolite by [10.1021/acsami.0c20892] claimed to use the lowest possible Si/Al ratios to screen through zeolites, but failed to reach the ones presented here, systematically in the case where a supercell needed to be used. Another [10.1021/acs.jpcc.3c06115, 10.1021/acs.jpcc.4c00066] alluded to that issue in the case of LTA, yet failed to identify some other topologies (ATT, JBY, LEV, LTL, MRT, OFF) which could reach an Si/Al ratio of 1. The choice of the supercell during aluminium placement is thus an easily overlooked, yet critical, point to consider.

2.1.3 Cations

The electroneutrality of the framework requires that each negative introduced by an Al substitution be compensated by the introduction of a cation in the framework. In many cases, Na^+ or H^+ are used during synthesis; they are sometimes replaced by other cations in later steps.

These cations have a primordial role for adsorption processes, because they constitute privileged adsorption sites for many small gases. The precise mechanism by which they contribute to the overall adsorption capacity of a material depends on their nature, their location, and the nature of the guest molecule however. Moreover, in addition to simple one-to-one adsorption cases of a small molecule on a cation, X-ray spectroscopy evidences the presence of adsorption complexes involving several cations and guest molecules at the same time [10.1039/C2CP23237E]. Finally, the very nature of the cations present in the structure can be quite diverse, ranging from

[2] accessible at https://europe.iza-structure.org/IZA-SC/ftc_table.php

Topology	Si/Al	Supercell
ABW	1	1×1×1
ACO	1	1×1×1
AEI	1	1×1×1
AEL	1	1×1×1
AEN	1	1×1×1
AET	1	1×1×1
AFG	1	1×1×1
AFI	1	1×1×1
AFN	1	1×1×1
AFO	1	1×1×1
AFR	1	2×2×4
AFS	1	1×1×1
AFT	1	1×1×1
AFV	1	3×3×2
AFX	1	1×1×1
AFY	1	1×1×1
AHT	1	1×1×1
ANA	1	1×1×1
ANO	1	1×1×1
APC	1	1×1×1
APD	1	1×1×1
AST	1	1×1×1
ASV	1	1×1×1
ATN	1	1×1×1
ATO	1	1×1×1
ATS	1	1×1×1
ATT	1	3×4×3
ATV	1	1×1×1
AVE	1	1×1×1
AVL	1	3×3×2
AWO	1	1×1×1
AWW	1	2×2×4
BCT	1	1×1×1
BEC	1.29	2×2×2
BIK	1.4	4×2×5
BOF	1.4	2×1×1
BOG	1.4	1×1×1
BOZ	1.88	1×1×1
BPH	1	1×1×1
BRE	1.29	4×2×4
BSV	1	1×1×1
CAN	1	1×1×1
CAS	1.4	1×1×1
CDO	1.62	4×2×2
CFI	1.46	2×5×1
CGF	1	2×2×4
CGS	1	1×1×1
CHA	1	1×1×1
CON	1.33	1×1×1
CSV	1.22	1×2×1
CZP	1	1×1×1
DAC	1.57	1×3×2
DDR	1.73	1×2×1
DFO	1	2×2×2
DFT	1	4×4×3
DOH	1.62	1×2×1
DON	1.29	1×1×1
EAB	1	1×1×1
EDI	1	4×4×4
EEI	1.44	1×1×1
EMT	1	1×1×1
EON	1.4	2×1×1
EOS	1.18	1×2×2
EPI	1.74	4×2×3
ERI	1	1×1×1
ESV	1.4	1×1×1
ETL	1.48	2×1×1
ETR	1	1×1×1
ETV	1.33	1×1×1
EUO	1.49	1×1×1
EWF	1.49	2×1×1
EWO	1.4	2×2×5
EWS	1.53	1×1×1
EZT	1	1×1×1
FAR	1	1×1×1
FAU	1	1×1×1
FER	1.57	1×1×3
FRA	1	1×1×1
GIS	1	1×1×1
GIU	1	1×1×1
GME	1	1×1×1
GON	1.29	1×1×1
GOO	1	1×1×1
HEU	1.25	1×1×2
IFO	1	1×1×1
IFR	1.29	1×1×2
IFW	1.29	1×1×1
IFY	1	1×1×2
IHW	1.55	1×1×1
IMF	1.55	1×1×1
IRN	1.19	1×1×1
IRR	1.6	2×2×2
ISV	1.31	2×2×1
ITE	1.56	1×2×1
ITG	1.33	1×1×2
ITH	1.33	1×2×1
ITR	1.33	1×1×1
ITT	1.56	2×2×3
ITW	1.4	1×1×1
IWR	1.33	2×2×2
IWS	1.27	1×1×2
IWV	1.62	1×1×2
IWW	1.33	1×1×1
JBW	1	5×4×3
JNT	1	1×1×1
JOZ	1.5	2×1×1
JRY	1	1×1×1
JSN	1	3×4×2
JSR	1.67	1×1×1
JST	2	1×1×1
JSW	1	1×1×1
JSY	1	1×1×1
JZO	1.37	1×1×1
JZT	1.35	1×1×1
KFI	1	1×1×1
LAU	1	2×2×4
LEV	1	3×3×2
LIO	1	1×1×1
LOS	1	1×1×1
LOV	1.25	2×2×1
LTA	1	2×2×2
LTF	1.27	1×1×4
LTJ	1	1×1×1
LTL	1	2×2×4
LTN	1	1×1×1
MAR	1	1×1×1
MAZ	1.4	1×1×2
MEI	1.43	1×1×1
MEL	1.67	1×1×1
MEP	1.71	1×1×1
MER	1	1×1×1
MFI	1.67	1×1×1
MFS	1.45	3×1×1
MON	1.67	2×2×1
MOR	1.67	1×1×2
MOZ	1	1×1×4
MRT	1	2×4×2
MSE	1.55	1×1×1
MSO	1	2×2×2
MTF	1.44	1×1×2
MTN	1.72	1×1×1
MTT	1.4	1×1×1
MTW	1.33	1×5×3
MVY	1	1×1×1
MWF	1	1×1×1
MWW	1.53	2×2×1
NAB	1.5	2×2×1
NAT	1.22	2×2×4
NES	1.52	1×1×1
NON	1.44	1×1×1
NPO	2	4×4×5
NPT	2	2×2×2
NSI	1.4	2×5×3
OBW	1.92	1×1×1
OFF	1	3×3×4
OKO	1.62	1×1×1
OSI	1	1×1×1
OSO	2	1×1×1
OWE	1	2×4×3
PAU	1	1×1×1
PCR	1.5	1×1×1
PHI	1	1×1×1
PON	1	1×1×1
POR	1	1×1×1
POS	1.29	1×1×1
PSI	1	1×1×1
PTF	1.5	1×1×1
PTO	1.4	1×1×1
PTT	1	1×1×1
PTY	1.5	1×1×1
PUN	1.25	1×1×1
PWN	1	1×1×1
PWO	1.5	1×1×1
PWW	1.5	1×1×1
RFE	1.4	1×1×1
RHO	1	1×1×1
RRO	1.25	2×1×1
RSN	1.59	4×1×4
RTE	1.4	1×1×2
RTH	1.53	2×1×3
RUT	1.57	2×2×3
RWR	1.6	4×4×1
RWY	3	1×1×1
SAF	1	1×1×1
SAO	1	1×1×1
SAS	1	1×1×1
SAT	1	1×1×1
SAV	1	1×1×1
SBE	1	1×1×1
SBN	1.5	1×2×1
SBS	1	1×1×1
SBT	1	1×1×1
SEW	1.36	1×2×1
SFE	1.33	1×1×1
SFF	1.46	2×1×2
SFG	1.47	1×2×2
SFH	1.29	5×1×2
SFN	1.29	1×5×2
SFO	1	2×2×4
SFS	1.67	1×1×1
SFW	1	1×1×1
SGT	1.49	2×2×1
SIV	1	1×1×1
SOD	1	1×1×1
SOF	1.5	1×1×1
SOR	1.18	1×1×2
SOS	1.4	2×4×3
SOV	1.29	1×1×1
SSF	1.25	2×2×2
SSY	1.33	1×1×1
STF	1.46	1×1×2
STI	1.25	1×1×1
STT	1.46	1×1×1
STW	1.5	1×1×1
SVV	1.33	1×1×1
SWY	1	1×1×1
SZR	1.25	1×1×2
TER	1.35	1×1×1
THO	1	2×4×4
TOL	1	1×1×1
TON	1.4	2×2×5
TSC	1	1×1×1
TUN	1.53	1×1×1
UEI	1	1×1×1
UFI	1.29	2×2×1
UOS	1.18	2×4×3
UOV	1.38	1×1×1
UOZ	1	1×1×1
USI	1	1×1×1
UTL	1.62	1×2×2
UWY	1.37	1×2×3
VET	1.43	1×1×1
VFI	1	1×1×1
VNI	1.73	3×3×1
VSV	1.78	4×4×1
WEI	1.5	1×1×1
YFI	1.4	1×1×1
YUG	1.56	3×2×4
ZON	1	4×2×2

Table 2.1: Minimal Si/Al and corresponding supercell obtained for all 239 non-interrupted topologies

monovalent alkali to polyvalent metal ions, and even molecular ions such as many of the organic structure directing agents (OSDAs) used during zeolite synthesis.

In order to perform numerical simulation of the adsorption process, it is thus necessary to have a model of the zeolite that places its cations in the correct positions. Fortunately, and opposite to the previous case of aluminiums, X-ray diffraction is a convenient technique which can be used to obtain the crystallographic sites in which the cations are located, as well as the respective occupancy of each such site.

2.2 Methodology for the prediction of cation placement

While experimental cation maps are available on some zeolite structures, two seminal problems remain. First, when screening through a large amount of zeolite topologies and Si/Al ratio, only a small fraction of structures have their experimental cation map available. Second, even with the crystallographic sites, not all cation repartitions that obey the given occupancies have the same energy. For example, the site I and I' of FAU being very close to one another, the occupation of a site I prevents either of the neighbouring sites I' from being occupied at the same time. Deciding on the exact location of cations in a zeolite model is thus no trivial matter.

2.2.1 Underlying algorithms

To obtain equilibrium data, such as average positions, on molecular systems, there are two general simulation frameworks which can be used: molecular dynamics and Monte-Carlo (MC) simulation. We only used MC and its variations, because they are the only strategy available to study a variable number of molecules, which is necessary for the simulation of adsorption as we will explain in more details in Section 3.2. The possible use of molecular dynamics and its variants will be discussed later in Section 2.5.

Canonical Monte-Carlo direct simulation

The Monte-Carlo (MC) method designates a large variety of algorithms which all converge towards their result by performing multiple random samplings. A wide subset of these form the Markov Chain Monte-Carlo (MCMC) schemes, which are a class of algorithms that aim at sampling a target probability distribution. In the context of statistical physics, such methods provide practical ways of obtaining macroscopic observable values from the simulation of the microscopic behaviour of particles. Indeed, an observable $\mathcal{O}$ can be computed as the ensemble average of its microscopic counterpart $\langle o \rangle$, which can be computed as an integral over the configuration space. For example, in the canonical ensemble where the number of particles, the external temperature and the volume of the system is fixed,

$$\mathcal{O} = \langle o \rangle = \frac{1}{Z} \int \mathrm{d}\boldsymbol{p}\; o(\boldsymbol{p}) \exp\left(-\beta U(\boldsymbol{p})\right) \tag{2.1}$$

where $U(\boldsymbol{p})$ is the energy of configuration $\boldsymbol{p}$ and Z is the partition function. In other words, $\mathcal{O}$ can be computed as an integral of values of o on a probability distribution of $\boldsymbol{p}$ given by $p \mapsto \frac{1}{Z} \exp\left(-\beta U(\boldsymbol{p})\right)$.

The direct evaluation of this probability distribution is impossible in general, because of the partition function Z whose value cannot be computed. Instead, the MCMC method gives a way to directly evaluate the observable $\mathcal{O}$ by providing a sequence of configurations $\boldsymbol{p}$ that follows the same probability distribution as the target. The integral can then be computed as simply the average value of o on this specific set of configurations $\boldsymbol{p}$.

The Metropolis-Hastings algorithm is the most common MCMC scheme used to provide such a sequence of configurations. It consists in starting from an initial state, *i.e.* an initial position for all the mobile particles of the system, and making it evolve through a sequence of steps. At each step, a trial move, *i.e.*, a modification of the state, is attempted and may, or may not, be accepted with a certain probability. In the case of a simulation in the canonical ensemble, the difference of energy ΔE between after and before the move gives the probability

$$P_{\text{accept}}(\Delta E) = \min\left(1, \exp\left(-\beta \Delta E\right)\right) \tag{2.2}$$

that the move be accepted. Intuitively, this means that any move that decreases the energy of the system is accepted, while the others are accepted with a probability that decreases as the energy difference increases, sharply at low temperature and more slowly at high temperature. In the limit of infinite temperature, all the moves are accepted; at zero temperature, only the moves that decrease the energy are. The details on how to compute energy are deferred to Section 2.4.

To avoid drift, the algorithm obeys detailed balance which, conceptually, represents the microreversibility of the simulated physical processes. In practise, this means that, for each MC move that brings the system from configuration $\boldsymbol{p}_{\mathcal{A}}$ to $\boldsymbol{p}_{\mathcal{B}}$, there must exist a "converse" MC moves that can bring the system from $\mathcal{B}$ to $\mathcal{A}$ such that $\pi_{\mathcal{A}} P_{\mathcal{A}\to\mathcal{B}} = \pi_{\mathcal{B}} P_{\mathcal{B}\to\mathcal{A}}$ where $\pi_{\mathcal{X}}$ is the stationary probability of state $\boldsymbol{p}_{\mathcal{X}}$ and $P_{\mathcal{X}\to\mathcal{Y}}$ is the probability of the move from $\boldsymbol{p}_{\mathcal{X}}$ to $\boldsymbol{p}_{\mathcal{Y}}$. In the canonical ensemble, $\pi_{\mathcal{X}}$ is equal to to $\frac{1}{Z}\exp\left(-\beta U\left(\boldsymbol{p}_{\mathcal{X}}\right)\right)$, while the probability of transition is $\min\left(1, \exp\left(-\beta \Delta E\right)\right)$. Hence, detailed balance simply translates to the algorithmic condition that the probability of choosing the trial move $\mathcal{A} \to \mathcal{B}$ must be equal to that of choosing the trial move $\mathcal{B} \to \mathcal{A}$, irrespective of the probability of accepting these moves.

The MC moves which are used for simulations in the canonical ensemble are usually the following:

- translation: displace each atom of the particle by the same vector, taken uniformly at random in a sphere.
- rotation: rotate the particle around one of its atoms (called the "bead") along three uniformly random Euler angles, possibly constrained to some values only, most often all values under a maximum angle.
- reinsertion: remove and reintroduce the particle uniformly at random in the system.

Overall, the algorithm thus consists in a sequence of steps, each of which is an attempt to displace a cation either locally (translation) or not (reinsertion) and is accepted or not depending on the difference of energy the move generates. Configurations corresponding to neighbour steps are thus either identical if the move was refused, or only separate by one atom move, so they are strongly correlated. To reduce computational cost, the computation of the microscopic

state function o is thus only realized once every cycle, each cycle corresponding to a number of steps. We fix the number of steps per cycle to 100.

To satisfy detailed balance, the sphere in which the translation vector is taken must have a fixed radius, the bead must be fixed and the angle constraints must be symmetric. However, the best maximum translation length $d_{\max}$ and maximum angle $\theta_{\max}$ are those that yield a MC move acceptance rate of 1/2. Since these optimal values depend on the precise simulation, it is customary to actually make these two constraints evolve along the simulation until reaching an acceptance rate of 1/2. To do so, at the end of each cycle, they are updated with the following formulas:

$$\begin{aligned} d_{\max} &\leftarrow \text{clamp}\left(d_{\max} \times \left(1 + (\mathrm{T}_{\text{ratio}} - 1/2) \times \sqrt{\frac{10}{99+i}}\right),\ 0.1\,\text{Å},\ 3\,\text{Å}\right) \\ \theta_{\max} &\leftarrow \frac{i-1}{i} \times \theta_{\max} + \frac{\mathrm{R}_{\text{ratio}}}{i} \times 120^{\circ} \end{aligned} \tag{2.3}$$

where i is the number of the current cycle (starting at 1), $\mathrm{T}_{\text{ratio}}$ (respectively $\mathrm{R}_{\text{ratio}}$) is the ratio of accepted over attempted translations (respectively rotations) in the entire simulation, and clamp(x, m, M) is x if $m \leq x \leq \mathrm{M}$, otherwise m if $x \leq m$, otherwise M (if $x \geq \mathrm{M}$). Having $d_{\max}$ and $\theta_{\max}$ change across the simulation breaks detailed balance, which is theoretically forbidden; yet it does not actually pose a problem in because the formulas used ensure that the two values evolve smoothly towards the optimum, and become almost constants for long enough simulations.

At the beginning of each step, an MC move is chosen at random. In our simulations, all three moves (translation, rotation and reinsertion) are chosen equiprobably. For the particular case of the movement of mono-atomic species, such as most cations in zeolites, the rotation MC move is useless and thus not used, while the reinsertion move collapses to a simple translation with infinite $d_{\max}$, both moves being chosen equiprobably.

The Metropolis-Hastings algorithm thus provides a way to walk through the configuration space. By privileging moves that decrease energy, the system spends more steps in low energy regions; by yet allowing opposite moves sometimes, the scheme ensures that all the configuration space is eventually visited. Hence, the algorithm guarantees ergodicity: in an infinitely long simulation, the fraction of steps spent in a part of the configuration space is proportional to its volume.

Combined aluminium and cation placement

Specifically in the case cationic zeolites, [Marie Jeffroy REF] proposes a simulation method to obtain both the placement of cations in the structure and the location of aluminium atoms across the T-sites. This consists in running an MC simulation where the cations are allowed all previously mentioned MC steps, while the aluminiums are only allowed exchanging positions with siliciums, or a restart step that randomly positions aluminiums and siliciums, both while obeying Löwenstein's rule as explained in **??**.

The statistical ensemble simulated in this setting includes both aluminium and cations, which means that the distribution sampled through such a simulation method represents the equilibrium for both species. From a chemical point of view, this situation is representative of a framework whose synthesis was done with that very cation, and was thermodynamically-driven. While this does correspond to reality in some cases, there is also evidence that the

synthesis is kinetically-driven in many other occurences: for such situation, obtaining the aluminium positions from statistical equilibrium is not representative of the actual framework. And in yet other situations, the cation is exchanged post-synthesis, which decorrelates the aluminium placement in the framework from the cation used for adsorption purposes.

Nonetheless, even in those cases, there is no less value in the aluminium placement obtained from such a hybrid strategy than from random sampling of the T-atoms. In fact, by virtue of the added dimensions of the free energy landscape, a kinetically trapped cation may move more freely following the rearrangement of the aluminiums in the structure, which can make the entire simulation converge more efficiently. If not for the hope of obtaining more relevant aluminium placements, this combined strategy remains valuable for its diminished computationally cost compared to parallel tempering, explained later in Section 2.2.2.

Unfortunately, the fact that the framework is not strictly constant as before prevents the use of precomputed energy grid, later discussed in Section 2.4.4. This was not a problem in the context of [Marie Jeffroy]'s study because the force field they used was such that a silicium-aluminium swap would not change the van der Waals interactions, but this is not the case of most force fields, including the one we use in our study [BoulfelfelSholl2021]. Being unable to precompute the short-term energy contribution of the framework would thus make the simulations significantly slower, tipping the balance against the use of this method.

Site-hopping

Cations are free to move in a canonical MC simulation with translation moves, but some reconfigurations can be problematic. In particular, the cations may not have enough energy to overcome the barriers inherent to the structure of the zeolitic framework in any reasonable simulation time. For example, a cation trapped in the site I of a FAU zeolite will never manage escape into a site III in a room-temperature simulation of reasonable length without a reinsertion move: this simply comes from the high activation barrier it needs to overcome while physically moving out of the sodalite cage. Reinsertion moves are thus useful to allow large configuration movements to occur without having to pass through physical energy barriers.

However, reinsertion consists in attempting to displace a cation to a random position in the framework, which, in the vast majority of cases, ends up rejected because it is too energetically unfavorable. A better move would consist in displacing the cation towards a favorable position, to increase the likelihood of the move being accepted: this is called biasing, which requires unbiasing the acceptation rate to maintain micro-reversibility.

Taking this idea to the extreme, all cations can be forced to sit on one of a fixed set of possible positions, called a site, and to only move by hopping onto another of these positions, irrespective of the distance with the previous position. One hop is then accepted or rejected based on the Metropolis-Hastings criterion, like any other MC simulation. This "site-hopping" approach entirely bypasses the issue of spatial energy barrier since the cations directly hop onto a site, without having to move through an unfavorable region to access it. Hence, it makes this kind of simulation particularly well adapted to sharp energy landscapes, like that of cations in zeolites.

A major constraint of this simulation method however, is that it requires knowing the cationic sites, which is precisely the purpose of the current development. One could weaken this constraint by allowing many pseudo-sites, for instance taking all the points of a regular grid, but

this basically amounts to reverting to reinsertion moves without sites, which defeats the purpose of this protocol. Site hopping is thus useful in the later stages of cation placement, once the positions of the sites have already been identified, in order to evaluate their population.

2.2.2 Meta-algorithms

The previous algorithms allow sampling the configuration space at a fixed temperature. However, even with reinsertion or site-hopping, only one particle can move at a time, which is sometimes not enough. For instance, in FAU zeolites, filling both sites I' when starting from an initial configuration where only the central site I is occupied requires that both the central cation move to the edge of the cage and that an external cation be set in the opposite window. Since both movements require passing through an activation barrier, the probability of it happening naturally is low, and thus requires particularly long simulation times to be observed. More generally, collective movements can remain difficult.

To circumvent this issue, the temperature of the simulation can be raised: this mechanically makes all MC steps more likely to be accepted, so higher-energy regions become more reachable. At the same time, a simulation at a hot temperature is much less likely to explore the details of the energy landscape enough to fall into the energy minima of interest, since the accessible configuration space became so much larger. Several meta-algorithms rely on the previously explained algorithms but use different temperatures in different settings, to attempt finding the optimal compromise between wide and detailed explorations.

Simulated annealing

The simplest idea consists in running the simulation while smoothly varying the temperature between cold and hot, alternating between increasing and decreasing phases, possibly interspersed with plateaus. This is called simulated annealing, because of the similarity with the recrystallization method used in metallurgy.

Parallel tempering

Simulated annealing explores the configuration space by alternating between cold phases, where the exploration is detailed but localized, and hot phases, where the exploration is fast but superficial. Parallel tempering, also called "replica exchange" proposes doing both phases at the same time.

In more detail, a parallel tempering simulation consists in running several MC simulations, the "replicas", in parallel, each at a different temperature. The temperatures are taken between room temperature and a hot temperature. An extra MC step is introduced, called the "exchange" step, which consists in exchanging the configurations of two replicas with neighbour temperatures. As with any MC step, the Metropolis-Hastings algorithm ensures that the exchange is accepted only if the difference of energy resulting from this exchange is not too high: as a consequence, for the exchange step to be accepted regularly, the temperatures must be taken so that the typical energies of the system between neighbour temperatures overlap, as shown on [FIGURE].

In this strategy, the coldest simulation is constantly exploring a local free energy minimum, while the hottest simulation easily accepts MC moves and thus walks through the entire configuration space. By exchanging configurations, new configurations regularly trickle down from the hot simulations to the colder ones, while the general Metropolis-Hastings strategy ensures convergence towards a global minimum. It is also more efficient than simulated

annealing since the exploration of the configuration space is not constrained by the small duration of the hot phase. Parallel tempering is therefore the method of choice used to obtain convergence when facing difficult energy landscapes such as the one encountered for cation placement.

Naturally, its main downside is its computational cost, proportional to the number of simulations and thus effectively related to the difference between the highest and the lowest temperatures. The choice of intermediate temperatures must be guided by an initial discovery of the typical energies encountered by system at each temperature, which requires a few dedicated simulations prior to the parallel tempering. Finding the optimal number of simulations requires striking a balance with the proportion of accepted MC exchange steps, whose impact on the overall simulation convergence can be difficult to measure. Additionally, simulations with neighbour temperatures must regularly synchronize to attempt an exchange, which strongly limits the amount of parallelization usable in the implementation of parallel tempering.

"Shooting star" methodology

Previously exposed methods such as simple canonical MC, simulated annealing or parallel tempering focus on making the system (or at least, one of its replicas) reach equilibrium after a certain number of initial steps, so that the rest of the simulation occurs at equilibrium. This is crucial for the problem of sampling the equilibrium distribution of states in general. For the case of cation placement in zeolites however, the observed crystallographic positioning of the cations indicates that this distribution is sharply peaked: only a discrete number of sites can be occupied, and the question is to find their positions and occupancy. As a consequence, we can use another, simpler, strategy, which is potentially less efficient for the actual sampling of the distribution, but can uncover relevant local minima faster.

This new methodology takes the same simulation elements as the ones previously detailed, but arranges them in a different fashion. We call it "shooting star" because it can be decomposed into one hot simulation, like the bright head of the meteor, and a series of cold simulations that stem from it, like the cooling incandescent debris left in its wake.

The first element is the exploration of the phase space. In order to fully explore it, a naive approach consists in doing steps, each of which randomly reinserts a particle in the system. Without the possibility for step rejection, this leads to sampling extremely unlikely states almost all the time however. We propose to perform global exploration by simply running a canonical MC simulation at a very high temperature, for instance 2000 K. Such an approach makes most kinetic obstacles irrelevant, but still prevents absurd configurations like when two atoms collide. This MC simulation is simply called the hot simulation.

The second element is the local minima exploration, which is grafted onto the previous one: every Δ_{spawn} cycles of the hot simulation, a snapshot of its current configuration is taken and used as the starting point for a cold simulation, *i.e.* another canonical MC simulation that runs at room temperature. Δ_{spawn} is chosen so that two consecutive starting points should be uncorrelated. The higher the temperature of the hot simulation, the less correlated its consecutive steps are, so Δ_{spawn} can actually be very low. Each cold simulation runs for M_{init} initial steps, that allow the system to cool down, and M_{prod} production steps at the target room temperature.

Similar to the previous accelerated methodologies, only a fraction of the explored states can actually be used to perform statistics. In the case of simulated annealing, it corresponded to the cold regions of the simulation; for parallel tempering, only the coldest replica is useful to obtain statistics at that temperature; for the current methodology, the relevant steps are the production of the cold simulation. The main advantage of the "shooting star" protocol resides in the efficient exploration of the configuration space, rooted in the hot simulation that is independent from the cold ones. Conversely, it lacks the physical meaning of simulated annealing – which is the numerical counterpart to the annealing process used in metallurgy – or the time-reversibility of parallel tempering and bare canonical MC.

In terms of parameters, the "shooting star" method requires choosing T_{hot} the hot temperature, T_{cold} the cold one, N_{cold} the total number of cold simulations to launch, the previously mentioned M_{init}, M_{prod} and Δ_{spawn}, as well as the number N_{init} of initial steps of the hot simulation before launching the first cold simulation. Of all these parameters, T_{cold}, N_{init} and $N_{prod} = M_{prod} \times N_{cold}$ must also be chosen for a single canonical MC simulation and Δ_{spawn} can always be taken around 100 (its value has little effect on overall performance and quality as long as it is not too low nor too high). The extra parameters T_{hot} is equivalent to the hottest temperature used in simulated annealing and parallel tempering. Finally, M_{init} must be evaluated by an initial run, similar to how the intermediate temperatures used in parallel tempering are obtained or the slope and plateau lengths of simulated annealing are usually chosen. The complexity of the overall setup is thus similar to that of simulated annealing.

2.2.3 Extraction of sites and population from simulation

Assuming we do not know the position of the cation sites, a canonical MC simulation (possibly with "shooting star" or another meta-algorithm) yields a sequence of cation configurations sampled at room temperature. This sequence can be flattened into a map of the zeolite framework, with one additional point at each position occupied by a cation in one of the steps. Looking at this map allows qualitatively identifying the crystallographic sites as the zones where the density of points is the highest. In order to get quantitative information on the site localization and population, we devise a clustering algorithm that extracts the sites from the previous density map.

The first step consists in dividing the space by a regular grid, whose voxels have the same angles as the unit cell but possibly different lengths, taken to be closest to a target value around 0.15 Å. To each voxel is then attributed the number of cations encountered within, across the recorded simulation cycles. This first step thus bins the density onto a regular grid.

In the second step, the bins are sorted by decreasing order of their value. Each voxel is then taken in order: if it is closer than a set distance to a previously recorded site, then the location of the site is updated as the average of its current position and that of the voxel, weighted by the respective density of both; the density of the voxel is then added to that of the site. Otherwise, if it is far enough from all existing sites (we use a minimum distance of 1.6 Å), it is added to the (initially empty) list of recorded sites, along with its density. This yields a series of sites and their respective occupancy. To be more efficient, the main loop early stops when the density of the inquired voxels go below a threshold value, taken as the maximum density of all bins divided by 100: as a consequence, the occupancies are corrected by a multiplicative factor so that the sum of the occupancies of all sites correspond to the number of cations. We note

that this should not be required in theory, but it is the necessary fix to a double discretization artifact, once inherent to the MC scheme, and one due to our binning.

We observe slightly better results by sorting the bins according the decreasing order their smoothed counterparts, where smoothing is done by convolving the density map by a small Gaussian, of standard deviation the size of the diagonal of a voxel. Indeed, this erases some small statistical fluctuations that may change the precise order of the voxels.

This initial list of sites is then symmetrized, to obtain a common list of sites irrespective of the aluminium placements. To do so, a strategy similar to the hashing of aluminium placements (explained in Section 2.1.2) is used: for each site, its image is computed for each symmetry of the framework. If one of the image is too close (we use a minimum distance of 1 Å) to one of the previously recorded sites, they are merged; otherwise, the one closest to a fixed arbitrary reference is recorded. Hence, the initial sites are "folded" to yield a list of symmetrically unique sites. Those are then "unfolded" again by applying all symmetries to obtain the list of all sites in the framework.

The final list of sites is such that for any site, all its symmetric images are also sites, and two sites cannot be closer than 1 Å. Note that the minimum distance of 1.6 Å used as a minimum between two starting points of initial sites does not necessarily transfer to the final sites, because of both the growing of the initial sites, and because of the merging operated during the "folding" part of resymmetrization.

2.3 Results

This section presents the cation location and population results in the studied zeolites. We focus our attention on the FAU (faujasite) zeolite first, because it is an already well-studied case that highlights the main difficulties encountered when attempting to locate cations.

2.3.1 Case study: FAU

The FAU zeolite is one of the most well-known zeolites used in the industry. It is composed of large cages, of 12 Å in diameter, separated by 12-membered rings, and small sodalite cages separated by hexagonal prisms. A detail of FAU with Si/Al = 1 is illustrated on Figure 2.3a.

Site localization

As previously explained, the sites are located by running "shooting star" simulations. We use the force field of [REF BoulfelfelSholl2021] with the hot simulation running at 2000 K for 20 000 cycles after 2000 initialization cycles, using both translations and random translation (*i.e.* reinsertion without rotation) moves equiprobably. 200 cold simulations spawn, once every 100 hot cycles, running for 200 production cycles after 9800 initialization cycles. Only the production cycles are used to record the density map.

The convergence of the "shooting star" simulation can be analyzed on Figure 2.2 in the particular case of `FAU_1_6e921e59`, which is simply the faujasite framework with Si/Al = 1 (`6e921e59` is the hash of the only aluminium placement compatible with Löwenstein's rule, as explained in Section 2.1.2). This figure mostly focuses on the initialization cycles of the cold simulations, to study the global convergence of the simulation.

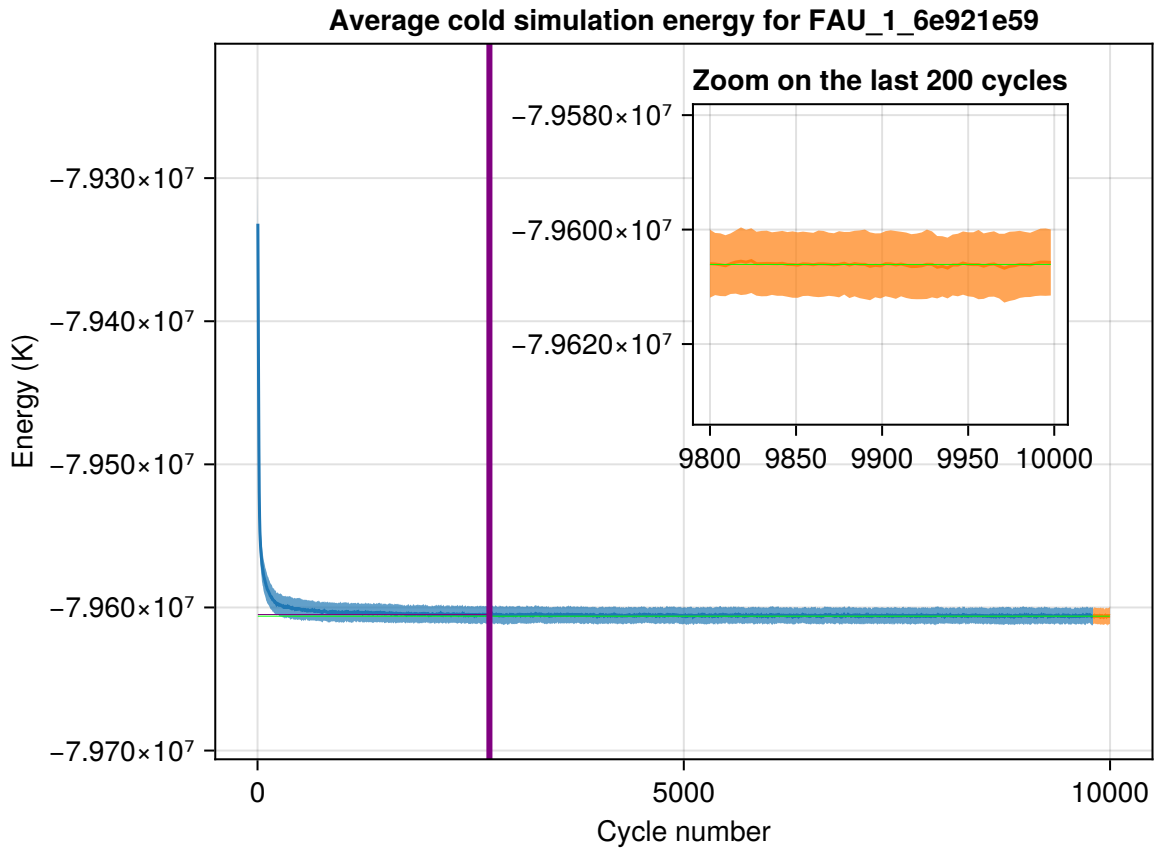
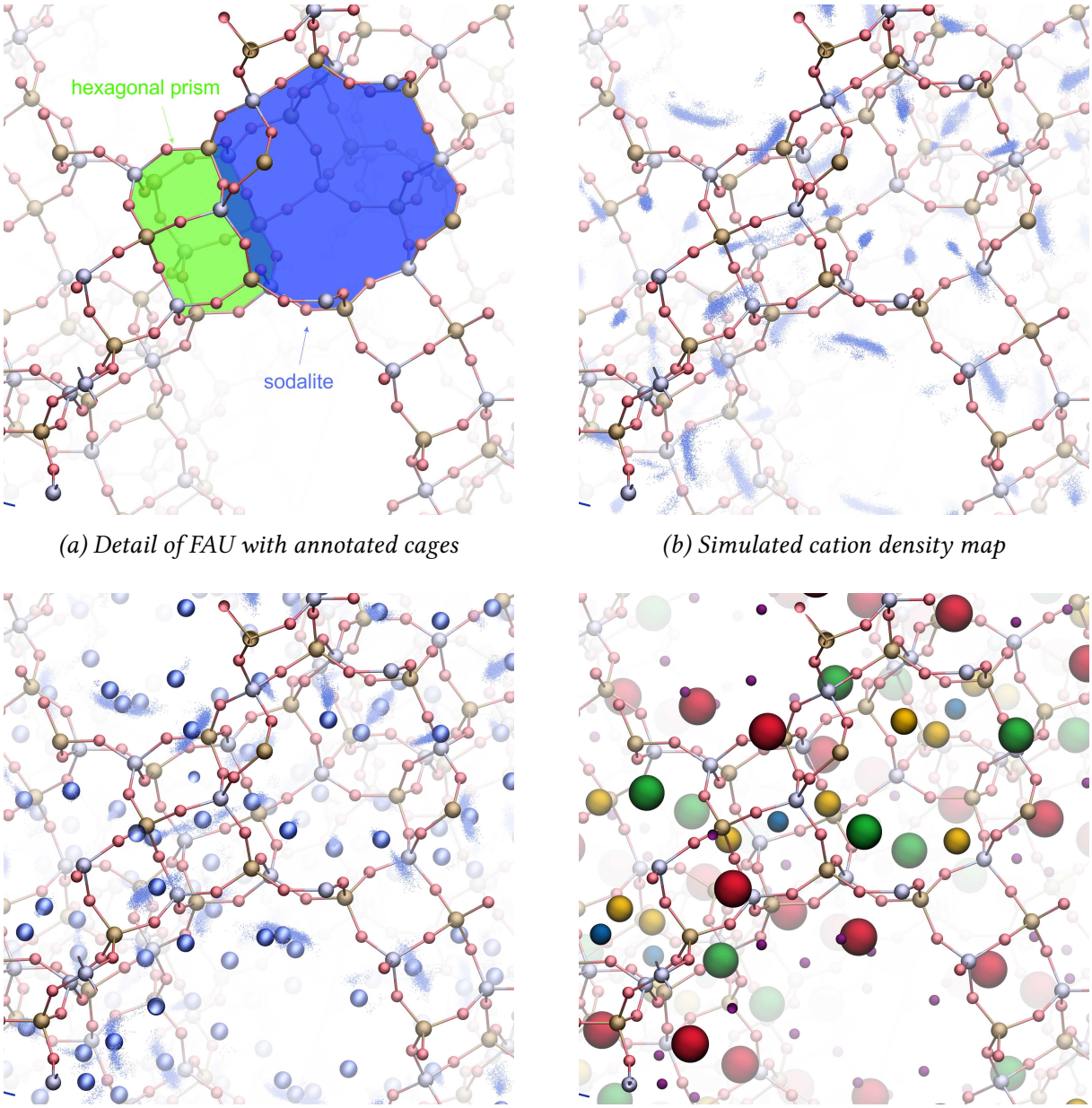


Figure 2.2: Convergence of the "shooting star" simulation of Na^+ in FAU with Si/Al = 1
The dark blue line represents the average energy of the cold simulations for each cycle number. The light blue halo around it represents plus or minus its standard deviation across the cold simulations. The inset focuses on the production cycles using dark and light orange for the same purpose. The vertical purple line identifies the cycle at which the simulation is estimated to have converged. The thin horizontal green line is the average of the energy of the cold simulations during the production cycles. The thin horizontal purple line is the average energy of the cold simulations at the estimated convergence cycle.

Assuming that the different cold simulations are uncorrelated, the standard deviation of their energy at a fixed cycle number gives an information on the statistical noise of the system, which appears to be approximately constant on the figure. Convergence is assessed based on the lack of evolution of the energy averaged across the cold simulations: the vertical purple line marks the first cycle from which the slope of this average energy (smoothed on a few cycles to avoid artifacts) remains below a target bound.

The vertical separation between the green (final energy) and the purple (convergence energy) horizontal lines is another way to quantify the error on the estimation of the convergence of the simulation. In the current case, the difference is well below the statistical noise, which is a further sign of convergence.

Such a simulation runs for all structures corresponding to the reference topology: in the case of FAU, and for all other topologies that can reach Si/Al = 1, this means one simulation for



(a) Detail of FAU with annotated cages

(b) Simulated cation density map

(c) Extraction of sites from the density map

(d) Cationic sites after resymmetrization

Figure 2.3: Cation localization for FAU

Si/Al = 1 and six (with six different aluminium placements) for Si/Al equal or approximately equal to 1.23, 1.4, 1.7, 2, 3, 5 and 10, hence 43 different "shooting star" simulations. The positions of each cation on each recorded cold production cycle across these simulations are flattened into a single density map of the framework, which serves as the basis for the location of cationic sites as explained in Section 2.2.3.

Direct site population

The site locations are obtained from the flattened density map across different Si/Al ratios, but the resulting occupancies are meaningless. To retrieve site population, the volume of the unit cell of the zeolite is divided into volumes centered around each site, such that each point in a volume is closest to the corresponding site than to any other site. This operation is actually the Voronoi decomposition of the crystal space around the sites.

Cationic site name	Number of sites per unit cell	Location	Color in figure 2.3d
I	16	At the center of the hexagonal prism.	Blue
I'	32	In the middle of the 6-membered window between the hexagonal prism and the sodalite cage, slightly displaced toward the sodalite cage.	Yellow
II	32	In the middle of the 6-membered window between thesodalite cage and the supercage.	Green
III	48	In the supercage, atop the 4-membered window between the sodalite and the supercage.	Red
III'	96	In the supercage, atop the 4-membered window between the hexagonal prism and the supercage.	None
J	96	In the 12-membered window between two supercages, atop the oxygen of the 4-membered window between the sodalite and the supercage.	Purple

Table 2.2: Description of cationic sites of FAU

To assert their physical meaning, the Voronoi cells are intersected with solid spheres centered on the site and of radius 2 Å. Thus, any point in space is either in one of the truncated cells, in which case it is associated with a site closer than 2 Å, or it is not associated with any site. Once established for a given topology, this map is used for all its corresponding structures (with different Si /Al ratios and aluminium placements).

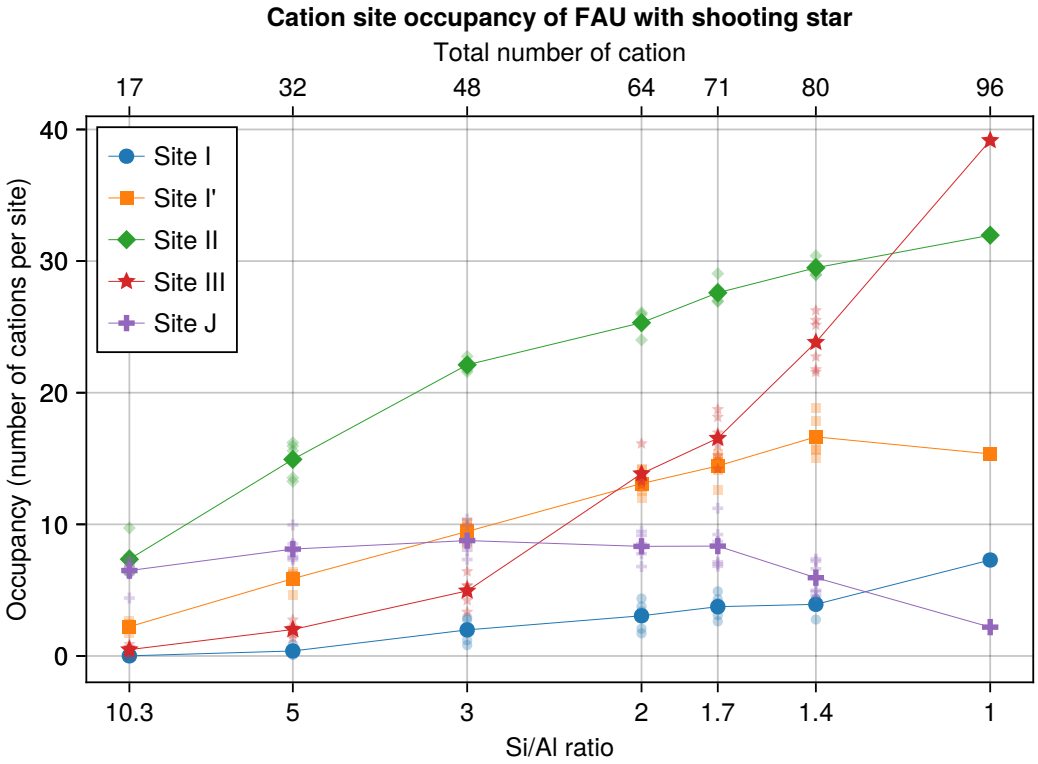
In the literature, the different sites of FAU are well-known and have been given names. Those detected by the previous algorithm are listed in Table 2.2 along their usual names and represented in Figure 2.3d, using one color per kind of site and with size proportional to their occupancy. Site III' is also mentioned because it is present in other studies, although it was not detected by the algorithm. By contrast, site J is not usually mentioned in studies of faujasite, presumably because it is unoccupied except in the presence of water,[62,63] so its apparition here is possibly an artifact of the force field we use, from Boulfelfel et al. [64].

With this information, the attribution of the cations to their site can be operated for each cold production cycle of the "shooting star" simulation. The average result is presented in Figure 2.4a.

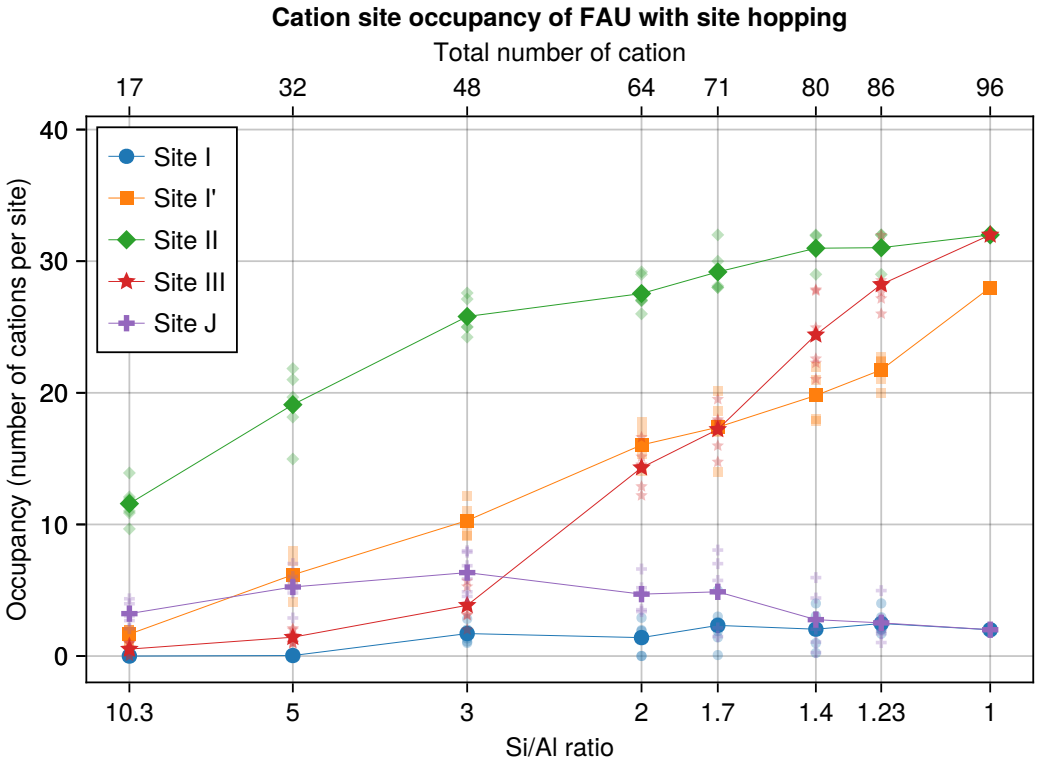
Population combined with site hopping

Alternatively to using the "shooting star" simulation results to establish the site population, the resulting site positions can be used to launch a site hopping simulation. Using 1 million production cycles (after 2 million initialization cycles) at 300 K with an output every 1000 production cycles takes up to three minutes per structure and yields a slightly different site population presented in Figure 2.4b.

The main difference comes from the occupation of the sites I' at low Si /Al, which becomes almost maximal for the lowest ratio since since there are 32 sites I'. It appears that the 'shooting star' simulation alone does not correctly reproduces this trends, while site hopping manages to

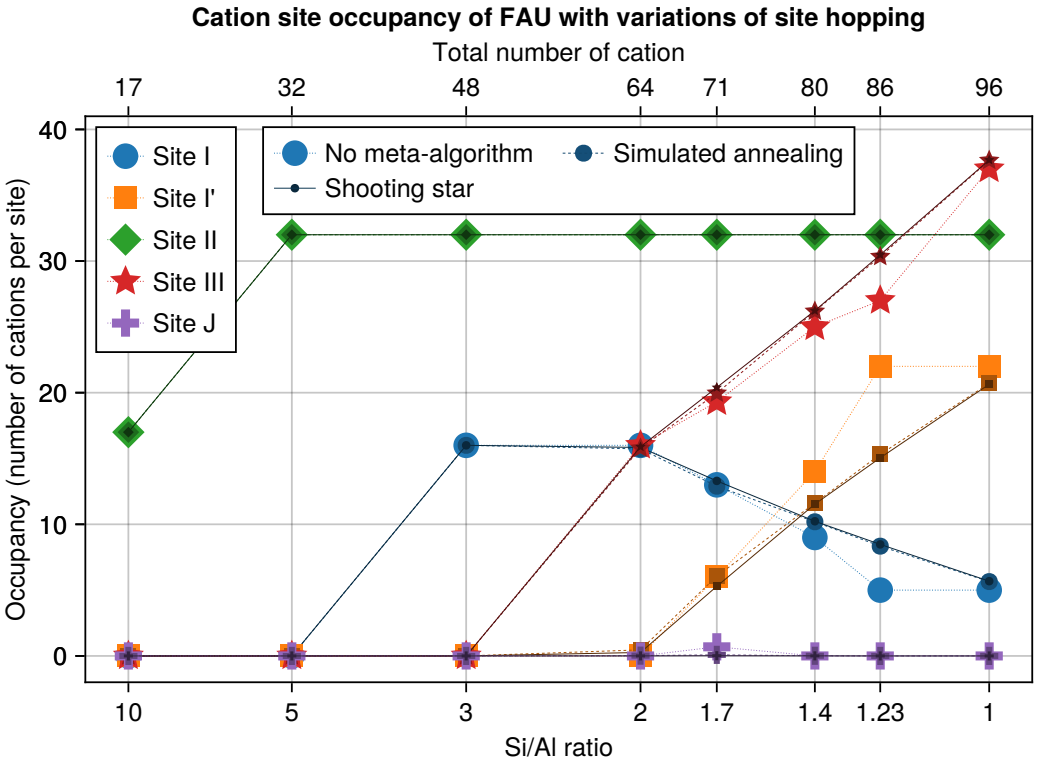


(a) Data from "shooting star" simulation

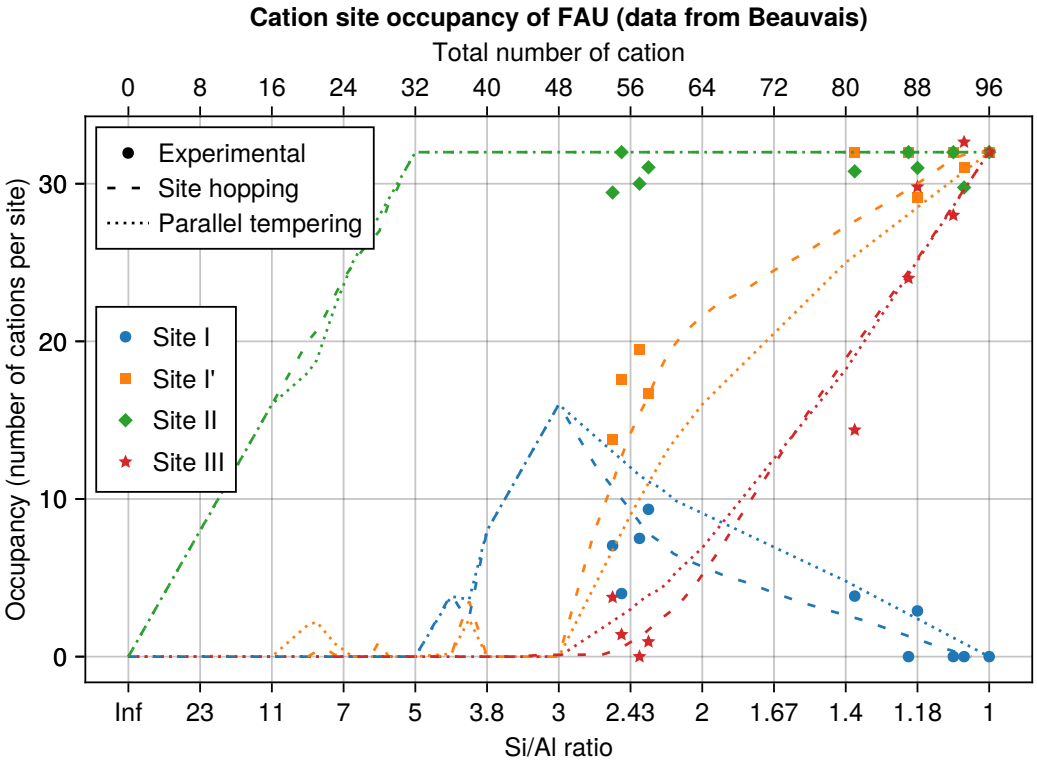


(b) Data from site hopping

Figure 2.4: Population of each site of FAU with force field from Boulfelfel et al. [64] Transparent symbols indicate the values for each of the 6 explicit aluminium placements.



(a) Data from site hopping with and without meta-algorithms



(b) Data from Christèle Beauvais's Ph.D. dissertation

Figure 2.5: Population of each site of FAU with force field from Di Lella et al. [62]

explore more combinatorial configurations of site occupancy, thus yielding more faithful site populations.

To compare those results with the literature, Figure 2.5b reproduces the site population of FAU obtained by Christèle Beauvais in her Ph.D. dissertation, superposing experimental previous works as well as simulation data obtained from both parallel tempering and site hopping, using the force field from Di Lella et al. [62]. Site J is not used in that work. Nonetheless, while the general trend of site occupations bears some similarity, the evolution also has some marked differences, notably for site I around Si/Al = 3.

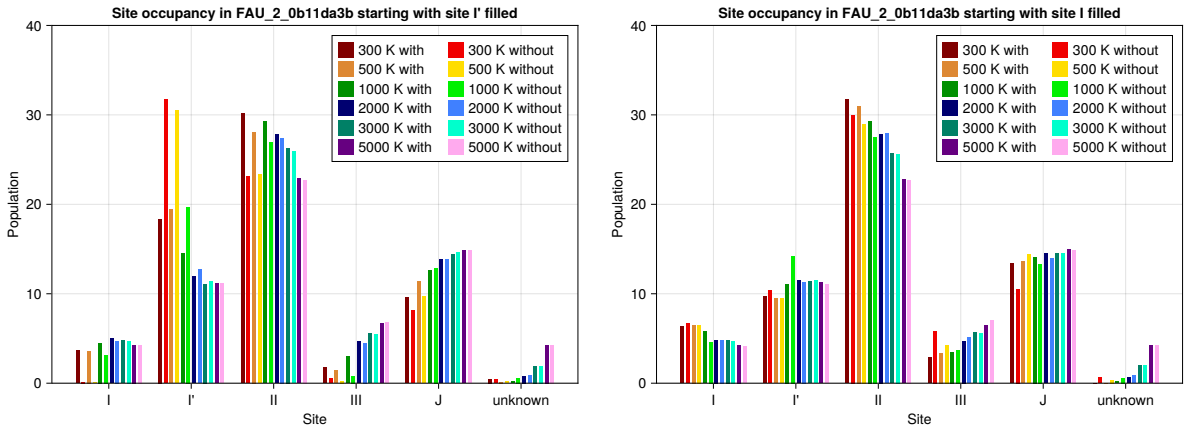
I investigated this discrepancy by running a new site hopping simulation, using the same force by Di Lella et al. [62]. The resulting site population, shown in Figure 2.5a (big light symbols), is indeed more faithful to the original, but still bears a notable population difference for the lowest Si/Al ratios, where site I' fails to reach its expected occupancy of 32. In this force field, the aluminiums and siliciums of the framework are represented by an identical T-atom, whose charge depend on the Si/Al ratio, hence there the absence of the transparent symbols indicating the result of each explicit aluminium placement.

Much like any base MC scheme, the site hopping algorithm can be combined with a meta-algorithm to improve its convergence. It is thus possible to run a simulated annealing version, by varying the temperature during the simulation, or even a "shooting star" simulation of site hopping. For simulated annealing, the temperatures ramped down 400 times, each time starting from 3000 K to 2000 K, then 1000 K, 300 K and finally 300 K again, equally spaced by 1000 cycles for each sub-ramp, and recording the site repartition at the end of the last 1000 cycles. For the "shooting star" variant, this hybrid simulation ran for 2 million hot cycles at 2000 K, spawning a cold simulation every 1000 cycles, each running for 20 000 cycles following 1 500 000 initialization cycles at 300 K.

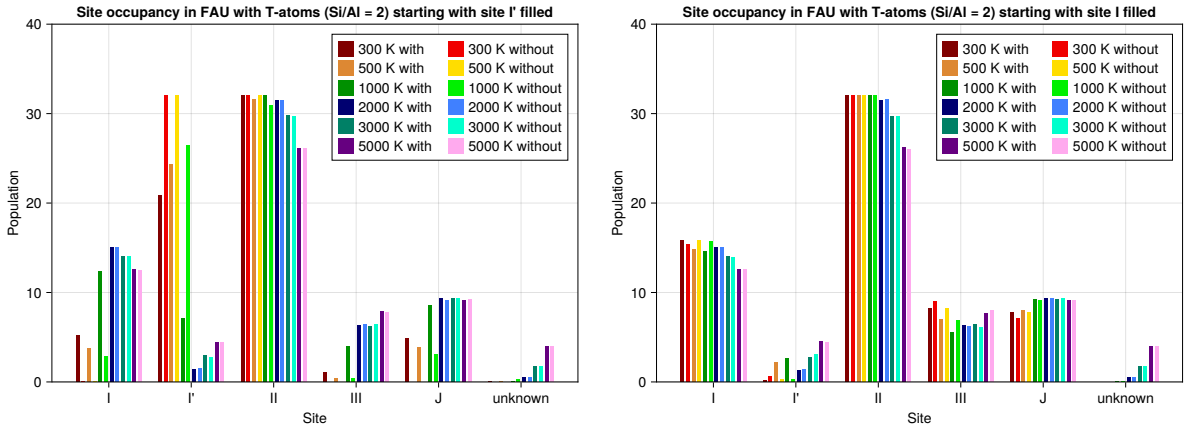
The results are superposed to the simple site hopping computation on Figure 2.5a (medium symbols for simulated annealing, small dark symbols for "shooting star"). The two evolutions with meta-algorithm superpose almost exactly, which shows that simulated annealing is complex enough to ensure convergence of site hopping simulations. These evolutions are smoother compared to the bare simulation without meta-algorithm, and show a consistent and marked difference around Si/Al = 1.23 in particular: this highlights the interest of using meta-algorithms to ensure proper convergence of even site-hopping simulations. These results still differ from those of Christèle Beauvais at low Si/Al remain, which is unexpected; it might be due to a difference in the exact positions used for the site, or even in the faujasite model used.

"Shooting star" simulation calibration

The temperature of the hot simulation, as well as the inclusion of reinsertion MC moves, was determined by running "shooting star" simulations at other temperatures with and without MC moves. Two starting positions of the cations were used, using a single structure with Si/Al = 2 hence 64 cations: both configurations start by filling the 32 sites II, then the first configuration fills the 32 sites I' (configuration I') while the second fills the 16 sites I and then selects 16 sites III at random (configuration I). As shown on Figure 2.3d, every site I is sandwiched between two very close sites I', so a site I' and its neighbor site I cannot be occupied by cations at the same time. These two starting configurations are thus extremes, in the sense that crossing



(a) Sites population with force field from Boulfelfel et al. [64]



(b) Sites population with force field from Di Lella et al. [62]

Figure 2.6: Hot temperature calibration results for "shooting star" simulations. "with" (resp. "without") in the legends refers to the presence (resp. absence) of reinsertion moves.

from one to the other requires transitioning through a state where one cation is in a site I' and another in either III or J, which is kinetically costly.

The average repartition of cations across the different sites corresponding to these two starting positions are represented on Figure 2.6, using two different force fields. The first one, from Boulfelfel et al. [64], requires having explicit aluminiums so a specific fixed placement was chosen, while the second one, from Di Lella et al. [62], is designed for uniform T-atoms whose properties were set to correspond to the target Si/Al = 2.

In both cases, sites I' remain strongly populated for starting configuration I' up to 1000 K without reinsertion moves, much less so with reinsertions, hence the inclusion of these moves significantly increases the convergence of the MC simulations.

The temperature itself plays an critical role: at low temperatures, the repartition remains close to the starting position, while at high temperatures the repartition becomes independent of the starting position. This effect is more pronounced on the simulations starting from configuration I', in particular when using the force field by Di Lella et al. [62]. The hot temperature of 2000 K was thus chosen because it is the first temperature that shows an

ABW	3	CFI	8	LOS	4	SAS	3
AEL	7	CHA	4	LTA	2	SAT	6
AFG	6	DOH	6	LTJ	2	SFE	12
AFO	6	DON	12	LTL	6	SFG	22
AHT	4	EMT	14	MAR	11	SFN	11
ANA	2	EWO	7	MEL	19	SGT	5
ANO	12	FAR	8	MSO	5	SOD	1
AST	1	FAU	5	MTW	8	SSF	11
ASV	3	FER	8	NPT	2	TOL	17
ATN	3	FRA	13	NSI	7	TON	6
ATS	7	GIU	10	OBW	6	TSC	8
ATV	5	IFO	8	PTT	5	UFI	8
AWW	5	IHW	11	RFE	10	UOZ	4
BOZ	15	IRN	14	RTE	5	VET	6
CAN	2	JRY	6	RWY	3	VFI	4
CAS	5	LIO	11	SAF	7		

Table 2.3: Number of cationic sites identified per investigated zeolite topology

identical repartition independent of the starting configuration, while still avoiding having cations out of site (the right-most "unknown" bucket in the figure). At that temperature, the influence of the reinsertion moves as negligible, but we kept them for the expected improved convergence overall.

2.3.2 Other zeolites

The methodology developed before is not specific to the FAU zeolite, and can be used on any other structure. At the time of writing, 63 zeolite topologies were automatically investigated by running "shooting star" simulations on them at different Si/Al ratios, in order to identify their cationic sites. Table 2.3 lists these topologies, along the number of symmetry-wise unique sites identified for each: for example, zeolite FAU has 5 unique sites, which are I, I', II, III and J.

These numbers exhibit quite large disparities, between only 1 for AST and SOD to 22 different unique sites for SFG. They correlate quite well with the asymmetric volume of the unit cells, that is the volume of the unit cell divided by the number of symmetries of the topology, as illustrated on Figure 2.7. This correlation is mathematically founded by the nature of the cationic sites the algorithm finds, which must obey the symmetries of the framework.

Nonetheless, like for the case of FAU, the repartition of the cations largely differ depending on the site, making the less populated ones less relevant for analysis. For example, the ANO topology represented on Figure 2.8 has 12 identified sites, of which 3 are never populated by more than one cation across all Si/Al ratios.

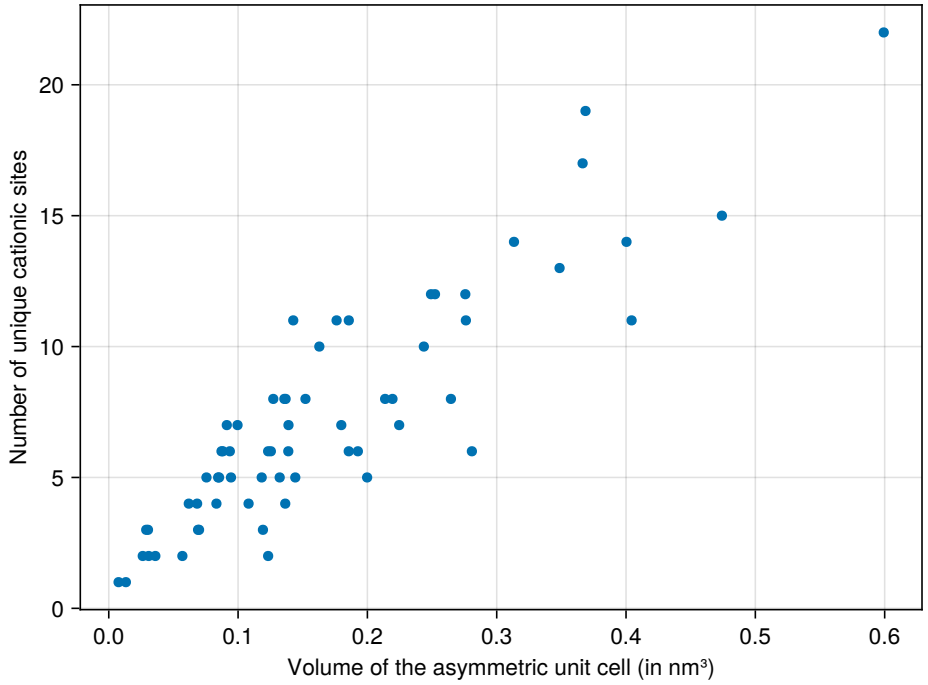


Figure 2.7: Correlation between the asymmetric volume of the unit cell and the number of unique site per zeolite

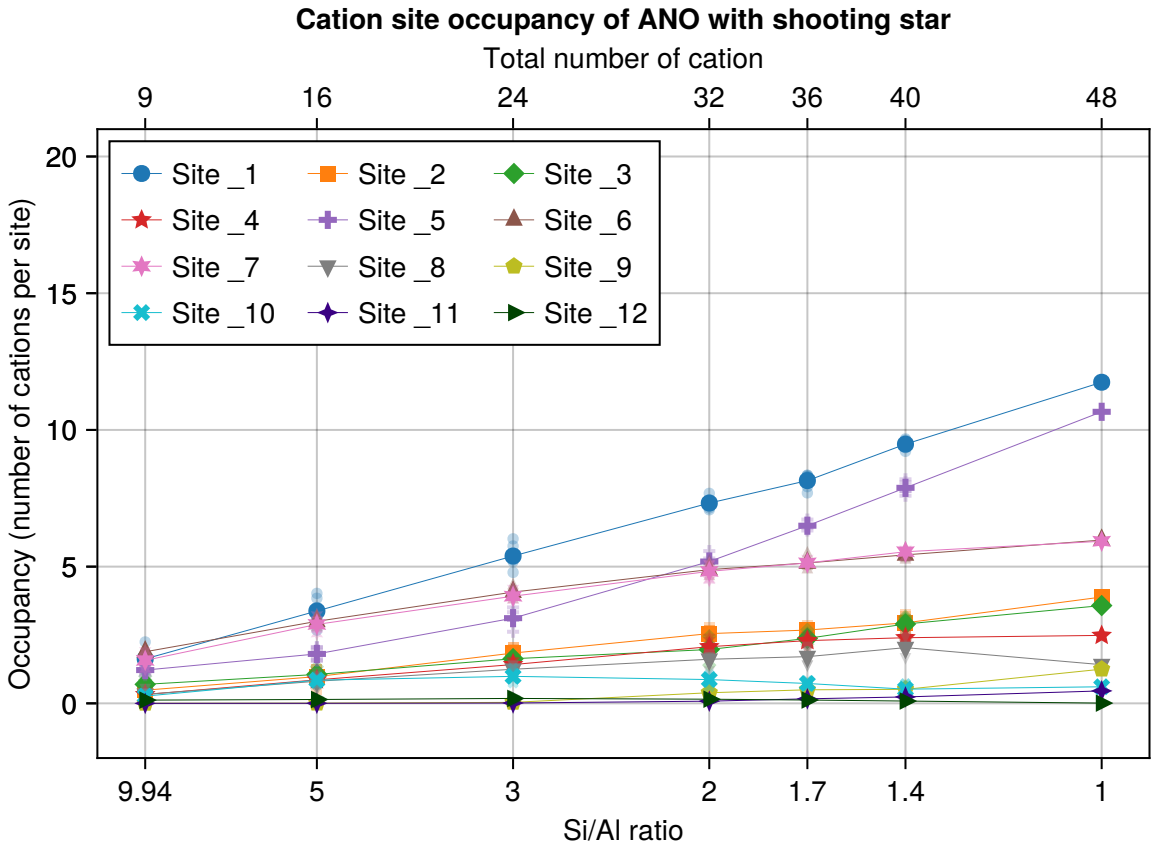


Figure 2.8: Population of each site of ANO

2.4 Energy computation

All of the previously exposed methods rely on Monte-Carlo steps which may, or may not, be accepted, depending on the difference of energy between the two configurations. By far, the greatest computational cost of any of these methods lies in these energy computations; it is therefore crucial to be able to compute it efficiently.

The RASPA2 software

RASPA2 is an open-source software developed by [REF] for classical simulations of molecular systems. Although general purpose, it is mainly used with nanoporous materials and in particular for the study of adsorption and diffusion within. RASPA2 implements Monte-Carlo schemes with the Metropolis-Hastings algorithm to perform canonical simulations, but also grand canonical for adsorption (see Chapter 3), as well as parallel tempering. Furthermore, it includes Gibbs ensemble simulation, flexible guest species and frameworks, and many other functionalities which were not needed here.

RASPA2 is written in C, a low-level programming language which allows reaching optimal performance, but is tedious to program in. It cannot be used on GPU and is single-threaded. In order to develop new simulation strategies in an easier setting, I re-implemented the algorithms used for both canonical and grand-canonical simulation of rigid species in rigid frameworks using the Julia programming language (see **??**).

The energy computation algorithms detailed herein are those which were implemented in this new code, but most of them follow that implemented in RASPA2. Moreover, RASPA2 was used to validate the new code, by checking that the simulations yielded the correct physical values (energies, adsorption capacities, and such). By contrast, the "shooting star" methodology presented before does not exist in RASPA2.

2.4.1 Force fields

The energy of a molecular structure results from the interaction of its constituent atoms, which are intrinsically quantum. At the molecular level, a popular method used is the quantum Density Functional Theory (DFT), which includes a variety of particular methods, each presenting its own compromise between precision and performance. Some alternatives include methods derived from molecular orbital theory. The number of energy computations for a single molecular simulation can be of the order of 10 000 however, which would be much too costly with such methods.

What is commonly used instead are classical force fields, which approximate quantum interactions as a sum of classical interactions between each pair of atoms. These pairwise interactions are defined by one formula per pair of nature of atoms; in the case of non-bonded atoms, each formula is simply a function of the distance between the two atoms. Zeolites are mostly rigid materials, which means that their deformation due to their vibration modes, or subsequent to the organization of cations or adsorbed gas inside, is negligible for the purpose of the molecular simulation. As a consequence, the interactions between framework atoms can be skipped, the only ones to consider for cation placement are those between the cation and the different atoms of the zeolites, which are not bonded.

The pairwise interaction formula is usually divided between the energy contribution of the charges of the atoms, and a short-range van der Waals term. Designing a force fields thus customarily consists in choosing the atomic charges and the formula for the van der Waals term. Several options have been used in the literature for the latter: a very common option is the Lennard-Jones potential

$$r \mapsto 4\varepsilon \left(\left(\frac{\sigma}{r} \right)^{12} - \left(\frac{\sigma}{r} \right)^{6} \right) \tag{2.4}$$

where r is the distance between atoms. One alternative is the Buckingham potential

$$r \mapsto \mathrm{A} \exp\left(-\mathrm{B} \times r\right) - \frac{\mathrm{C}}{r^6} \tag{2.5}$$

The potential parameter ε, σ, A, B or C are part of the design of the force field.

2.4.2 Short-term interactions

Since the van der Waals term decays quickly towards zero, it is computationnally useful to set it to exactly zero beyond a cutoff distance. To do so while avoiding the introduction of unphysical forces, one possibility consists in shifting the formula by a constant, so that the function remains continuous at the cutoff. If no force is required, such as in the MC methods, it is possible to leave the potential unshifted and discontinuous at the cutoff; a tail correction is then added to the total energy to account for the missing interactions (this will be detailed in Section 3.2.3). In any case, knowing that the interaction becomes zero after the cutoff allows some computational optimizations, such as using cell lists in order to iterate over the relevant atom pairs only, instead of all atom pairs.

For simplicity of the implementation, the dimensions of the simulated unit cell must be taken so that for any atoms A and B, there is at most one periodic image of B which is lower than the cutoff distance from atom A. This translates to the constraint that the orthogonal lengths of the unit cell must be as long as at least twice the cutoff. We take this cutoff as 12 Å, as is common in the literature [REF] and is the default in RASPA2.

Given this constraint, the total energy of a configuration due to the van der Waals interaction can be computed by summing the contribution of each the pair of atoms which are closer than the cutoff distance. This can done naively with a double loop over all atoms, computing the distance of each pair and rejecting that above the threshold. Here, the distance considered is the periodic distance, *i.e.* the smallest distance between periodic images of the two atoms, respective to the periodicity of the crystal. Computing it is up to 6 times more costly than without periodicity, and can become a computational bottleneck for simple enough force fields.

To avoid unecessary distance computations, a possibility consists in binning the space of the unit cell, and skipping the pairs of atoms that belong to bins further than the cutoff distance. This idea underlies the principle of neighbour list which can be used to obtain the list of relevant pairs of atoms efficiently. We use the CellListMap.jl [10.1016/j.cpc.2022.108452] julia package for this purpose, although we only enable it when the cell is large enough. Indeed, the naive approach can actually be more efficient in many cases where the orthogonal lengths of the cell are close to twice the cutoff distance.

2.4.3 Ewald summation

Energy decomposition

In opposition to the van der Waals term, charge interactions do not have a cutoff, because Coulomb's law does not decay fast enough towards zero. The interaction between atoms A and B, of respective charge q_A and q_B, is:

$$\mathrm{E}_{\mathrm{A,B}}^{\mathrm{Coulomb}} = \frac{q_\mathrm{A} q_\mathrm{B}}{4\pi\varepsilon_0 \times \left\|\boldsymbol{r}_\mathrm{B} - \boldsymbol{r}_\mathrm{A}\right\|} = \frac{q_n q_m}{4\pi\varepsilon_0 \times \left\|\boldsymbol{r}_m - \boldsymbol{r}_n + \boldsymbol{l}\right\|} \tag{2.6}$$

where n and m are the unique atom numbers in a reference unit cell of respectively A and B, and $\boldsymbol{l}$ is the lattice vector between the origin of the unit cell of A and that of B. Therefore the total contribution of charge interactions in a crystal with N atoms per unit cell is:

$$\mathrm{E}^{\mathrm{Coulomb}} = \frac{1}{2}\sum_{\mathrm{A}\neq\mathrm{B}} \mathrm{E}_{\mathrm{A,B}}^{\mathrm{Coulomb}} = \frac{1}{8\pi\varepsilon_0}\sum_{\boldsymbol{l}} \sum_{\substack{n=1 \\ n\neq m \text{ if } \boldsymbol{l}=0}}^{\mathrm{N}} \sum_{m=1}^{\mathrm{N}} \frac{q_n q_m}{\left\|\boldsymbol{r}_n - \boldsymbol{r}_m + \boldsymbol{l}\right\|} \tag{2.7}$$

where $\boldsymbol{l}$ iterates over the lattice vectors.

The semi-convergence of the sum over $\boldsymbol{l}$ makes it impossible to compute directly. Hence, several methods have been developped to obtain the result such as particle mesh Ewald (PME) and particle-particle-particle-mesh ($\mathrm{P}^3\mathrm{M}$). We focus on the initial Ewald summation technique underlying them, which is used in RASPA2 and which we also implemented.

Since the crystal is a periodic system, any function of the system that depends on a position in space can be decomposed into a simple sum by Fourier transform. This is in particular true of the electric potential, hence the problematic sum above could be computed in Fourier space. However, the charges are located on the atoms, and are thus discrete, which causes singularities that prevent the computation. The idea behind Ewald summation consists in fictitiously smoothing the point charges into Gaussians: these smooth charges yield a potential computable in Fourier space, while the potential resulting from the real point charge minus their Gaussian counterpart is computed in real space. Electric potential is additive, so the total electric potential $\Phi(\boldsymbol{r})$ is then obtained by adding the real space term $\Phi^{\mathrm{direct}}(\boldsymbol{r})$ and the Fourier term $\Phi^{\mathrm{reciprocal}}(\boldsymbol{r})$. Finally, the resulting energy is simply the action of the point charges on the potential:

$$\mathrm{E}^{\mathrm{Coulomb}} = \frac{1}{2}\sum_{n=1}^{\mathrm{N}} q_n \left(\Phi^{\mathrm{direct}}(\boldsymbol{r}_n) + \Phi^{\mathrm{reciprocal}}(\boldsymbol{r}_n)\right) - \mathrm{E}^{\mathrm{self}} \tag{2.8}$$

$\mathrm{E}^{\mathrm{self}}$ designates a spurious self-interaction energy term which must be substracted.

Direct term

To each atom A of the crystal is associated a Gaussian charge density

$$\boldsymbol{r} \mapsto q_\mathrm{A}\left(\frac{\alpha}{\sqrt{\pi}}\right)^3 \exp\left(-\alpha^2\left\|\boldsymbol{r}_\mathrm{A} - \boldsymbol{r}\right\|^2\right) \tag{2.9}$$

Thus, the direct term is the potential resulting from the point charge minus this Gaussian. This can be obtained by solving Poisson's law:

$$-\nabla\Phi(\boldsymbol{r}) = 4\pi\rho(\boldsymbol{r}) \tag{2.10}$$

which yields, after some simplification steps not detailed here:

$$\Phi_{\mathrm{A}}^{\mathrm{direct}}(r) = q_{\mathrm{A}} \frac{\operatorname{erfc}(\alpha r)}{r} \tag{2.11}$$

where r is the distance to atom A and erfc is the complementary error function erfc = 1 − erf, with erf : $x \mapsto \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} \mathrm{d}t$. Since erfc quickly decays towards 0, the total direct potential at a given point can be computed by only considering the contribution of the closest charges around, so

$$\Phi^{\mathrm{direct}}(\boldsymbol{r}) \approx \sum_{n=1}^{\mathrm{N}} \frac{q_n}{\left\|\boldsymbol{r} - \boldsymbol{r}_n\right\|_{\mathrm{p}}} \left(\operatorname{erfc}\left(\alpha \left\|\boldsymbol{r} - \boldsymbol{r}_n\right\|_{\mathrm{p}}\right)\right) \tag{2.12}$$

where $\left\|\boldsymbol{x}\right\|_{\mathrm{p}}$ designates the distance to the origin of the closest periodic image of $\boldsymbol{x}$.

Reciprocal term

For the reciprocal term, we start by writing the total charge density coming from the Gaussians given in Equation (2.9):

$$\rho^{\mathrm{reciprocal}}(\boldsymbol{r}) = \sum_{\boldsymbol{l}} \sum_{n=1}^{\mathrm{N}} \frac{q_n}{4\pi\varepsilon_0} \left(\frac{\alpha}{\sqrt{\pi}}\right)^3 \exp\left(-\alpha^2 \left\|\boldsymbol{r}_n + \boldsymbol{l} - \boldsymbol{r}\right\|^2\right) \tag{2.13}$$

Its Fourier decomposition over the periodic unit cell of volume V is:

$$\begin{aligned}
\hat{\rho}^{\mathrm{reciprocal}}(\boldsymbol{k}) &= \frac{1}{\mathrm{V}} \int_{\mathrm{V}} e^{-i\boldsymbol{k}\cdot\boldsymbol{r}} \rho^{\mathrm{reciprocal}}(\boldsymbol{r}) \mathrm{d}\boldsymbol{r} \\
&= \frac{1}{\mathrm{V}} \int_{\Omega} e^{-i\boldsymbol{k}\cdot\boldsymbol{r}} \sum_{n=1}^{\mathrm{N}} \frac{q_n}{4\pi\varepsilon_0} \left(\frac{\alpha}{\sqrt{\pi}}\right)^3 \exp\left(-\alpha^2 \left\|\boldsymbol{r}_n + \boldsymbol{r}\right\|^2\right) \mathrm{d}\boldsymbol{r} \\
&= \frac{2\pi\alpha^3}{\mathrm{V}\pi^{3/2}} \sum_{n=1}^{\mathrm{N}} \frac{q_n}{4\pi\varepsilon_0} e^{-i\boldsymbol{k}\cdot\boldsymbol{r}_n} \int_0^{\infty} \left\|\boldsymbol{r}'\right\|^2 \exp\left(-\alpha^2 \left\|\boldsymbol{r}'\right\|^2\right) \int_0^{\pi} e^{-i\boldsymbol{k}\cdot\boldsymbol{r}'} \sin(\theta) \mathrm{d}\theta \mathrm{d}\boldsymbol{r}' \\
&= \frac{1}{4\pi\varepsilon_0 \mathrm{V}} \exp\left(\frac{-k^2}{4\alpha^2}\right) \sum_{n=1}^{\mathrm{N}} q_n e^{-i\boldsymbol{k}\cdot\boldsymbol{r}_n}
\end{aligned} \tag{2.14}$$

after several simplification steps not detailed here. Using Poisson's law (Equation (2.10)), the electric potential due to the Gaussians can then finally be expressed as:

$$\Phi^{\mathrm{reciprocal}}(\boldsymbol{r}) = \frac{1}{\varepsilon_0 \mathrm{V}} \sum_{\boldsymbol{k} \neq \boldsymbol{0}} \sum_{n=1}^{\mathrm{N}} \frac{q_n}{k^2} e^{i\boldsymbol{k}\cdot(\boldsymbol{r} - \boldsymbol{r}_n)} \exp\left(\frac{-k^2}{4\alpha^2}\right) \tag{2.15}$$

The term $\kappa_{\boldsymbol{k}} = \frac{1}{k^2} \exp\left(-k^2/(4\alpha^2)\right)$ decays fast, so the sum over $\boldsymbol{k} \neq \boldsymbol{0}$ is actually computed as a sum over a finite number of values of $\boldsymbol{k}$ close to $\boldsymbol{0}$. These values are chosen so that the remaining sum of $\kappa_{\boldsymbol{k}}$ for any $\boldsymbol{k}$ not retained is below the target precision of the computation.

Self-interaction term

The last remaining term is the self-interaction, which must be removed. Its existence comes from the way the reciprocal term is computed: $\Phi^{\mathrm{reciprocal}}(\boldsymbol{r})$ includes the contribution of all charges on point $\boldsymbol{r}$, but there should not be any contribution of atom A at position $\boldsymbol{r}_{\mathrm{A}}$.

Moreover, in the case of rigid polyatomic molecules, like the small gases we will discuss in Chapter 3, there should not be any contribution of atom A at the positions $\boldsymbol{r}_\mathrm{B}$ of other atoms B part of the same molecule.

Each atom $1 \leq n \leq \mathrm{N}$ thus contributes a spurious potential term:

$$\Phi_n^{\text{self}} = \frac{2\alpha q_n}{\sqrt{\pi}} + \frac{1}{2} \sum_{\substack{m \\ m \smile n,\ m \neq n}} \frac{q_m \operatorname{erf}\left(\left\|\boldsymbol{r}_n - \boldsymbol{r}_m\right\|_\mathrm{p}\right)}{4\pi\varepsilon_0 \left\|\boldsymbol{r}_n - \boldsymbol{r}_m\right\|_\mathrm{p}} \tag{2.16}$$

where $m \smile n$ is true when both atoms m and n belong to the same molecule.

Total charge energy

Combining the last three terms, we obtain the overall Coulomb energy:

$$\begin{aligned} \mathrm{E}^{\text{Coulomb}} = &\sum_{\boldsymbol{l}} \sum_{n=1}^{\mathrm{N}} \sum_{\substack{m=1 \\ m \not\smile n}}^{n} \frac{q_n q_m}{8\pi\varepsilon_0} \frac{\operatorname{erfc}\left(\alpha \left\|\boldsymbol{r}_n - \boldsymbol{r}_m + \boldsymbol{l}\right\|\right)}{\left\|\boldsymbol{r}_n - \boldsymbol{r}_m + \boldsymbol{l}\right\|} \\ &+ \frac{1}{2\varepsilon_0 \mathrm{V}} \sum_{\boldsymbol{k} \neq \boldsymbol{0}} \sum_{n=1}^{\mathrm{N}} \sum_{m=1}^{\mathrm{N}} \frac{q_n q_m}{k^2} e^{i\boldsymbol{k}\cdot(\boldsymbol{r}_m - \boldsymbol{r}_n)} \exp\left(\frac{-k^2}{4\alpha^2}\right) \\ &- \cancel{\sum_{n=1}^{\mathrm{N}} \frac{2\alpha q_n}{\sqrt{\pi}}} - \frac{1}{2} \sum_{n=1}^{\mathrm{N}} \sum_{\substack{m=1 \\ m \smile n,\ m \neq n}}^{\mathrm{N}} \frac{q_n q_m}{4\pi\varepsilon_0} \frac{\operatorname{erf}\left(\left\|\boldsymbol{r}_n - \boldsymbol{r}_m\right\|_\mathrm{p}\right)}{\left\|\boldsymbol{r}_n - \boldsymbol{r}_m\right\|_\mathrm{p}} \end{aligned} \tag{2.17}$$

The term summing the $2\alpha q_n/\sqrt{\pi}$ cancels out because the overall system needs to be electrically neutral to carry a physical meaning, so the sum of all q_n is 0. This invariant is checked in the code to avoid any error.

2.4.4 Precomputation

Different algorithms that solve a particular problem generally differ in the specific space-time tradeoff they choose. Obtaining the energy of a configuration presents the same possibility: while entirely computing the energy from scratch requires a set amount of computation, it is possible to make subsequent computations faster at the cost of a greater memory footprint of the algorithm, by storing some intermediate results which can be reused afterwards.

Energy grid

In the case of simulations involving a rigid framework, the most obvious precomputation that can be done is that of the short-term interactions of the guest species with the framework atoms. Indeed, such interactions only depend on the nature and the position of the guest atom, so it is possible to store them on an energy grid.

More precisely, fore each possible guest atom, the unit cell of the framework is divided into a regular grid, with a set spacing between consecutive points. We use 0.15 Å, which is fine enough to be accurate but not too fine to avoid using too much memory. For each grid point, the sum of the interactions of all framework atoms with a guest atom in that position is stored at that point. The grid is then written to disk.

During the simulation, in order to compute the short-term interactions between the framework atoms and a guest atom at a given position, the value is interpolated from the surrounding grid

points. Using a simple linear interpolation of the values does not yield very satisfactory results however, since derivative discontinuity would push that atoms towards the grid points, which would result in simulation artifacts. To prevent this, RASPA2 stores not only the value of the energy, but also the three first derivatives, three values of the second derivatives and one of the third, following a scheme by [REF]. Moveover, the grid is forced to be orthorombic: this means that it may not be aligned with the unit cell of the framework, but it does not matter as long as each point in the unit cell can be surrounded by recorded grid points. I reimplemented the same grid algorithm, which ensures a smooth interpolation everywhere in space.

Ewald precomputation

Short-term interactions that can be stored in grids include the short-term direct part of the Ewald summation. The reciprocal term cannot however be simply stored as such.

Yet, it is still possible to precompute a large part of the reciprocal interaction which is due to the framework. To take the potential given by Equation (2.15) back to the energy, it is useful to introduce the sums:

$$\mathrm{S}_{\boldsymbol{k}} = \sum_{n=1}^{\mathrm{N}} q_n e^{-i\boldsymbol{k}\cdot\boldsymbol{r}_n} \tag{2.18}$$

for each $\boldsymbol{k}$. With this notation, the contribution of the reciprocal term to the energy is

$$\begin{aligned}
\mathrm{E}^{\mathrm{reciprocal}} &= \frac{1}{2}\sum_{m=1}^{\mathrm{N}} q_m \Phi^{\mathrm{reciprocal}}(\boldsymbol{r}_m) \\
&= \frac{1}{2\varepsilon_0 \mathrm{V}} \sum_{m=1}^{\mathrm{N}} q_m \sum_{\boldsymbol{k}} \kappa_{\boldsymbol{k}} e^{i\boldsymbol{k}\cdot\boldsymbol{r}_m} \mathrm{S}_{\boldsymbol{k}} \qquad \text{then using } \overline{\mathrm{S}_{\boldsymbol{k}}} = \sum_{m=1}^{\mathrm{N}} q_m e^{i\boldsymbol{k}\cdot\boldsymbol{r}_m} \\
&= \frac{1}{2\varepsilon_0 \mathrm{V}} \sum_{\boldsymbol{k}} \kappa_{\boldsymbol{k}} \overline{\mathrm{S}_{\boldsymbol{k}}} \times \mathrm{S}_{\boldsymbol{k}} = \frac{1}{\varepsilon_0 \mathrm{V}} \sum_{\boldsymbol{k}} \kappa_{\boldsymbol{k}} \left\|\mathrm{S}_{\boldsymbol{k}}\right\|^2
\end{aligned} \tag{2.19}$$

Hence, to avoid redundant computations, the list of relevant $\boldsymbol{k}$, associated to their factor $\kappa_{\boldsymbol{k}} = \exp\left(-k^2/\left(4\alpha^2\right)\right)/k^2$, is computed once and for all. For each atom n of the framework, the factors $q_n \exp\left(i\boldsymbol{k}\cdot\boldsymbol{r}_n\right)$ are also computed once. Distinguishing between the atoms of the framework and the guests allow separating $\mathrm{S}_{\boldsymbol{k}}$ into $\mathrm{S}_{\boldsymbol{k}}^{\mathrm{framework}} + \mathrm{S}_{\boldsymbol{k}}^{\mathrm{guest}}$, so that finally, the energy can be computed as

$$\mathrm{E}^{\mathrm{reciprocal}} = \underbrace{\frac{1}{2\varepsilon_0 \mathrm{V}} \sum_{\boldsymbol{k}} \kappa_{\boldsymbol{k}} \left\|\mathrm{S}_{\boldsymbol{k}}^{\mathrm{framework}}\right\|^2 + \frac{1}{2\varepsilon_0 \mathrm{V}} \sum_{\boldsymbol{k}} \kappa_{\boldsymbol{k}} \left(\overline{\mathrm{S}_{\boldsymbol{k}}^{\mathrm{framework}}} + \overline{\mathrm{S}_{\boldsymbol{k}}^{\mathrm{guest}}}\right) \times \mathrm{S}_{\boldsymbol{k}}^{\mathrm{guest}}}_{\text{precomputed part}} \tag{2.20}$$

Finally, the entire self-interaction term can be computed once for all, since it does not depend on the configuration. This is only the case here because the simulated moleculses are rigid bodies; for deformable structures, the self-interaction term depends on the actual conformation of the guests, which may depend on the state.

Optimizations for computing the difference of energies

A simulation that follows the Metropolis-Hastings algorithm walks across configurations, each pair of contiguous states being separated by a single MC move. In this context, computing the energy of each trial configuration to compare it to the previous one is wasteful: the only part

of the energy impacted by the MC move is that which stems from the moved species. Instead of computing the energy of the trial configuration, it is thus better to directly compute the energy difference due to the move.

For the short-term interactions, this is straightforward: the energy difference is simply the sum of the interactions of the species at the new position minus that at its previous position. If there are N species, moving one thus costs O(N) computations for these terms, instead of $O(N^2)$ required to compute this energy the first time. The direct part of the Ewald summation follows the same principle.

For the reciprocal part, the difference in energy is derived from Equation (2.20) by removing the first constant term and computing:

$$\Delta E^{\text{reciprocal}} = \frac{1}{2\varepsilon_0 V} \sum_{\boldsymbol{k}} \kappa_{\boldsymbol{k}} \left(\overline{S_{\boldsymbol{k}}^{\text{framework}}} \times \Delta S_{\boldsymbol{k}}^{\text{guest}} + \Delta \left\| S_{\boldsymbol{k}}^{\text{guest}} \right\|^2 \right) \tag{2.21}$$

Since both values of $S_{\boldsymbol{k}}^{\text{guest}}$ need to be computed before and after the move to take the difference of their norm, it is actually equally efficient to directly compute both values of $E^{\text{reciprocal}}$ before and after the move, directly following Equation (2.20), and taking the difference, instead of the previous formulation.

In order to allow computing this efficiently, each term $q_n e^{-i\boldsymbol{k}\cdot\boldsymbol{r}_n}$ is thus stored in memory (instead of recomputing it each time). The new values $q_n e^{-i\boldsymbol{k}\cdot\boldsymbol{r}'_n}$ computed for the trial move are also temporarily kept in memory, so they can be used to replace the previous ones if the trial move is accepted. Finally, the value of each $S_{\boldsymbol{k}}^{\text{guest}}$ is also kept in memory, so that it can be updated by reusing the previous computation.

RASPA2 does not implement this optimization, and simply recomputes the $e^{-i\boldsymbol{k}\cdot\boldsymbol{r}_n}$ terms each time, which is slower but requires less memory.

Blocking pockets

Some MC trial moves attempt to displace an atom to a location that would lead to unphysically high energies – typically, too close to a framework atom. Such positions can be determined once and for all, and the relevant MC moves automatically rejected without the need for any energy computation.

The simplest strategy to do so consists in making a boolean grid, aligned with the energy grids used for short-term interactions, which stores a 1 when the voxel is blocked or 0 if it is allowed. The first step of any MC trial move then consists in checking whether each atom is in an allowed voxel, or rejecting the move otherwise. In practise, the simplest implementation consists in making one such boolean grid per atom kind (since the value of the framework-atom interaction energy depends on the atom), and for each actual atom position, checking the closest grid point of its kind.

In addition to its use as computational optimization, such boolean grids can also be used to mark some parts of the framework as inaccessible, irrespective of the corresponding energy. This is useful when small window size prevents a species – for instance a gas molecule – from entering a pore – for instance the sodalite cages in LTA zeolites [10.1021/ja953856q]. Indeed, without explicitly blocking the pocket, an MC move could make a species appear in the pore,

whereas it should be kinetically impossible to enter. This strategy will thus be used when studying the adsorption of gases in zeolites (see Chapter 3), but for the small cations under study here, there is no need to additionally block any part of the frameworks.

RASPA2 only implements blocking for the latter case – *i.e.* explicit pocket removal. It does not use a boolean grid: instead, the blocked space is described as a combination of blocking spheres. For each trial move, the distance between the atoms of the moved guest and the center of each sphere is computed: if one is below its radius, the move is rejected. This approach requires less precomputation and thus less memory than storing the boolean grids, but its runtime cost is much greater, and increases with the number of blocking spheres, preventing it from being used as an optimization like before. It is thus less efficient than the approach I implemented.

These boolean grids are also useful to improve the acceptance rate of reinsertions. When a reinsertion move is attempted, a new position for the species is chosen at random in the unit cell, but it is then immediately checked against the blocking pocket grids. If the attempted position is forbidden, a new random position is chosen, and so iteratively until finding a position compatible with the boolean grid constraints. This trial position is then used as any other trial moves, and can be accepted or rejected based on the energy criterion. Crucially, biasing the choice of the reinsertion trial move in this fashion improves its acceptance rate while still maintaining the detailed balance of the simulation, because a reinserted species was necessarily located in a position that was not blocked by the boolean grids either. Note that this bias cannot be used for local translations or rotation however: if a trial translation moves a species in a blocked pocket, the move must be rejected.

Site-hopping precomputations

The main computational advantage of site hopping simulations is that the number of sites is fixed in advance, which allows precomputing all framework-site and site-site interactions.

To do so, the value of the framework-site interaction energy is computed and stored for each site by computing the energy associated with putting only one cation in the framework at the position of the site. However, this energy is actually ill-defined, since placing only one cation in the framework means that the unit cell has a net charge, and thus the (infinite periodic) material has an infinite charge. Moreover, even disregarding the physical absurdity of this scenario, a net charge would lead to a non-zero $\sum_{n=1}^{N} 2\alpha q_n/\sqrt{\pi}$ term, even though it was removed in Equation (2.17). Still, the computations can be carried to yield a numerical value, which is thus stored in a list framework$[i]$ where i designates the index of a particular site s_i. The reciprocal ewald contribution of this energy is also temporarily stored in a separate list as r_i.

Likewise, for each pair (s_i, s_j) of sites, the energy of the framework containing only two cations, one in s_i and one in s_j, is computed as $e_{i,j}$, although it also does not carry physical sense either. What is stored in a matrix is interaction$[i, j] = e_{i,j} - r_i - r_j$. This is only possible if the number of sites is not too high, since the number of interactions to store grows quadratically with the number of sites and would become prohibitively high otherwise.

Finally, the tail correction tc_n associated with the framework filled with n cations is computed once, where n is constrained by the number of aluminiums in the structure.

Using these unphysical values, the actual physical energy of a real configuration with n cations on sites $s_{k_1}, s_{k_2}, \ldots, s_{k_n}$ can then be computed as

$$\mathrm{E}(s_{k_1}, s_{k_2}, \ldots, s_{k_n}) = \mathrm{tc}_n + \sum_{i=1}^{n} \mathrm{framework}[k_i] + \sum_{1 \leq i < j \leq n} \mathrm{interaction}[k_i, k_j] \tag{2.22}$$

and the energy associated with a trial move of one cation can thus be computed with only n array reads, which is optimally efficient.

As previously emphasized, the components framework$[i]$ and interaction$[i, j]$ should not be considered individually to assess the stability of a particular site. Indeed, framework$[i]$ < framework$[j]$ does not necessarily mean that site s_i is more populated than s_j, because the site-site interaction energies could make the total energy E systematically higher when s_i is populated rather than s_j, which is the true measure of a site stability.

2.5 Possible alternatives

Molecular dynamics (MD) could also be used to place cations in zeolites. Generally speaking, the principle of MD consists in running a simulation where the particles follow Newton's laws of motion, with forces based on force fields like for MC schemes. The simulation is divided in time steps, between which the forces are computed on each particle to derive their position at the next time step. Hence, the trajectories of the particles in MD follow a physical path.

Temperature can be included in different ways in MD, all of which use an element of randomness that eventually impacts the velocities of particles, in order to ensure that the average values extracted from the simulation correspond to those which would be obtained at that temperature. This means that the precedent discussion on the inaccessibility of high-energy regions to the system at room temperature still applies in MD.A simple MD simulation is thus doomed to only explore local energy minima, failing to reach ergodicity like with the simple canonical MC scheme.

To circumvent this issue, it is necessary to bias the simulation in order to force the system to explore regions of high energy. One such strategy consists in doing metadynamics [REF], which is a variation of MD in which the potential energy landscapes is modified during the simulation. The effective free energy of a point in configuration space is computed as the sum of the real free energy of that point without bias, added to the sum of some Gaussian functions centred on previously explored points. Every few steps, another Gaussian function centred on the current configuration point is added to the bias, so that the effective free energy of explored regions increases through the simulation, which allows the system to eventually escape local minima and explore other regions.

A Gaussian is a function of the form $f : \boldsymbol{x} \mapsto \mathrm{A} e^{-(\boldsymbol{x}-\boldsymbol{\mu})^\top \Sigma (\boldsymbol{x}-\boldsymbol{\mu})}$. In practise, the variable $\boldsymbol{x}$ of the Gaussians used in metadynamics are not directly the 3N coordinates of the N particles, but a small-dimensional vector whose coordinates are called "collective variables". These collective variables are themselves function of the 3N initial coordinates. The parameter A and Σ of the Gaussians being usually fixed, each Gaussian is stored by its centre $\boldsymbol{\mu}$, whose coordinates are thus the collective variables of the current configuration point. Therefore, compared to regular MD, running a metadynamics simulation incurs the additional cost of computing the collective

variables $\boldsymbol{x}$ of each visited configuration point, storing the Gaussians' centres, and computing their value at each step.

For cation placement the main issue is to find a small set of relevant collective variables, since the cost of metadynamics grows with their number. We did not explore this venue because we expected it to be similar in difficulty to the MC alternatives, with no clear reason why it should be more efficient. MD in general is also known to converge more slowly than MC, because large configuration change made possible by MC steps are prohibited by the physics-driven approach of MD, and the same argument could be made of metadynamics compared to other accelerated MC schemes. Exploring metadynamics or its variants could nonetheless be of interest in the search for more alternative, and potentially more efficient, algorithms.

3

MOLECULAR SIMULATION OF ADSORPTION

Adsorption is the physical phenomenon by which a molecule, the adsorbate, attaches to a surface, the adsorbent. One particular application of adsorption in the industry is for gas separation, by using an adsorbent that specifically binds one constituent of the target gas mixture more than the other.

In this chapter, we detail the nature of this phenomenon in the case of small gases in zeolites. We then explain the usual techniques which are used for its numerical simulation, as well as a possible alternative.

3.1 Molecular description

3.1.1 Physisorption and chemisorption

The atomic nature of adsorption depends on the kind of chemical interaction the supports it. On the one hand, if the adsorbate forms a strong (*i.e.* covalent, sometimes ionic) bond with the adsorbent, then the adsorption is called a chemisorption, because it involves a chemical reaction. In that case, the surface is modified. On the other hand, if the adsorbate and adsorbent are only bond by a weak interaction, the adsorption is called a physisorption and the surface is left intact.

Chemisorption involves a stronger bond that physisorption, and is thus less reversible. Since it changes the surface of the adsorbent, it cannot be used industrially to separate large amounts of gas, as the adsorbent would need to be provided in equivalently large amount. It can however be used in the industry as an intermediate step of a catalytic cycle that ends up regenerating the surface. In our case however, we will implicitly refer to physisorption when mentioning adsorption.

3.1.2 Adsorption sites in crystalline materials

An adsorption site designates a particular space where the adsorbate is likely to remain trapped on the surface of the adsorbent. In other words, the interaction between the adsorbate and the adsorbent is maximally attractive when the adsorbate is located in one of the adsorption sites.

The position of the adsorption site depends on the adsorbent, but also the adsorbate. At very low density for example, a small polar molecule like H_2O will preferentially adsorb in locations where it can orient itself to maximize the stabilizing Coulomb interaction of the framework. A large apolar molecule like a xylene on the other hand, will adsorb in locations that maximize its Van der Waals interactions with the adsorbent. Such sites are not necessarily related.

Moreover, the location of the adsorption sites becomes less relevant when the density of the adsorbate becomes high. Indeed, the adsorbate-adsorbate interactions can play a key role in the global adsorption capacity of a material, and the stronger these interactions, the more the internal structure of the adsorbate will reorganize to accommodate these interactions, which may displace some molecules from the ideal adsorption site.

In the case of crystalline materials, adsorption sites can be identified through XRD spectroscopy. For example, Marqueño et al. [65] identify the preferred adsorption sites for CO_2 in silica MFI zeolite (silicalite) in a context of very high mechanical pressure, compatible with DFT

computations. Yet, the obtained positions for CO_2 molecules is not guaranteed to carry to another gas pressure than the one used in the study.

Overall, the situation is quite different from the cationic sites, discussed in Section 2.2.1, mostly because the cations cannot reach a "high-density" regime, given that their number is constrained by the number of aluminium atoms. In turn, this makes the use of site-hopping or other site-derived algorithms less relevant in the context of adsorption simulation.

3.1.3 Small gases in zeolites

Small gases such as CO_2, N_2, O_2, CH_4 or Ar, among others, are typical adsorbates that need to be separated for industrial purposes. In general, molecular sieves like zeolites can be used for this purpose

For molecules with significant size disparities, a simple approach consists in using a framework whose pore size allows one of the molecule to enter and adsorb, but not the other. Given the proximate sizes of the previous species however, this method cannot be used to separate these gases. Instead, what plays a key role is the adsorption capacity of the material with respect to each of these gases: if there is a material that adsorbs CO_2 easily but not N_2 for instance, then it can be used to separate the two.

3.2 Grand Canonical Monte-Carlo

3.2.1 Principle

The adsorption capacity of a gas in a material at a fixed temperature and pressure is the number of molecules of gas that are retained by the material when in contact with a gas in these conditions, at equilibrium. Since it is a macroscopic observable $\mathcal{O}$, which can be experimentally measured, it can also be expressed as the statistical average of its microscopic counterpart o in a well-chosen statistical ensemble. For the current case, the adsorption capacity results from a chemical equilibrium between the gas out of the material, at its given temperature T and pressure P, and the gas in the pores of the material, at the same temperature T. The pressure of the gas in the material is an ill-defined concept, because it depends on the density of the gas, which itself is not uniform at any mesoscopic scale in a porous material. Instead, the chemical equilibrium translates to the equality of the chemical potentials μ of the gas in both conditions, hence μ is the relevant statistical variable to describe the system in the material, which accounts for the fixed gas pressure P out of the material. Finally, we will focus on rigid crystalline materials, hence the volume V of a unit cell can be used as an extensive variable to limit the size of the system.

The statistical ensemble resulting from the constraints of fixing the chemical potential μ, the temperature T, and the volume V is called the grand canonical ensemble. Similar to Equation (2.1) used for the canonical ensemble (where the number of particles N is fixed instead of their chemical potential μ), $\mathcal{O}$ can be expressed using an integral over the microstates $(\mathrm{N}, \boldsymbol{p}_\mathrm{N})$:

$$\mathcal{O} = \langle o \rangle = \frac{1}{\Xi} \sum_{\mathrm{N}} \int \mathrm{d}\boldsymbol{p}_\mathrm{N}\, o\left(\boldsymbol{p}_\mathrm{N}\right) \exp\left(-\beta\left(\mathrm{U}\left(\boldsymbol{p}\right) - \mu\mathrm{N}\right)\right) \tag{3.1}$$

where N is the number of gas particles, $\boldsymbol{p}_\mathrm{N}$ is their configuration, and Ξ is the grand canonical partition function.

Likewise, this integral can be numerically evaluated by relying on a Monte-Carlo Markov Chain to provide a sequence of configurations $(\mathrm{N}, \boldsymbol{p}_\mathrm{N})$ such that the statistical average $\langle o \rangle$ can be retrieved by computing the arithmetic average of $o\left(\boldsymbol{p}_\mathrm{N}\right)$ over these configurations: this particular kind of simulation is naturally called a Grand Canonical Monte-Carlo (GCMC) simulation. In the specific case of the adsorption capacity, the microscopic function is simply $o\left(\boldsymbol{p}_\mathrm{N}\right) = \mathrm{N}$, thus it is computed by running a GCMC simulation and simply averaging the number of particles in the system across the simulation.

3.2.2 Implementation

In practice, GCMC is implemented like a canonical MC simulation with one additional element: a swap MC move that can either insert or delete a particle. The insertion move simply consists in adding one additional gas particle to the system, while the deletion move removes one. If there are no particle in the systems anymore, the deletion move systematically fails; yet, it is crucial that the probability of trying an insertion or a deletion remains equal independently of the number of particles in the system, to ensure micro-reversibility. For this reason, only the probability of trying a swap move is specified as a simulation parameter, while the probability of trying either an insertion or a deletion cannot be individually chosen and are necessarily equal to half the probability of a swap move.

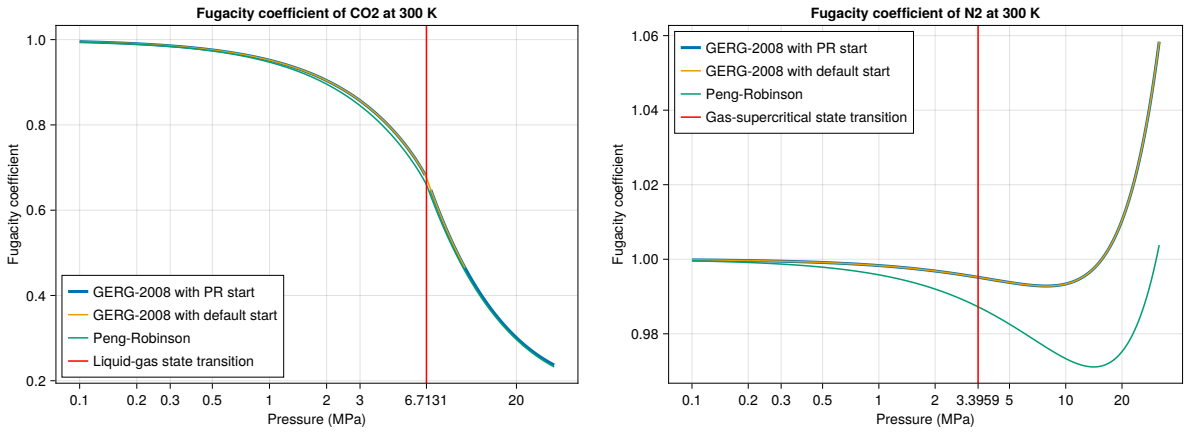
The acceptance probability of a swap move is not decided purely on the basis of the difference of energy in the system, as is the case in Equation (2.2) for the canonical ensemble, because it also depends on the chemical potential μ of the system. With N, the number of particles in the system before the trial swap move, the acceptance probabilities for insertion and deletion are:

$$\begin{aligned} \mathrm{P}_\mathrm{accept}(\Delta\mathrm{E}, \mathrm{N} \to \mathrm{N}+1) &= \min\left(1,\ \frac{\beta\varphi\mathrm{PV}}{\mathrm{N}+1}\exp\left(-\beta\Delta\mathrm{E}\right)\right) \\ \mathrm{P}_\mathrm{accept}(\Delta\mathrm{E}, \mathrm{N} \to \mathrm{N}-1) &= \min\left(1,\ \frac{\mathrm{N}}{\beta\varphi\mathrm{PV}}\exp\left(-\beta\Delta\mathrm{E}\right)\right) \end{aligned} \tag{3.2}$$

The dependency to the chemical potential μ is conveyed by the pressure P of the gas exterior to the zeolite. This value is corrected by the fugacity coefficient φ, which accounts for the difference between the real gas and the ideal gas model.

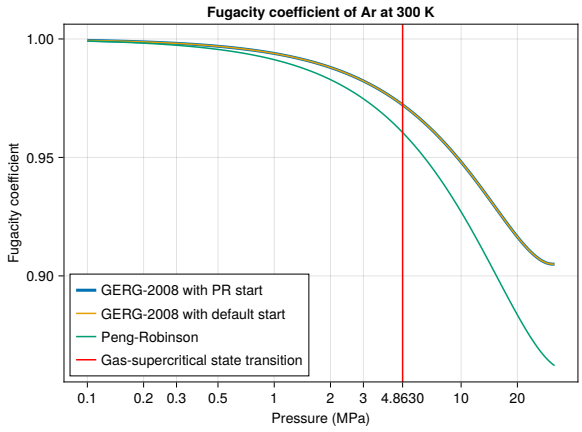
While the pressure P and volume V of the systems are simulation inputs, the fugacity coefficient φ of the gas needs to be computed prior to the simulation. To do so, RASPA uses the Peng-Robinson equation of state,[66] which is adequate for many common fluids. Since our range of gas applications is more restricted than the general-purpose RASPA software, we use the GERG-2008[67] equation of state, which is the ISO standard for natural gases due to its accuracy compared to experimental measurements, with an extended validity range up to 700 K and 70 MPa, in the scope of our study.

In terms of programming, the fugacity coefficient can directly be obtained from the `Clapeyron.jl`[68] Julia package, which provides a convenient unified access to many useful thermodynamic tools. Due to the way it is implemented, Clapeyron.jl used to fail and return `NaN` instead of the valid fugacity coefficient for CO_2 at high pressures: to solve this issue, we do an initial volume

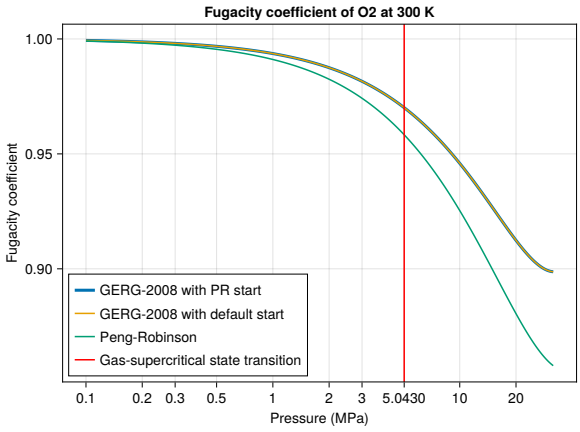


(a) CO_2 at 300 K

(b) N_2 at 300 K



(c) Ar at 300 K



(d) O_2 at 300 K

Figure 3.1: Fugacity coefficients of small gases computed with different `Clapeyron.jl` *models*

computation for the gas in the target conditions using the Peng-Robinson model, and we use the result as the starting point to compute the coefficient with the GERG-2008 model. Figure 3.1a illustrates the differences between these approaches: the latter allows to compute fugacities at any pressure except those very close to the critical point (304.13 K and 7.3773 MPa for CO_2), while the former simply fails at high pressures (above 1 MPa). The other gases illustrated on Figure 3.1 show that the Peng-Robinson fugacity is usually close but slightly lower than that obtained with GERG-2008, which would result in an equally slight underestimation of the adsorption capacity. The difference between the two equations of state remains in the uncertainty margin of the GCMC simulation however, so it should not lead to a visible difference in the measured adsorption capacity.

3.2.3 Computational aspects

Compared to canonical MC, the efficient implementation of GCMC poses the additional challenge of changing the number of particles across the simulation. This requires almost no accommodation for the computations that operate in the real space, *i.e.* framework pre-computed grids, and pairwise interactions (including the direct part of Ewald summation). For these, the only adaptation consists in making the neighbour lists resizable, which is already implemented

in `CellListMap.jl`, and otherwise keeping the list of atoms in the system up to date, to correctly check each pair of atoms.

Ewald summation

Although the direct part of Ewald summation can be treated like the other short-range pair interactions, the reciprocal-space part of Ewald summation requires adapting the optimization described in Section 2.4.4 to account for the possible introduction or removal of a species. For such a move, both values of $S_{\boldsymbol{k}} = \sum_n q_n e^{i\boldsymbol{k}\cdot\boldsymbol{r}_n}$ before and after the swap need to be computed, exactly like in Equation (2.21). Likewise, the value before the swap is already stored in memory, but the computation of the new value is slightly different: instead of computing the difference as $\sum_n q_n \left(e^{i\boldsymbol{k}\cdot\boldsymbol{r}'_n} - e^{i\boldsymbol{k}\cdot\boldsymbol{r}_n} \right)$, the difference is now $\pm \sum_{n\ \text{swapped}} q_n e^{i\boldsymbol{k}\cdot\boldsymbol{r}_n}$. In the case of a deletion, each element of the latter is already stored in memory, whereas in the case of an insertion, this new computation is stored in a temporary place, like for any other trial move, ready to be stored permanently if the move is accepted.

The self-interaction term is no longer constant: each new molecule contributes a term equal to $\sum_m \frac{q_n q_m}{4\pi\varepsilon_0} \mathrm{erf}\left(\|\boldsymbol{r}_n - \boldsymbol{r}_m\|_\mathrm{p} \right) / \|\boldsymbol{r}_n - \boldsymbol{r}_m\|_\mathrm{p}$ where m iterates over the atoms of the molecule. This term is computed once for each kind of molecules and stored, so that the contribution of the self-interaction term can be updated with a single addition or subtraction when a swap move occurs. Of course, any molecule whose number can vary must be electrically neutral to ensure the the overall system remains electrically neutral; this is also enforced in the code.

Tail correction

Another difference is the tail correction, which is not constant anymore either. The tail correction, whose relevance for the particular context of porous materials modeling was studied by Jablonka et al. [69], is an approximation of the energy resulting from the pair interactions further than the cutoff: if the pair interaction is shifted in the force field, then this energy is null, but otherwise it needs to be taken into account.

To do so, a useful ansatz consists in assuming that the radial distribution function $g(r)$ is equal to 1 beyond the cutoff distance r_c, for all species. In turn, this allows analytically deriving the energy resulting from the average interaction of particles beyond the cutoff. For example, two particles interacting with a Lennard-Jones potential (see Equation (2.4)) yield, for a unit cell volume V and cutoff distance r_c, a tail correction of:

$$\mathrm{tc}^{\mathrm{LJ}} = \frac{8\pi}{3\mathrm{V}} \varepsilon\sigma^3 \times \left(\frac{1}{3} \left(\frac{\sigma}{r_c} \right)^9 - \left(\frac{\sigma}{r_c} \right)^3 \right) \tag{3.3}$$

Similarly, two particles interacting with a Buckingham potential (see Equation (2.5)) yield a tail correction of:

$$\mathrm{tc}^{\mathrm{Buckingham}} = \frac{2\pi}{\mathrm{V}} \left(\mathrm{A}\exp(-\mathrm{B}r_c) \times \frac{2 + \mathrm{B}r_c(2 + \mathrm{B}r_c)}{\mathrm{B}^3} - \frac{\mathrm{C}}{3r_c^3} \right) \tag{3.4}$$

All tail corrections $\mathrm{tc}_{\mathrm{A,B}}$ between one atom of kind A and one of kind B are precomputed for the given framework volume V and cutoff r_c, but the total tail correction is the sum

$$\mathrm{tc} = \sum_\mathrm{A} \sum_\mathrm{B} \mathrm{N_A N_B tc_{A,B}} \tag{3.5}$$

where N_A and N_B are the number of atoms of kind i and j. Knowing the number of each atom kind per molecule, it is thus possible to precompute the tail correction $\mathrm{tc}(i,j)$ associated with each pair of molecule kinds (i,j), as well as the tail correction $\mathrm{tc}(i)$ due to the interaction of one molecule of kind i and the framework. Those terms being precomputed, the total tail correction can be expressed as

$$\mathrm{tc} = \mathrm{tc}_{\mathrm{framework}} + \sum_i \left(n_i \mathrm{tc}(i) + n_i^2 \mathrm{tc}(i,i) \right) + \sum_i \sum_{j \neq i} n_i n_j \mathrm{tc}(i,j) \tag{3.6}$$

where $\mathrm{tc}_{\mathrm{framework}}$ is the constant tail correction due to the interaction of the framework atoms with each other, and n_i, n_j are the number of molecules of kinds i and j.

Finally, the evolution of the tail correction when adding (+) or removing (−) one particle of kind i is thus

$$\Delta_{\pm}\mathrm{tc} = \pm \mathrm{tc}(i) \pm (2n_i \pm 1)\, \mathrm{tc}(i,i) \pm 2 \sum_{j \neq i} n_j \mathrm{tc}(i,j) \tag{3.7}$$

Resizable lists

Overall, having a variable number of species requires extra bookkeeping, to make sure that all atoms are accounted for. To maximize performance in current computers, the data structures must have good cache locality: this means that, when memory needs to be read, it should be done in contiguous accesses as much as possible. For this purpose, the positions of the atoms, the $\kappa_{\boldsymbol{k}}$ factors, the precomputed $q_n e^{i\boldsymbol{k}\cdot\boldsymbol{r}_n}$ and $S_{\boldsymbol{k}}$, the self-interaction terms, etc., are all stored in tight arrays, with the appropriate indexing for each. When an atom is removed, there are two choices: either leave a gap in the array, and keep track separately of which parts of the array actually refer to a valid data if need be, or move the contents of the array to fill the gap.

Both algorithmic strategies have their merit, and warrant a trial to check which performs best. We did not take the time to implement both however, and instead chose the latter systematically, because it is slightly simpler to reason about. In practice, such an approach consists in replacing the content of the gap in the array by the data at the end of the array, before reducing the size of the array, and thus removing the temporarily duplicated data at the end. The indices relative to the content of the array also need to be updated. The complexity of each array update is thus proportional to the number of elements to remove, which is generally optimal. Insertions are operated at the end of the arrays, with an optimal complexity.

3.3 GCMC on a grid

Even with an efficient implementation, regular GCMC simulation remain resource-consuming, especially in the context of a large scale material screening. Improving this performance is thus valuable, especially if the precision cost can be controlled.

3.3.1 Principle

Similar to the methods exposed in Section 2.4.4, a typical strategy for reducing the computational cost consists in precomputing as much as possible, to decrease the workload of the main loop of the algorithm. In the case of GCMC, the main computation occurring at each move trial is that of the interactions between species, both in direct and reciprocal spaces. Since the

configuration space for the molecules in the unit cell of the framework is continuous, and thus infinite, it is impossible to precompute all the possible pair interactions.

However, that space can be discretized: to do so, the mobile species are forced to occupy one position each among a fixed set of possibilities. The conceptually simplest set is a regular grid: each molecule must reside atop one grid point. For polyatomic species, like all gases of interest with the exception of rare gases, this definition is incomplete however. There are thus three options:

1. the grid can include additional dimensions to account for the orientation of the molecule. In that case there must be a fixed convention dictating which atom of the molecule is placed on the grid point. That atom, called the "bead", is typically that around which the rotation MC move operates.

2. each grid point can come with an associated fixed orientation of the molecule sitting on it. In that case, the lowest energy orientation with respect to the framework is probably the most reasonable choice, but it may grossly misrepresent the actual orientation of the molecules if those are mostly driven by the interaction with other molecules.

3. the molecule can be represented as a single point sitting on the grid, with no orientation. In that case, the interaction energy of the molecule with the framework can be taken as the average of the energies corresponding to the different orientations and, likewise, the molecule-molecule interactions can be computed by taking averages over the orientation of both molecules. These averages could be either arithmetic or Boltzmann averages, depending on the interpretation: a Boltzmann average implicitly assumes that the molecules are at equilibrium, which is not very well defined for a single molecule sitting on a grid point, while an arithmetic average implies has an orientation independent of the corresponding energy, which is not exact either.

Our study is restricted to small gases of the air which are all linear and centrosymmetric: CO_2, N_2, O_2, etc., so the bead is conventionally chosen as the center of the molecule. Choosing an orientation thus amounts to choosing a direction for a rod (the molecule) across possible angles. To accurately cover this configuration space while limiting the number of used angles, we use a Lebedev grid, which is a set of N_l points (θ_i, φ_i) on the surface of the sphere such that the integral of a function f can be well approximated by the sum

$$I[f] = \int_0^{\pi} sin(\theta) d\theta \int_0^{2\pi} d\varphi f(\theta, \varphi) \approx 4\pi \sum_{i=1}^{N_l} w_i f(\theta_i, \varphi_i) \tag{3.8}$$

with the appropriate weights w_i. The values of w_i, θ_i and φ_i have been precomputed once and for all for different values of N_l. Because of orthogonality constraints on the Lebedev points, not all values of N_l are possible: the ones which are usable in my code are N_l = 6, 14, 26, 38, 50, 74, 86, 110, 146, 170, 194, 230, 266, 302, 350, 434, 590, 770, 974, 1202, 1454, 1730, 2030, 2354, 2702, 3074, 3470, 3890, 4334, 4802, 5294 and 5810, allowing for any fine-tuning of the precision level. Only half of these points are actually used since the both the target gases and the Lebedev grids are centrosymmetric.

The number of possible configurations of a molecule is $\alpha \times a \times b \times c$ where (a, b, c) is the size of the spatial grid and α is either N_l in the first of the three options above, or 1 for the second

and third option. Since we use a Van der Waals cutoff of 12 Å, the framework with the smallest grid will necessarily have its perpendicular lengths greater or equal to $2 \times 12\,\text{Å} = 24\,\text{Å}$. Thus, with a distance of 0.2 Å between two grid points, a, b and c are greater or equal to than 120 so the size of the configuration space is at least $\alpha \times 120^3$.

The goal of the grid MC approach is to reduce the amount of computations done at each step, to accelerate the GCMC simulations, by precomputing the pair interactions between grid points. The number of pair interactions to precompute is thus, in theory, at least $\left(\alpha \times 120^3\right)^2/2 = \alpha^2 \times 120^6/2$. It can actually be reduced further with translation symmetry considerations: the interaction between grid points $(x_1, y_1, z_1, \theta_1)$ and $(x_2, y_2, z_2, \theta_2)$ is identical to that between grid points $(x_1 + \Delta, y_1 + \Delta, z_1 + \Delta, \theta_1)$ and $(x_2 + \Delta, y_2 + \Delta, z_2 + \Delta, \theta_2)$ for any Δ. It is thus sufficient to precompute the interaction between $(0, 0, 0, \theta)$ and all other grid points, so the minimum number of precomputations is actually $\alpha^2 \times 120^3 \approx \alpha^2 \times 2 \cdot 10^6$. Each precomputation yields an energy, represented by a floating-point number, which takes 8 B of space, and the typical RAM of a computer is on the order of 10 GB, so the maximum value of α is approximately $\sqrt{10^{10}/\left(8 \times 2 \cdot 10^6\right)} = 25$ for the smallest possible spatial grid.

In conclusion, using explicit angles is theoretically doable only for small enough materials and by using the smallest numbers of angles, otherwise the precomputation results cannot be stored in memory. Moreover, we expect the numerical cost of the precomputations to be considerable itself, since it requires $\alpha^2 \times 2 \cdot 10^6$ pair interaction computations. This is similar to the cost of simulating a system of $N = \alpha^2$ particles in interaction across one million MC steps, which is a typical GCMC simulation length, since each MC trial move requires computing 2N pair interactions (N for the particle before the move, N for the particle after).

3.3.2 Validation

To check the validity of this simulation strategy, we start by comparing the loading of Ar in a zeolite obtained by regular GCMC and through this grid MC approach. The result for zeolite MFI is presented on Figure 3.2, using a grid step size of 0.2 Å. The perfect superposition of the three isotherms indicate that the grid MC simulation method is apt to reproduce the adsorption property of noble gas in zeolites with great accuracy, even at high pressure.

With more complex molecules, angles have to be taken into account, using either of the two last options among the three presented before. Their comparison is illustrated with the adsorption of CO_2 in FAU in Figure 3.3a, shows that averaging orientations (option 3) provide more qualitative results compared to selecting one single orientation per grid point (option 2). Yet, none manage to qualitatively reproduce the adsorption behavior at intermediate to high pressures.

As evoked previously, the nature of the angular averaging, either arithmetic or Boltzmannian, of energies stems from a hypothesis on the distribution of microstates, neither of which is actually valid. The influence of this choice on the simulated loading of CO_2 in FAU is presented in Figure 3.3.

In Figure 3.3a, it appears that the choice of averaging for the molecule-molecule interaction has little influence on the result until the high pressures. Unsurprisingly, the Boltzmann average, which favors attractive interaction, increases the loading compared the arithmetic average,

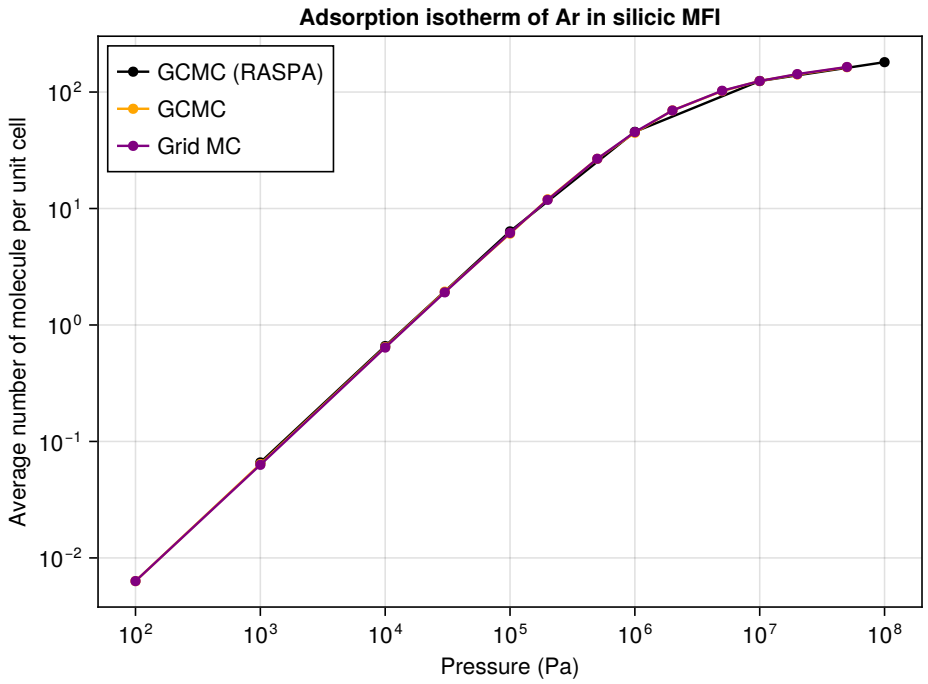


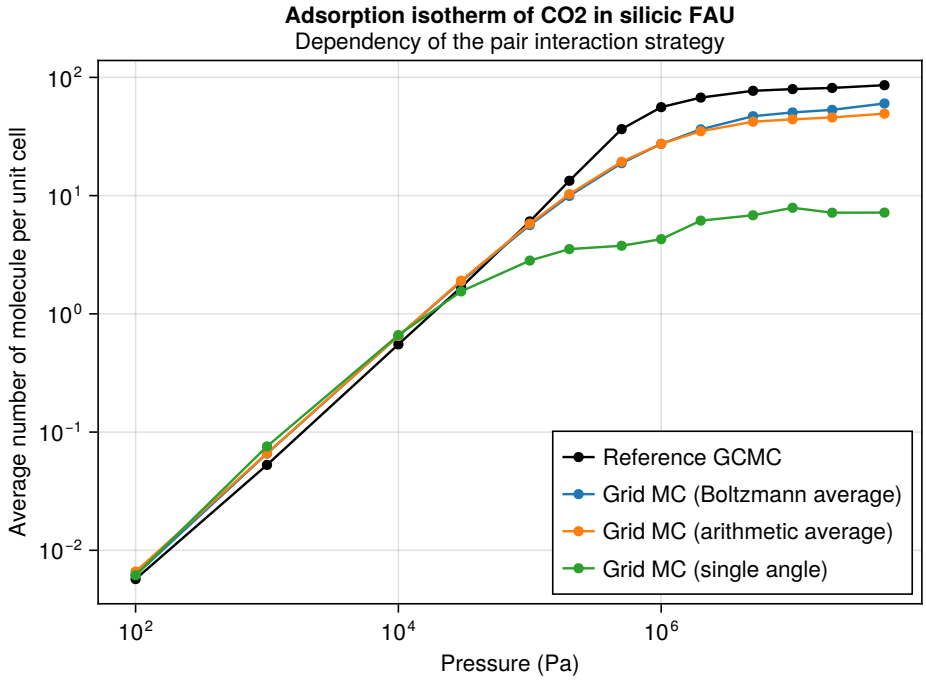
Figure 3.2: Adsorption isotherm of Ar on silicic MFI obtained with different simulation strategies. The black line is the RASPA implementation of GCMC while the orange line is our implementation.

but the general trend is very similar and neither appear to be particularly more right than the other.

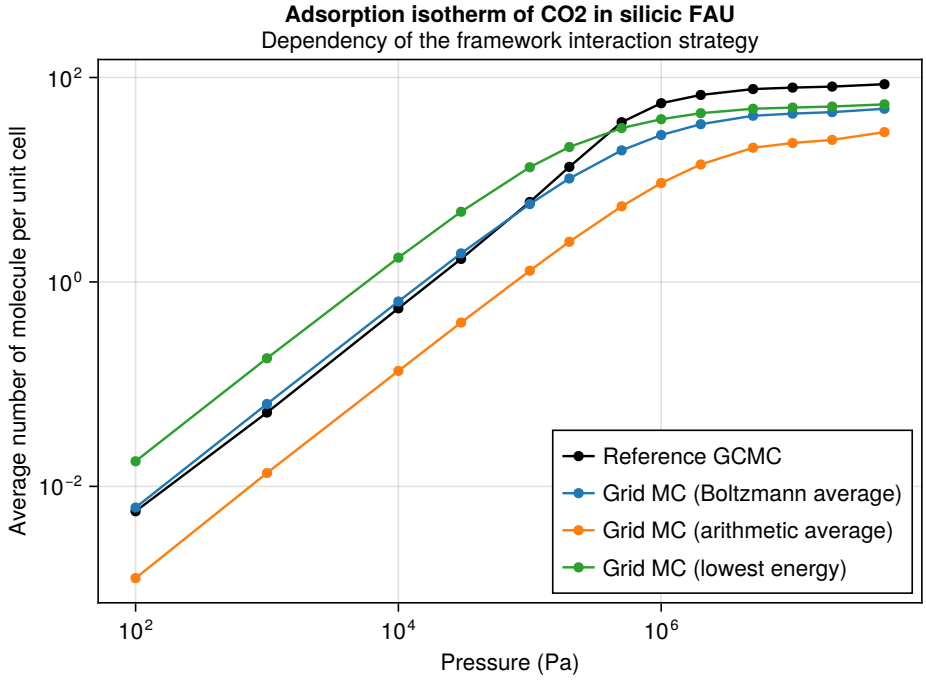
On the other hand, Figure 3.3b illustrates the drastic difference in the choice of averaging for the framework-molecule interaction. Only the Boltzmann average yields an isotherm closely following that of the reference GCMC simulation at low pressures: the arithmetic averaging of angles yields a large underestimation of the loading, while taking only the lowest energy across angles yields a large overestimation.

From these observations, it appears that the molecules of CO_2 preferentially orient themselves according to the interactions with the framework at low pressure, which should be obvious since low pressures correspond to a regime where molecule-molecule interactions are negligible. More importantly, irrespective of the pressure regime, the entropic contribution is paramount: the molecules do not simply orient themselves according to the lowest energy configuration, but freely rotate in accordance with Boltzmann distribution.

Regarding the numerical method itself, the main limit is the behaviour at intermediate and high pressure, which significantly parts from expectation. While this necessarily stems from the use of a grid, it does not depend on the grid step size: Figure 3.4 shows the evolution of the isotherm using option 3 with Boltzmann averaging for framework-molecule interactions and arithmetic averaging for molecule-molecule interactions. The isotherm does not evolve much when the grid step size is small enough, which points to the fact the issue lies in the nature of the simulation method itself.



(a) Influence of the molecule-molecule interaction strategy. "single angle" refers to option 2, while for option 3, the fixed framework-molecule interaction strategy is the Boltzmann average.



(b) Influence of the framework-molecule interaction strategy with option 3. The fixed molecule-molecule interaction strategy is the arithmetic average. "lowest energy" refers to taking the minimum instead of taking an average.

Figure 3.3: Adsorption isotherm of CO_2 on silicic FAU obtained with different grid MC strategies.

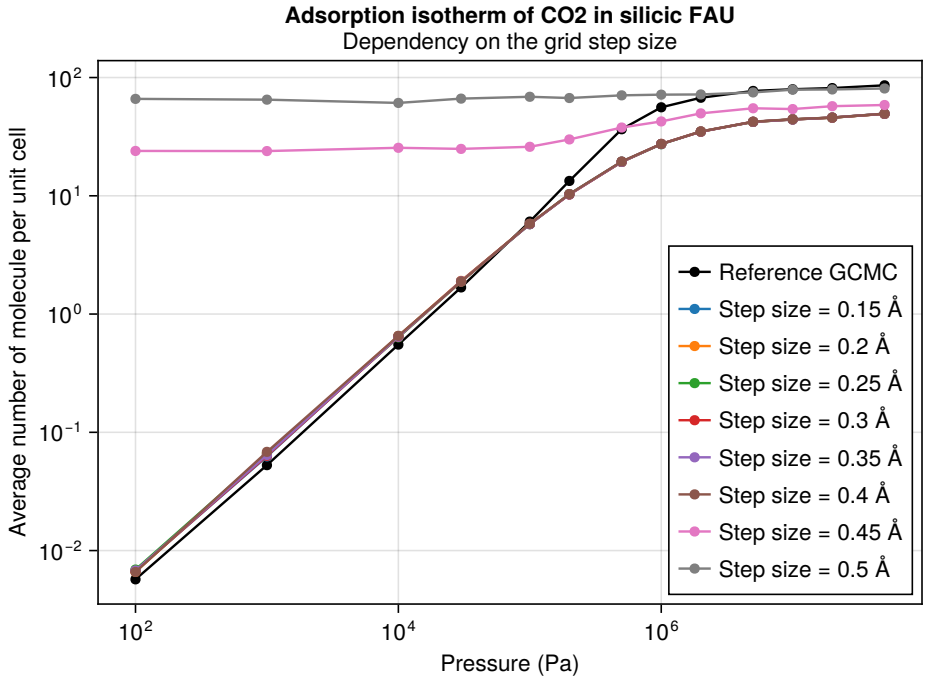


Figure 3.4: Adsorption isotherm of CO_2 on silicic FAU obtained with different grid MC step sizes

3.3.3 Computational aspects

The main interest of the grid MC approach is that all pair interaction energies can be precomputed, which makes each trial move much less computationally costly, and thus the overall simulation much faster. Indeed, the variation of energy due to a single MC trial move can be expressed as

$$\Delta \mathrm{E}^{\text{grid MC}} = \begin{cases} \delta(\mathrm{N}+1, \boldsymbol{r}) \text{ if insertion of a new molecule at } \boldsymbol{r} \\ -\delta(i, \boldsymbol{r}) \text{ if deletion of the molecule } i \text{ at } \boldsymbol{r} \\ \delta(i, \boldsymbol{r}') - \delta(i, \boldsymbol{r}) \text{ if move of the molecule } i \text{ from } \boldsymbol{r} \text{ to } \boldsymbol{r}' \end{cases} \tag{3.9}$$

where N is the number of molecules in the system and

$$\delta(i, \boldsymbol{r}) = \text{framework}(\boldsymbol{r}) + \sum_{j \neq i} \text{interaction}(\boldsymbol{r}, \boldsymbol{r}_j) + \text{tc}_{\text{framework}} + (2\mathrm{N}+1)\text{tc}_{\text{interaction}} \tag{3.10}$$

The different terms above are all precomputed for the different values of the grid point $\boldsymbol{r}$:

- framework($\boldsymbol{r}$) refers to the framework-molecule interaction, which is computed once and for all according to either of the three options discussed before.
- interaction($\boldsymbol{r}, \boldsymbol{r}'$) is the interaction between two molecules, one at $\boldsymbol{r}$ and the other at $\boldsymbol{r}'$.
- $\text{tc}_{\text{framework}}$ is the tail correction for the atoms of the framework and the atoms of the molecule, and $\text{tc}_{\text{interaction}}$ is the tail correction for the atoms of two molecules interacting.

The tail correction terms as stored as simple numbers, framework($\boldsymbol{r}$) is stored as a tridimensional array of energies, $\boldsymbol{r}$ being mapped to a triplet of integers indexing the grid points. The term that

takes the most time to precompute, interaction($\boldsymbol{r}, \boldsymbol{r}'$), is stored in a custom data structure made of a tridimensional index array, along the energy data, in a list. Due to translation symmetry, the query of the interaction between $\boldsymbol{r}$ and $\boldsymbol{r}'$ becomes that between the origin and $\boldsymbol{r} - \boldsymbol{r}'$. The corresponding triplet of integers maps to an index in the tridimensional array, and that index is the one used to retrieve the energy from the energy list. The index can also be 0, to indicate that the interaction energy is zero (i.e. beyond the cutoff): this way, the energy list only stores relevant values.

The precomputation itself is also optimized, to avoid making it the dominant computational cost of running grid MC simulations. To do so, the framework interactions computation benefits from all the precomputations explained in Section 2.4.4, in particular the tabulated atomic grids. For the molecule-molecule interaction averaged over the angles, the energy is interpolated from a linear grid: the interaction energy is computed by putting the bead of the first molecule at the origin and the other at a distance d, where d increments from 0 to the cutoff distance by steps of one fifth of the average MC grid step. The interpolation is done with cubic splines through the `BasicInterpolators.jl` julia package.

4

DATABASE APPROACH

4.1 Screening studies

Research on the adsorption properties of zeolites in particular, and nanoporous materials in general, aims at understanding the nature and respective importance of the different interactions that drive the adsorption phenomenon. This understanding can afterwards be turned into principles, which are then applied to predict the adsorption capacities of similar materials on similar gases, where these principles remain valid.

The natural starting point for this research is a study of a particular material with a particular gas, which leads to particular observations that may then be generalized. Many such studies have been performed and published on zeolites, each involving a considerable amount of work, which provide very precise results on a given topology, with up to a few different Si/Al ratios. Yet, the conclusions can seldom be carried over to provide predictions on other topologies.

The converse strategy consists in doing screening studies, in which a vast array of structures is tackled at the same time. Given the cost of doing so experimentally, these studies are mostly computational, thus providing less precise results than actual experiments, but on a broader scale.

Most zeolite screening studies focus on all-silica zeolites, an approach that sidesteps the issue of placing cations. However, most zeolite adsorption applications require cationic zeolites, since cations critically increase the adsorption capacity of the materials.

Even among the few studies which tackle cationic zeolites across topologies and Si/Al ratios, not all properly place cations in the framework, although it is known that this can lead to incorrect subsequent adsorption measurements.[70] For example, [71] targets 57 different topologies with 5 different possible cations, but fails to mention where these are placed before the start of the Grand Canonical Monte-Carlo (GCMC) simulations, likely resulting in them being trapped in a non-optimal placement. [72] also screens cationic zeolites, but places cations at the so-called "minimum energy positions" instead of using a robust method like parallel tempering; they also use a force field made for FAU zeolites only, which is inadequate to screen other zeolites. We are only aware of the works of [70], which tackles many zeolite with various Si/Al ratios for screening purposes, that properly places cations through parallel tempering.

In this chapter, we explain how we constitute a database of cationic zeolite structures and how it can be used for isotherm prediction.

4.2 Adsorption isotherm

4.2.1 Definition and classification

The behaviour of a material with respect to the adsorption of a gas is typically assessed by measuring the evolution of its adsorption capacity with pressure, at a fixed temperature. The resulting curve is called an isotherm, and it is usually classified according to its general shape into one of several possible categories, represented on Figure 4.1.

The shape of the isotherm depends on the surface of the materials, the strength of its interaction of the adsorbate and of the adsorbate-adsorbate interactions. For example, type I isotherms are often observed on materials with very small external surface and monodisperse pores of

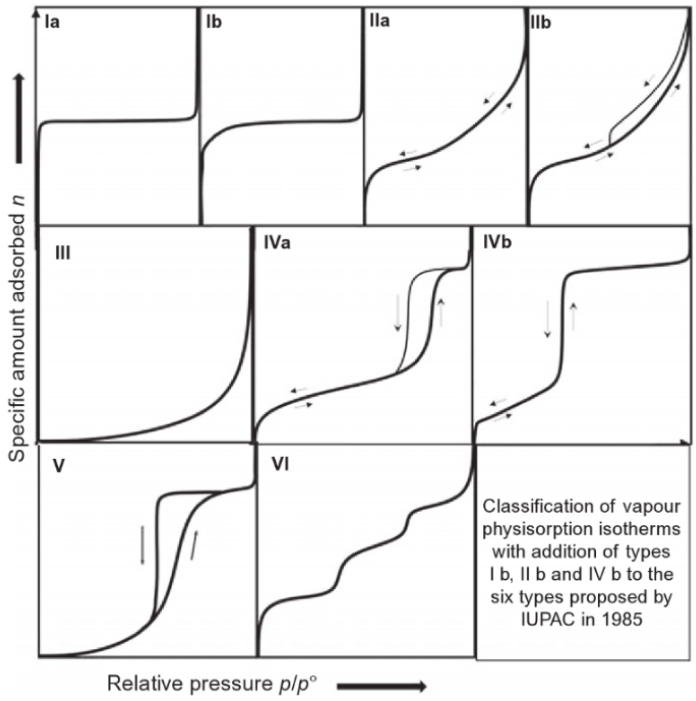


Figure 4.1: Classification of gas isotherm by IUPAC, taken from [74], adapted from [75]. p^0 designates the saturation pressure of the adsorbate.

small size, where adsorption quickly reaches saturation. On the other hand, type III isotherms occur when the adsorbent-adsorbate interaction is weak and adsorption is not limited by the pore size (either because adsorption occurs on the external surface or because the pores are very large). Some of these types exhibit hysteresis loops: for example, type IV(a) isotherms, typical of mesoporous adsorbents, have the adsorption branch lower than the desorption branch (when the gas is withdrawn), which comes from the capillary condensation of the adsorbate at high pressure. Real isotherms are often combinations or slight variations of these idealized types.[73]

At very low pressure, all isotherms are expected to be linear: this is Henry's law, and Henry's adsorption constant K is proportional to the slope of the isotherm. In that limit, all adsorbates behave like ideal gases since, by definition, there is no adsorbate-adsorbate interaction at low enough pressure.

The other end of the pressure spectrum is more complex. When adsorption occurs on the external surface of the adsorbent, for instance with clays, and if adsorbent-adsorbent interactions are strong enough to allow multi-layer adsorption, then the number of adsorbed species never stops growing with increasing pressures. However, with zeolites and more generally any adsorbent where adsorption occurs in finite pores, the adsorption capacity plateaus at high pressure when all the pores are filled with the adsorbate. It should be mentioned though that the adsorption capacity can still increase beyond that limit, because of the compressibility of the adsorbate in liquid or supercritical state. These regimes of extremely high pressure (beyond 100 MPa) are of lesser industrial relevance because reaching these conditions is costly and hazardous. They are, however, visible in simulations.

4.2.2 Experimental observation and numerical prediction

Experimentally, adsorption isotherms are measured by progressively increasing the input pressure of the adsorbate in the material at the fixed temperature, and measuring the amount adsorbed. The desorption isotherm follows up by progressively decreasing the input pressure, and similarly measuring the amount of adsorbate.

The measurement itself usually consists in either of the three following methods:[73]

- volumetry: a fixed amount of gas at the target pressure is put in contact with the adsorbent, and what is measured is the difference of pressure before and after adsorption.
- breakthrough: the gas starts flowing through the adsorbent at a given time, and the outlet concentration and volumetric flowrates are measured.
- gravimetry: the difference in mass of the sample containing the adsorbent before and after adsorption is measured by balancing its weight against buoyancy.

By reproducing the experiment with a reference gas (usually N_2 or Ar), these techniques give access to the net adsorption, which is the absolute amount of adsorbed species minus that which would be in a fluid occupying the same space as the adsorbent at the same pressure and temperature. Knowledge of the volume of adsorbent allows retrieving the absolute adsorption capacity, which is directly accessible in simulations.

Some more anecdotic isotherm measurement techniques including impedance spectroscopy as well as NMR allow obtaining directly the absolute adsorption,[73] but these methods are much too costly to be used routinely.

The adsorption capacity can be predicted numerically through GCMC simulations, as explained in Section 3.2. Beyond the limits due to the precision of the energy computation, either from a force field or from DFT, and convergence of the simulations, those are also dependent on a model of the adsorbent, which cannot take into account the complexity of real materials. In the case of zeolites, the location of the aluminium and of the cations has already been discussed through Chapter 2, but more generally, all kinds of defects and small crystal size effects can lead to experimental isotherms differing from the simulated counterparts. This limits the accuracy of individual numerical simulation compared to precise experimental adsorbents. They remain useful however to extract tendencies and direct the search for new adsorbents, which will be experimentally tested eventually.

4.3 Database constitution

Finding general tendencies in the adsorption behaviour across zeolites requires doing the analysis of many different structures. To do so, the first step consists in establishing a database of models for the materials themselves. The general workflow for creating these structures, starting from the idealized zeolite structure defined by its topology, down to the cation placement, has already been described in Chapter 2. To summarize, the two steps are:

1. Aluminium placement. Start by finding the minimum Si/Al ratio by replacing as many silicium with aluminium atoms as possible, with random exchanges and restarts. Check multiple supercell sizes. Then, starting from these filled structures, replace some alu-

minium by silicium atoms at random and do more exchange steps to generate 6 aluminium placements for each target Si/Al ratio.

2. Cation placement: using either parallel tempering or "shooting star" simulations. The only cation used in the current study is sodium.

For the entire database, we used the same force field which is that developed by [64] for adsorption of small gases on cationic zeolites with mobile cations.

4.3.1 Aluminium placement

The current database contains 6 aluminium placements for all 239 known and not interrupted zeolite topologies, with Si/Al ratios of 1, 1.23, 1.4, 1.7, 2, 3, 5 and 10. The actual Si/Al ratios of each structures are those closest to the previous references, which may sometimes differ. When the minimum Si/Al ratio (presented in Table 2.1) is not among the previous references, it is added separately, and the lower Si/Al ratios cannot be generated. For example, the list of actual Si/Al ratios for topology YUG is 1.56, $\frac{121}{71} \approx 1.704$, 2, 3, 5 and $\frac{349}{35} \approx 9.971$.

The algorithm sometimes fails at finding 6 distinct aluminium placements: in that case, the maximum number of distinct placements is kept. To check whether a new placement is distinct from previous ones, the following algorithm is used. First, for each topology, the list of T-sites is mapped to the list of integers between 1 and N, the number of T-sites of the topology. Each placement is represented by a sequence of N bits, where a bit of 1 at position i indicates an aluminium in T-site i and 0 stands for silicium. Then, all M symmetries of the topology are applied to the placement: this results in M N-bit sequences. The (lexicographically) smallest among those serves as a unique signature that identifies the aluminium placement. When a new placement is found, is signature is computed as previously and compared to those already encountered: if it is different from all of them, the new placement is stored.

In the case where Si/Al = 1 for example, there can be up to only 2 distinct aluminium placements that obeys Löwenstein's rule: taking an arbitrary T-atom as a reference, one placement corresponds to that reference set to Al, and the other to that reference set to Si. For many topologies, these two placements are actually symmetry-equivalent, so they share the same signature and only one of them is actually stored. This is not necessarily the case however: for example, topology JSW has two distinct aluminium placements for Si/Al = 1.

Overall, 8983 different structures with aluminium placed were generated.

4.3.2 Cation placement

For all 239 known and non-interrupted topologies, and for each of the up to 6 different aluminium placements corresponding to the minimum Si/Al, a "shooting star" simulation was launched with the hot run at 2000 K for 20 000 cycles (and 2000 initialization cycles), spawning one cold simulation every 100 cycles. Each of the resulting 200 cold simulations ran at 300 K for 10 000 cycles. This served as an experiment to assess the convergence of the simulations: out of the 239 topologies, 57 converged in less than 5000 cycles (averaged across aluminium placements). For these topologies, "shooting star" simulations were run for all previously created aluminium placements across the different Si/Al ratios, using the same parameters (except for the number of cold cycles raised to 15 000). For each "shooting star" simulation, the 200 cold simulations are divided into six groups, and for each of these groups, the structure with

the lowest corresponding energy is kept. Hence, this methodology yields 6 cation placement per structure.

This methodology is a reduced version of the full cation placement strategy illustrated on the FAU topology in Section 2.3.1. Such a strategy starts by identifying cationic sites using the previous simulation results, then follows up by running site hopping simulations augmented by "shooting star" (or another meta-algorithm) for each desired Si/Al ratio and aluminium placement. This ensures the optimal repartition of the cations in the different sites, although the site locations themselves may not be perfectly adequate for each precise structure, since they are identified based on a cation density map averaged over different structures. The 6 placements yielding the lowest energy should thus be optimized again by running one short MC simulation each, starting from that placement and only using local translation moves. Finally, for each of these 6 simulations, one of the cycles after energy convergence picked at random can be used to yield a representative cation placement.

At the time of writing, this full methodology was not deployed yet. Although it requires more implementation effort, it should not incur a considerable computational cost compared to the initial shooting star simulations. The main reason for this is the speed of the site hopping simulations, which is considerably faster than a regular simulation.

Overall, 14 268 different structures with both aluminium and cations placed were obtained.

4.4 Prediction

Using the structures from the database, it is possible to study the adsorption behaviour of zeolites at scale, across topologies and Si/Al ratios. To do so, one simply needs to run a GCMC simulation for each combination of zeolite structure (including Si/Al ratio, aluminium placement, nature and placement of the cations), temperature, pressure, and gas.

In order to investigate the relationship between the topology of zeolites and their adsorption properties, we attempted to build a predictive model that could output a plausible isotherm for each combination of three parameters, which are 1) the zeolite topology, 2) the Si/Al ratio and 3) the temperature, using Na^+ as cation and either CO_2 or N_2 as gas.

4.4.1 Why isotherms?

Adsorption isotherms are often used for the characterization of experimental structures. The technique consists in measuring the isotherm using one of the methods presented in Section 4.2.2, then fitting the obtained curve against a set of pre-computed isotherms, called the kernels. Each kernel is computed based on a pore geometry (*i.e.* its shape, like a slit or a sphere, and its size) and the thermodynamical properties of the gas, often modeled using simple fluid theory. Fitting the isotherm then corresponds to finding coefficient on each kernel such that the sum of the kernels weighted by the coefficients yields the best approximation of the isotherm. The best fit thus provides the repartition of pore geometries based on their corresponding coefficients.

Another typical context of use for isotherm fitting is for the identification of a material, or the evaluation of its purity or degree of crystallization. To do so, a simple method consists in measuring one or more isotherms and comparing them to those obtained on a reference

structure, to check if they match. In general data science, two curves can be compared by computing their root-mean-square deviation (RMSD), but this does not provide much physical meaning to the comparison, and requires that both isotherms have points measured at the same pressures. More often, the reference isotherm has been fitted against one model, hence the isotherms of the experimental structure can be fitted against the same model, so that the coefficients of the model, which carry some physical meaning, can be compared.

In our setting, the goal is quite different since we aim at predicting the adsorption capacity from a theoretical structure, instead of an experimental one. The concept of isotherm is actually not strictly necessary to this goal: one could imagine making a black-box model that directly outputs the adsorption capacity given the three parameters (topology, Si/Al ratio, temperature) as well as the gas pressure. This approach, however, leads to a model that cannot be physically interpreted, and that offers no guarantee on the general shape of the isotherms, with the risk of them being unphysical. On the contrary, making a model that directly predicts the coefficients of a given isotherm model from the three parameters yields a result that already embeds the physics of adsorption through the isotherm mode. Of course, it also offers the additional guarantee of a smooth and plausible interpolation of the simulated data points across pressures.

To make such a model, the first step thus consists in fitting the simulated isotherm against a given adsorption model, in order to obtain the coefficients associated with each triplet of parameters. Retrieving the relation between the parameters and the coefficient can then be cast as a data science problem.

4.4.2 Adsorption models

The literature is rife with adsorption models, either grounded in theory or empirical ones. The precise choice of a model for an isotherm stems from nature of the adsorption when known, otherwise the shape of the isotherm, the nature of the material or even the scientific community. Since we use isotherm models simply as a mean for extracting a common mathematical descriptor of the isotherms across numerous settings, our only criterion is that the model should be rich enough to accurately represent the isotherms. Beyond that, the simpler the model, *i.e.* the smaller the number of coefficients necessary to represent an isotherm, the better it will be for prediction.

Next are the isotherm models which were investigated. This list is not exhaustive, but it covers a sizeable proportion of the most commonly used models for small gas adsorption in zeolites.[76]

Langmuir

The simplest adsorption model corresponds to a physical situation where the adsorbate behaves as an ideal gas, and binds with the adsorbent in a reversible reaction of equilibrium constant β. In that setting, the Langmuir adsorption model states that the fraction of occupied adsorption sites is $\beta P/(1 + \beta P)$ where P is the pressure, hence the total adsorption capacity is:

$$n_{\text{Langmuir}}(P) = \alpha \frac{\beta P}{1 + \beta P} \tag{4.1}$$

where α is the maximum population of the site.

This very simple model is not accurate for zeolites however. One of the reasons is that zeolites contain multiple adsorption sites, both spatially – there are multiple adsorption reactions happening simultaneously – and energetically – they have different values of β.

N-site Langmuir

One simple way to circumvent the previous issues consists in simply summing multiple Langmuir adsorption models. This physically assumes some kind of independence between the different adsorption sites, which is debatable, and the general formula is

$$n_{\text{N-site Langmuir}}(\text{P}) = \sum_{i=1}^{\text{N}} \alpha_i \frac{\beta_i \text{P}}{1 + \beta_i \text{P}} \tag{4.2}$$

Such a general model can be made to fit any isotherm, simply by increasing the number of sites N until reaching a high enough complexity of the model. In practice, N = 3 is good enough to fit many isotherms, simply because the fit can be optimized across six coefficients, although the underlying physical model may not be accurate at all.

Freundlich

This simple empirical model is commonly used when the adsorption is known to happen on a heterogeneous surface, and depends on two parameters A and γ:

$$n_{\text{Freundlich}}(\text{P}) = \text{A} \times \text{P}^{\gamma} \tag{4.3}$$

Beyond its simplicity, the main limit of the model is its behaviour at high pressure: the model diverges to $+\infty$ whereas, in practice, adsorption reaches saturation (before the highly condensed phase).

Sips

The Sips model combines both Langmuir and Freundlich isotherms, with the following formula:

$$n_{\text{Sips}}(\text{P}) = \alpha \frac{(\beta \text{P})^{\gamma}}{1 + (\beta \text{P})^{\gamma}} \tag{4.4}$$

This model has two limit regimes: at low pressure, it is equivalent to the Freundlich model while at high pressure, it reaches a plateau similarly to the Langmuir model. It also reduces to the Langmuir model in the particular case of $\gamma = 1$.

Toth

The Toth model is another empirical variation of the Langmuir model:

$$n_{\text{Toth}}(\text{P}) = \alpha \frac{\beta \text{P}}{\left(1 + (\beta \text{P})^{\gamma}\right)^{1/\gamma}} \tag{4.5}$$

Like the Sips model, it reduces to the Langmuir model for $\gamma = 1$, and reaches a plateau at high pressure. A slight difference with the Sips model is the adsorption capacity at low pressure, which becomes proportional to P and thus obeys Henry's law, whereas both Freundlich and Sips models have a dependency in P^{γ}.

Redlich-Peterson

Like the Sips model, the Redlich-Peterson model combines both Langmuir and Freundlich models:

$$n_{\text{Redlich-Peterson}} = \alpha \frac{\mathrm{P}}{1 + (\beta \mathrm{P})^{\gamma}} \tag{4.6}$$

It also follows Henry's law by reducing to a linear function of P at low pressure, but its behaviour at high pressure is similar to Freundlich's, making it only adapted to pressure regions below the saturation point.

Jensen-Seaton

By adding a fourth parameter, Jensen and Seaton propose an adsorption model that present both Henry's law linear behaviour at low pressure, as well as a non-constant affine behaviour at high pressure that models the compressibility region.

$$n_{\text{Jensen-Seaton}} = \alpha\beta \mathrm{P} \times \left(1 + \left(\frac{\beta \mathrm{P}}{1 + \delta \mathrm{P}}\right)^{\gamma}\right)^{-1/\gamma} \tag{4.7}$$

This model reduces to the Toth model when $\delta = 0$ and to the Langmuir model when $\gamma = 1$ (with second parameter $\beta' = \beta + \delta$).

Sips-Toth

A custom-made adsorption model, absent from the literature, consists in using a fourth parameter ζ to act as a continuous switch between the Toth model ($\zeta = 0$) and the Sips model ($\zeta \to +\infty$):

$$n_{\text{Sips-Toth}} = \alpha \times \left(\frac{(\beta \mathrm{P})^{\gamma}}{1 + (\beta \mathrm{P})^{\gamma}}\right)^{\frac{1+\gamma\zeta\mathrm{P}}{\gamma(1+\zeta\mathrm{P})}} \tag{4.8}$$

Like its two building blocks, this model collapses to a simple Langmuir isotherm when $\gamma = 1$, and reaches a plateau at high pressure. It also obeys Henry's law at low pressure.

Summary

Table 4.1 summarizes the key properties of the previously presented isotherm models. The number of coefficients characterizes the complexity of the model. Its behaviour at low pressure should obey Henry's law, while it should be affine at very high pressure to correctly model the compressibility phase, or at least behave as a plateau otherwise to account for the stationary phase at high pressures.

Several of these models can be combined by simply adding them together. For this purpose, one extra dummy model was added, called "Linear", which simply consists in doing a linear regression of the isotherm:

$$n_{\text{Linear}} = \eta \times \mathrm{P} \tag{4.9}$$

Of course, this model does not fit any actual isotherm by itself, but it can be added to any model that reaches a plateau at high pressure to fix its behaviour for the compressibility phase, by making it affine as expected.

The result of isotherm fitting according to all of these models, with and without linear addition, is represented in Figure 4.2.

Name	Coefficients	Henry's law at low P	Behaviour at high P
Linear	1	No	Affine
N-site Langmuir	2N	Yes	Plateau
Freundlich	2	No	Polynomial
Sips	3	No	Plateau
Toth	3	Yes	Plateau
Redlich-Peterson	3	Yes	Polynomial
Jensen-Seaton	4	Yes	Affine
Sips-Toth	4	Yes	Plateau

Table 4.1: Main characteristics of different adsorption models

The quality of the fit generally increases with the number of parameters, simply because of the increased flexibility of the model. Similarly, adding one extra linear parameter improves the quality of the overall fit, in particular at high pressure, since it corresponds to the linear compressibility regime. The case of the Jensen-Seaton adsorption model complements this point: there is no improvement in the fit due to the linear parameter in that case, because Jensen-Seaton isotherms already behave linearly at high pressure.

However, not all models perform equivalently. For example, even with three sites and thus six parameters, Langmuir fits very poorly compared to the Sips-Toth model plus linear term, which is the overall best and only require five parameters. With only four parameters, Toth plus linear is equivalent to Jensen-Seaton. Both Freundlich and Redlich-Peterson fail to fit because of the polynomial behaviour at high pressure.

4.4.3 Isotherm fitting

In practice, fitting an isotherm means finding the parameters α, β, γ, ... of the model such that the theoretical isotherm resulting from the model is as close as possible to the input one. This closeness can be properly defined through the RMSD for instance. Finding the best parameters can thus be formulated as an optimization problem, *i.e.* a problem of the form

$$\text{minimize } f(\alpha, \beta, \gamma) \quad \text{such that } \alpha > 0, \beta > 0, 0 < \gamma < 5 \tag{4.10}$$

for example, where f represents the deviation between the model with the given parameters and the actual isotherm.

Unfortunately, the expressions of the different adsorption models make this problem non-convex, which is a category of optimization problems generally considered difficult. To solve it, we use the `LsqFit.jl` and `BlackBoxOptim.jl` Julia packages to do global optimization. Crucially, these two packages do not rely on f being differentiable, since that would require finding the derivative of the deviation with respect to the parameters for all investigated adsorption models and their combination.

Both packages use meta-heuristic and stochastic algorithms to perform the actual optimization, but they need a starting point, *i.e.* an initial value for the parameters, to work. The result often depends on the quality of this initial point: the closer it is to the optimum, the better the result. Hence, we tried to use `LsqFit.jl`, with an initial set of parameters found from linearizations of the model:

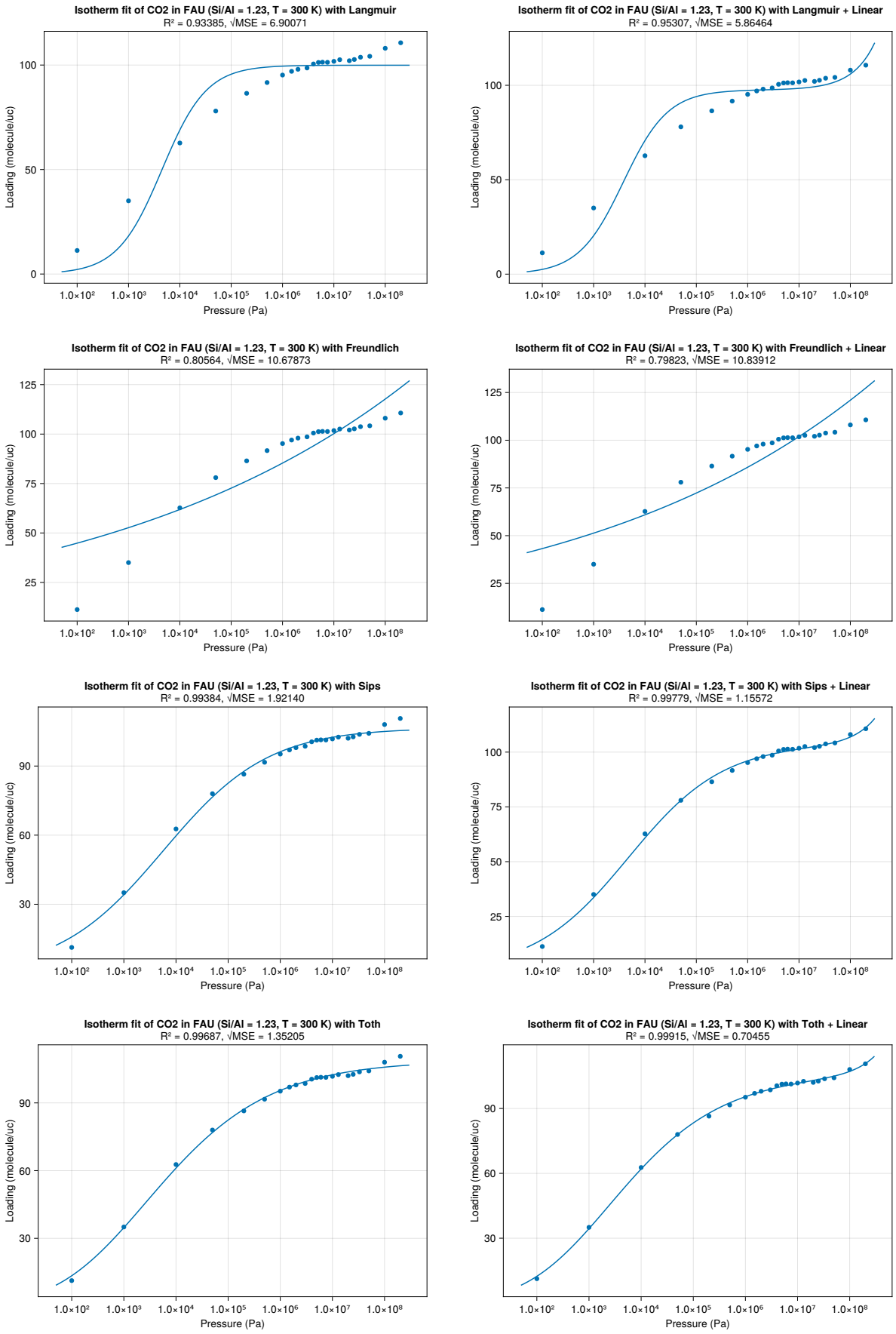


Figure 4.2: Simulated adsorption isotherm fitted with different models

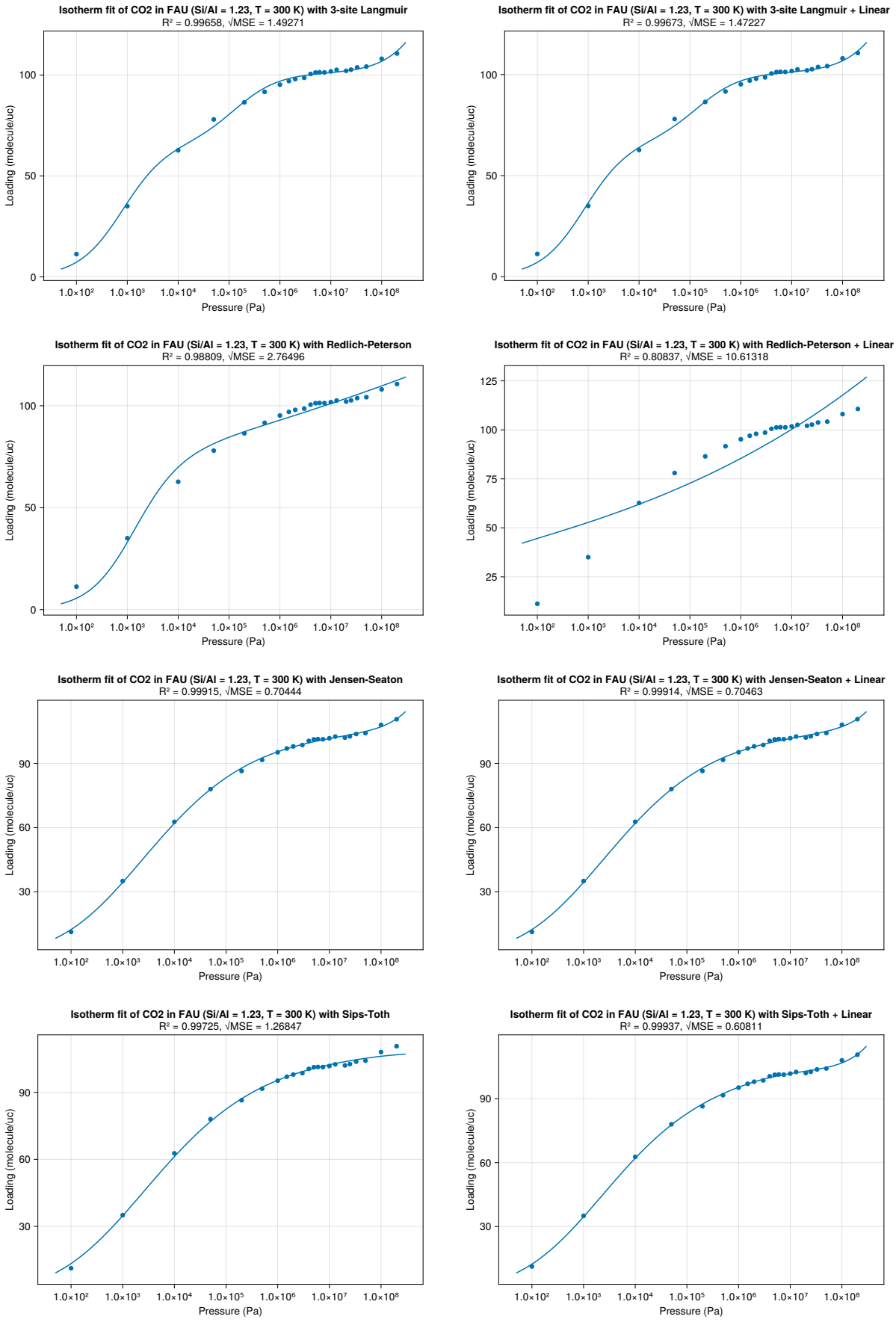


Figure 4.2: Simulated adsorption isotherm fitted with different models (continued)

- For Langmuir, the linear regression $n^{-1} = f(\mathrm{P}^{-1})$ yields $n^{-1}_{\text{Langmuir}} = {\alpha'}_0^{-1}\left(1 + {\beta'}_0^{-1}\mathrm{P}^{-1}\right)$. We then test three different initial points: $\alpha_0 = \alpha'_0 \times \theta$ and $\beta_0 = \beta'_0/\theta$ with $\theta = 0.833$, 1.44 and 0.579.
- For Freundlich, the linear regression $\log n = f(\log \mathrm{P})$ directly gives $\log n_{\text{Freundlich}} = \log \mathrm{A}_0 + \gamma_0 \log \mathrm{P}$.
- For Sips, a first linear regression $\log n = f(\log \mathrm{P})$ gives an approximate value γ_0, then a second linear regression $n^{-1} = f(\mathrm{P}^{\gamma_0})$ yields $n^{-1}_{\text{Sips}} = \alpha_0\left(1 + \beta^{-\gamma_0}\mathrm{P}^{-\gamma_0}\right)$.
- For Toth, we take the three initial points α_0 and β_0 from the Langmuir isotherm and test $\gamma_0 = 0.667$, 2.25 and 0.296 respectively.
- For Redlich-Peterson, a first linear regression $\log(\mathrm{P}/n) = f(\log \mathrm{P})$ gives $\log(\mathrm{P}/n_{\text{Redlich-Peterson}}) = \gamma_0 \log \mathrm{P} + \mathrm{C}$, then a second linear regression $\mathrm{P}/n = f(\mathrm{P}^{\gamma_0})$ yields $\mathrm{P}/n_{\text{Redlich-Peterson}} = \alpha_0^{-1}\left(1 + \beta_0^{\gamma_0}\mathrm{P}^{\gamma_0}\right)$.
- For Jensen-Seaton, we try the three sets of initial parameters α_0, β_0 and γ_0 taken with the method for Toth, and fix $\delta_0 = 8 \times 10^{-10}$.
- For Sips-Toth, we try both Sips and Toth parameters, with the switching parameter ζ_0 equal to either 1, 10^{-5} or 10^{-10}.

When adding a linear term to a model, the linear slope is taken from a linear regression of the four last point in the isotherm, then the rest of the initial point is taken from the isotherm minus this initial slope.

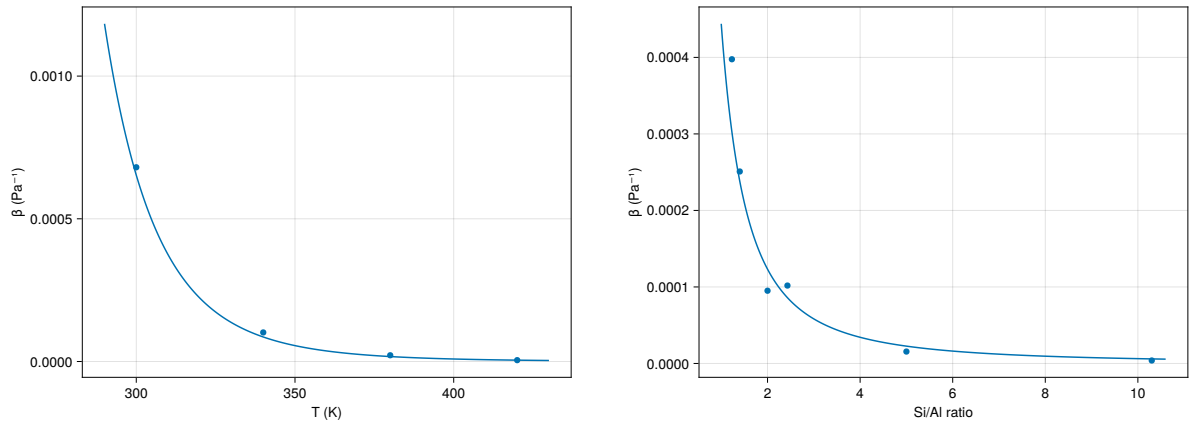
Unfortunately, this rational but simplistic method does not give good results even for simple models, because the initial points are too far from the optimum. A more systematic approach consists in checking a larger set of initial parameters with the `BlackBoxOptim.jl` package. We then use the best set of parameters found as the initial point for an `LsqFit.jl` optimization, and keep whichever set of parameters yields the lowest RMSD.

The only requirement for `BlackBoxOptim.jl` is to delimit the minimum and maximum values for each parameter. We use $0\,\mathrm{Pa}^{-1} < \beta < 1\,\mathrm{Pa}^{-1}$, $0 < \gamma < 5$, $0\,\mathrm{Pa}^{-1} < \delta < 1\,\mathrm{Pa}^{-1}$, $0\,\mathrm{Pa}^{-1} < \zeta < 0.5\,\mathrm{Pa}^{-1}$, and $2\mathrm{M}/3 < \alpha < 4\mathrm{M}/3$ where M is the maximum value of the isotherm. Overall, we find that this approach, starting with `BlackBoxOptim.jl` to have an initial point followed by `LsqFit.jl` for fine-grained optimization, gives the best result.

External to the Julia ecosystem, it is also possible to perform isotherm fitting using proprietary softwares like OriginLab, or other free open-source alternatives like the PyGAPS package in Python.[77] The latter provides isotherm fitting capability for (up to 3-site) Langmuir, Freundlich, Toth, Jensen-Seaton and a few other models, but not Sips, Redlich-Peterson, nor custom functions like the Sips-Toth model presented before. The PyGAPS framework is flexible enough to allow adding custom models however, so it could be interesting to compare the quality of our fits with theirs.

4.4.4 Simple models

Once a sufficient number of fitted isotherms are available, the next goal is to somehow interpolate across them across Si /Al ratio and temperatures for a given topology. Through fitting,



(a) Evolution with temperature for Si/Al = 2.43

(b) Evolution with Si/Al ratio for T = 340 K

Figure 4.3: Evolution of the β parameter in the Jensen-Seaton fit of CO_2 adsorption isotherms on FAU. The full line is the simple model obtained in Equation (4.12).

this problem is reduced to finding function $\alpha(r, \mathrm{T})$, $\beta(r, \mathrm{T})$, $\gamma(r, \mathrm{T})$, … *i.e.* to expressing each parameter of the adsorption model as a function of the Si/Al ratio r and the temperature T.

For each parameter p, we tried to express $p(r, \mathrm{T})$ through simple models, using `GLM.jl`, a Julia package for fitting linear models. In detail, we tried to fit expressions of the form

$$f_p^{(p)}(p) = \mathrm{A}^{(p)} \times f_r^{(p)}(r) + \mathrm{B}^{(p)} \times f_\mathrm{T}^{(p)}(\mathrm{T}) + \mathrm{C}^{(p)} \tag{4.11}$$

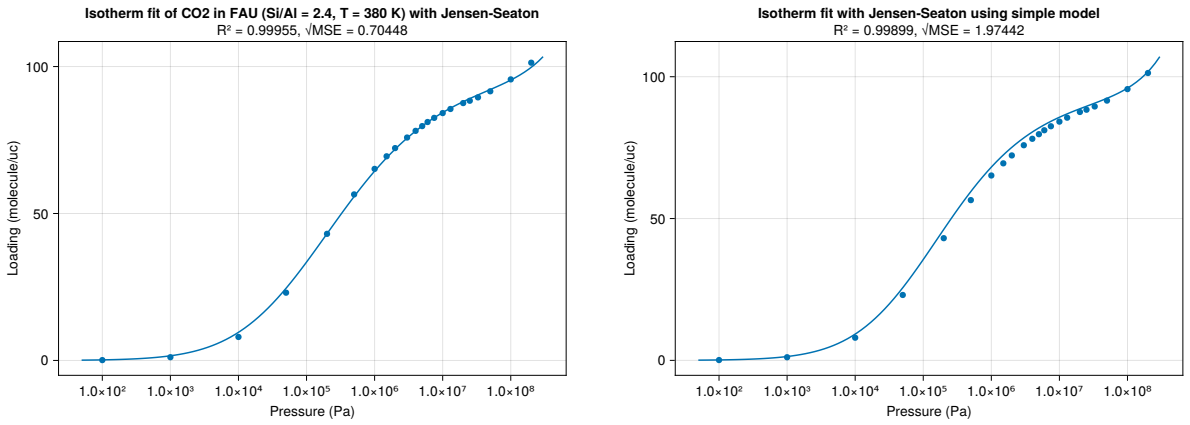
where each function $f^{(p)}$ was either the identity, the inverse, or the log. Such a fit consists in finding the optimal parameters $\mathrm{A}^{(p)}$, $\mathrm{B}^{(p)}$ and $\mathrm{C}^{(p)}$ to minimize the RMSD between both sides of the expected equality. There are 27 such expressions to test for each parameter p since each of the three function $f^{(p)}$ only has three choices, and the expression which yields the lowest RMSD is kept as the model, along with the corresponding parameters.

For example, using the Jensen-Seaton adsorption model for CO_2 adsorption in FAU zeolites, the following expression is found for the β parameter:

$$\log \beta(r, \mathrm{T}) = -22.927688538843174 - 1.8489495470102486 \log r + 5170.651401050577/\mathrm{T} \tag{4.12}$$

Figure 4.3 represents the comparison between the interpolated value of $\beta(r, \mathrm{T})$ and the values collected from the isotherms fitting in the previous setting. In this case, the fit works well, but the simple model above does not always provide such a good level of agreement.

Overall, this approach provides a simple model for the evolution of the parameters which works reasonably well to interpolate an isotherm. For example, Figure 4.4 shows the difference in quality between the direct Jensen-Seaton fit of an isotherm, and the one obtained from the simple model. In this example, the simple model was built using Si/Al ratios of 1, 1.23, 1.4, 2, 5 and 10 and temperatures of 300 K, 340 K and 420 K, thus excluding the investigated point with Si/Al ratio of 2.4 at T = 380 K. While not as quantitative as the actual fit, the simple model allows retrieving an isotherm which is close to the reference without needing any additional computation.



(a) Direct Jensen-Seaton fit

(b) Interpolated fit using simple model

Figure 4.4: Fitted adsorption isotherms of CO_2 in FAU with Si/Al = 2.4 at T = 380 K

4.5 Perspectives

At the time of writing, we have only computed gas adsorption isotherm on three topologies: CHA, LTA and FAU. The previous methodology using simple models to interpolate isotherms constitute a convenient method to obtain adsorption capacities given enough data for the target topology, but does not allow generalizing to other topologies. To reach this target, more isotherm data is required on various topologies, from which point it would become possible to create yet another simple model whose target are the coefficients A, B and C of the previous simple model, depending only on the topology.

Another approach consists in relying on the more general tools of machine learning, instead of the custom approach detailed before. Deep learning could be used as a black-box method to directly learn the $p(r, \mathrm{T}, \tau)$ functions, where τ designates the framework topology, numerically represented using ring statistics or coordination sequences for instance. Yet, once again, this requires much more isotherm data than what is currently available.

Fortunately, our cationic zeolite structure database now contains 64 topologies with representative structures across Si/Al ratios. Once the adsorption isotherms are computed on all of these structures, it should provide enough data to start creating a more robust predictive model.

LIST OF PUBLICATIONS

1. Lionel Zoubritzky and François-Xavier Coudert. “CrystalNets.jl: Identification of Crystal Topologies”. In: *SciPost Chem.* 1 (2022), p. 005. DOI: 10.21468/SciPostChem.1.2.005.

BIBLIOGRAPHY

[1] N. W. Ockwig, O. Delgado-Friedrichs, M. O'Keeffe, and O. M. Yaghi. "Reticular Chemistry: Occurrence and Taxonomy of Nets and Grammar for the Design of Frameworks". In: *ChemInform* 36.24 (2005). ISSN: 0931-7597. DOI: 10.1002/chin.200524292.

[2] O. M. Yaghi, M. O'Keeffe, N. W. Ockwig, H. K. Chae, M. Eddaoudi, and J. Kim. "Reticular Synthesis and the Design of New Materials". In: *Nature* 423.6941 (2003), pp. 705–714. ISSN: 0028-0836. DOI: 10.1038/nature01650.

[3] D. J. Tranchemontagne, J. L. Mendoza-Cortés, M. O'Keeffe, and O. M. Yaghi. "Secondary Building Units, Nets and Bonding in the Chemistry of Metal–Organic Frameworks". In: *Chem. Soc. Rev.* 38.5 (2009), p. 1257. ISSN: 0306-0012. DOI: 10.1039/b817735j.

[4] H. Furukawa, Y. B. Go, N. Ko, Y. K. Park, F. J. Uribe-Romo, J. Kim, M. O'Keeffe, and O. M. Yaghi. "Isoreticular Expansion of Metal–Organic Frameworks with Triangular and Square Building Units and the Lowest Calculated Density for Porous Crystals". In: *Inorg. Chem.* 50.18 (2011), pp. 9147–9152. ISSN: 0020-1669. DOI: 10.1021/ic201376t.

[5] S. Bureekaew, V. Balwani, S. Amirjalayer, and R. Schmid. "Isoreticular Isomerism in 4,4-Connected Paddle-Wheel Metal–Organic Frameworks: Structural Prediction by the Reverse Topological Approach". In: *CrystEngComm* 17.2 (2015), pp. 344–352. DOI: 10.1039/c4ce01574f.

[6] J. Kim, B. Chen, T. M. Reineke, H. Li, M. Eddaoudi, D. B. Moler, M. O'Keeffe, and O. M. Yaghi. "Assembly of Metal–Organic Frameworks from Large Organic and Inorganic Secondary Building Units: New Examples and Simplifying Principles for Complex Structures". In: *J. Am. Chem. Soc.* 123.34 (Aug. 2001), pp. 8239–8247. ISSN: 0002-7863. DOI: 10.1021/ja010825o.

[7] M. O'Keeffe, M. A. Peskov, S. J. Ramsden, and O. M. Yaghi. "The Reticular Chemistry Structure Resource (RCSR) Database of, and Symbols for, Crystal Nets". In: *Acc. Chem. Res.* 41.12 (2008), pp. 1782–1789. ISSN: 0001-4842. DOI: 10.1021/ar800124u.

[8] M. O'Keeffe and O. M. Yaghi. "Deconstructing the Crystal Structures of Metal–Organic Frameworks and Related Materials into Their Underlying Nets". In: *Chem. Rev.* 112.2 (2012), pp. 675–702. ISSN: 0009-2665. DOI: 10.1021/cr200205j.

[9] S. R. Batten. "Topology and Interpenetration". In: *Metal-Organic Frameworks: Design and Application*. John Wiley & Sons, MacGillivray, L. R., 2010, pp. 91–130.

[10] S. Hyde, O. D. Delgado-Friedrichs, S. Ramsden, and V. Robins. "Towards Enumeration of Crystalline Frameworks: The 2D Hyperbolic Approach". In: *Solid State Sci.* 8.7 (2006), pp. 740–752. ISSN: 1293-2558. DOI: 10.1016/j.solidstatesciences.2006.04.001.

[11] S. Ramsden, V. Robins, and S. Hyde. "Three-Dimensional Euclidean Nets from Two-Dimensional Hyperbolic Tilings: Kaleidoscopic Examples". In: *Acta Cryst. A* 65.2 (2009), pp. 81–108. DOI: 10.1107/s0108767308040592.

[12] J. C. Tan, T. D. Bennett, and A. K. Cheetham. "Chemical Structure, Network Topology, and Porosity Effects on the Mechanical Properties of Zeolitic Imidazolate Frameworks". In: *Proc. Nat. Acad. Sci.* 107.22 (2010), pp. 9938–9943. ISSN: 0027-8424. DOI: 10.1073/pnas.1003205107.

[13] V. A. Blatov, A. P. Shevchenko, and D. M. Proserpio. "Applied Topological Analysis of Crystal Structures with the Program Package ToposPro". In: *Cryst. Growth Des.* 14.7 (2014), pp. 3576–3586. ISSN: 1528-7483. DOI: 10.1021/cg500498k.

[14] V. A. Blatov. "Multipurpose Crystallochemical Analysis with the Program Package TOPOS". In: *IUCr CompComm Newsletter* 7.4 (2006), pp. 4–38.

[15] O. Delgado-Friedrichs and M. O'Keeffe. "Identification of and Symmetry Computation for Crystal Nets". In: *Acta Cryst. A* 59.4 (2003), pp. 351–360. ISSN: 0108-7673. DOI: 10.1107/s0108767303012017.

[16] E. V. Alexandrov, A. P. Shevchenko, and V. A. Blatov. "Topological Databases: Why Do We Need Them for Design of Coordination Polymers?" In: *Cryst. Growth Des.* 19.5 (2019), pp. 2604–2614. ISSN: 1528-7483. DOI: 10.1021/acs.cgd.8b01721.

[17] S. R. Hall, F. H. Allen, and I. D. Brown. "The Crystallographic Information File (CIF): A New Standard Archive File for Crystallography". In: *Acta Cryst. A* 47.6 (1991), pp. 655–685. ISSN: 0108-7673. DOI: 10.1107/s010876739101067x.

[18] O. Delgado-Friedrichs and M. O'Keeffe. "Crystal Nets as Graphs: Terminology and Definitions". In: *J. Sol. State Chem.* 178.8 (2005), pp. 2480–2485. ISSN: 0022-4596. DOI: 10.1016/j.jssc.2005.06.011.

[19] V. Blatov, M. O'keeffe, and D. Proserpio. "Vertex-, Face-, Point-, Schläfli-, and Delaney-symbols in Nets, Polyhedra and Tilings: Recommended Terminology". In: *CrystEngComm* 12.1 (2010), pp. 44–48. DOI: 10.1039/b910671e.

[20] O. Delgado-Friedrichs, S. T. Hyde, M. O'Keeffe, and O. M. Yaghi. "Crystal Structures as Periodic Graphs: The Topological Genome and Graph Databases". In: *Struct. Chem.* 28.1 (2017), pp. 39–44. ISSN: 1572-9001. DOI: 10.1007/s11224-016-0853-3.

[21] O. Delgado-Friedrichs. "Barycentric Drawings of Periodic Graphs". In: *International Symposium on Graph Drawing*. Springer, 2003, pp. 178–189.

[22] M. Treacy, M. Foster, and K. Randall. "An Efficient Method for Determining Zeolite Vertex Symbols". In: *Micropor. Mesopor. Mater.* 87.3 (2006), pp. 255–260. ISSN: 1387-1811. DOI: 10.1016/j.micromeso.2005.08.024.

[23] V. A. Blatov, O. Delgado-Friedrichs, M. O'Keeffe, and D. M. Proserpio. "Three-Periodic Nets and Tilings: Natural Tilings for Nets". In: *Acta Cryst. A* 63.5 (Sept. 2007), pp. 418–425. ISSN: 0108-7673. DOI: 10.1107/s0108767307038287.

[24] K. Theisen, B. Smit, and M. Haranczyk. "Chemical Hieroglyphs: Abstract Depiction of Complex Void Space Topology of Nanoporous Materials". In: *Journal of chemical information and modeling* 50.4 (2010), pp. 461–469. DOI: 10.1021/ci900451v.

[25] A. Gandhi and M. F. Hasan. "A Graph Theoretic Representation and Analysis of Zeolite Frameworks". In: *Comput. Chem. Eng.* 155 (2021), p. 107548. ISSN: 0098-1354. DOI: 10.1016/j.compchemeng.2021.107548.

[26] D. Weininger. “SMILES, a Chemical Language and Information System. 1. Introduction to Methodology and Encoding Rules”. In: *J. Chem. Inf. Comput. Sci.* 28.1 (1988), pp. 31–36. ISSN: 0095-2338. DOI: 10.1021/ci00057a005.

[27] S. R. Heller, A. McNaught, I. Pletnev, S. Stein, and D. Tchekhovskoi. “InChI, the IUPAC International Chemical Identifier”. In: *J. Cheminformatics* 7.1 (2015), pp. 1–34. DOI: 10.1186/s13321-015-0068-4.

[28] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah. “Julia: A Fresh Approach to Numerical Computing”. In: *SIAM review* 59.1 (2017), pp. 65–98. ISSN: 0036-1445. DOI: 10.1137/141000671.

[29] J. Bezanson, J. Chen, B. Chung, S. Karpinski, V. B. Shah, J. Vitek, and L. Zoubritzky. “Julia: Dynamism and Performance Reconciled by Design”. In: *Proceedings of the ACM on Programming Languages* 2.OOPSLA (2018), pp. 1–23. ISSN: 2475-1421. DOI: 10.1145/3276490.

[30] A. P. Shevchenko and V. A. Blatov. “Simplify to Understand: How to Elucidate Crystal Structures?” In: *Struct. Chem.* 32.2 (2021), pp. 507–519. ISSN: 1572-9001. DOI: 10.1007/s11224-020-01724-4.

[31] E. Alexandrov, V. Blatov, A. Kochetkov, and D. Proserpio. “Underlying Nets in Three-Periodic Coordination Polymers: Topology, Taxonomy and Prediction from a Computer-Aided Analysis of the Cambridge Structural Database”. In: *CrystEngComm* 13.12 (2011), pp. 3947–3958. DOI: 10.1039/c0ce00636j.

[32] C. Bonneau, M. O’Keeffe, D. M. Proserpio, V. A. Blatov, S. R. Batten, S. A. Bourne, M. S. Lah, J.-G. Eon, S. T. Hyde, and S. B. Wiggin. “Deconstruction of Crystalline Networks into Underlying Nets: Relevance for Terminology Guidelines and Crystallographic Databases”. In: *Cryst. Growth Des.* 18.6 (2018), pp. 3411–3418. ISSN: 1528-7483. DOI: 10.1021/acs.cgd.8b00126.

[33] M. Li, D. Li, M. O’Keeffe, and O. M. Yaghi. “Topological Analysis of Metal–Organic Frameworks with Polytopic Linkers and/or Multiple Building Units and the Minimal Transitivity Principle”. In: *Chem. Rev.* 114.2 (2014), pp. 1343–1370. ISSN: 0009-2665. DOI: 10.1021/cr400392k.

[34] B. J. Bucior, A. S. Rosen, M. Haranczyk, Z. Yao, M. E. Ziebel, O. K. Farha, J. T. Hupp, J. I. Siepmann, A. Aspuru-Guzik, and R. Q. Snurr. “Identification Schemes for Metal–Organic Frameworks to Enable Rapid Search and Cheminformatics Analysis”. In: *Cryst. Growth Des.* 19.11 (2019), pp. 6682–6697. ISSN: 1528-7483. DOI: 10.1021/acs.cgd.9b01050.

[35] L. S. Xie, E. V. Alexandrov, G. Skorupskii, D. M. Proserpio, and M. Dincă. “Diverse π–π Stacking Motifs Modulate Electrical Conductivity in Tetrathiafulvalene-Based Metal–Organic Frameworks”. In: *Chem. Sci.* 10.37 (2019), pp. 8558–8565. DOI: 10.1039/c9sc03348c.

[36] F. M. A. Noa, M. Abrahamsson, E. Ahlberg, O. Cheung, C. R. Göb, C. J. McKenzie, and L. Öhrström. “A Unified Topology Approach to Dot-, Rod-, and Sheet-MOFs”. In: *Chem* 7.9 (2021), pp. 2491–2512. ISSN: 2451-9294. DOI: 10.1016/j.chempr.2021.07.006.

[37] A. Schoedel, M. Li, D. Li, M. O’Keeffe, and O. M. Yaghi. “Structures of Metal–Organic Frameworks with Rod Secondary Building Units”. In: *Chem. Rev.* 116.19 (2016), pp. 12466–12535. ISSN: 0009-2665. DOI: 10.1021/acs.chemrev.6b00346.

[38] S. J. Chung, T. Hahn, and W. Klee. “Nomenclature and Generation of Three-Periodic Nets: The Vector Method”. In: *Acta Cryst. A* 40.1 (1984), pp. 42–50. ISSN: 1600-5724. DOI: 10.1107/s0108767384000088.

[39] A. Ching, S. Edunov, M. Kabiljo, D. Logothetis, and S. Muthukrishnan. "One Trillion Edges: Graph Processing at Facebook-scale". In: *Proceedings of the VLDB Endowment* 8.12 (2015), pp. 1804–1815. ISSN: 2150-8097. DOI: 10.14778/2824032.2824077.

[40] T. Granlund et. al. *The GNU Multiple-Precision (GMP) Arithmetic Library*. https://gmplib.org/.

[41] W. Eberly, M. Giesbrecht, P. Giorgi, A. Storjohann, and G. Villard. "Solving Sparse Rational Linear Systems". In: *Proceedings of the 2006 International Symposium on Symbolic and Algebraic Computation*. 2006, pp. 63–70. DOI: 10.1145/1145768.1145785.

[42] W. Eberly, M. Giesbrecht, P. Giorgi, A. Storjohann, and G. Villard. "Faster Inversion and Other Black Box Matrix Computations Using Efficient Block Projections". In: *Proceedings of the 2007 International Symposium on Symbolic and Algebraic Computation*. 2007, pp. 143–150. DOI: 10.1145/1277548.1277569.

[43] J. D. Dixon. "Exact Solution of Linear Equations Using P-adic Expansions". In: *Numer. Math.* 40.1 (1982), pp. 137–141. ISSN: 0029-599X. DOI: 10.1007/bf01459082.

[44] E. Kaltofen and B. D. Saunders. "On Wiedemann's Method of Solving Sparse Linear Systems". In: *International Symposium on Applied Algebra, Algebraic Algorithms, and Error-Correcting Codes*. Springer, 1991, pp. 29–38. DOI: 10.1007/3-540-54522-0_93.

[45] O. Delgado-Friedrichs, S. T. Hyde, S.-W. Mun, M. O'Keeffe, and D. M. Proserpio. "Nets with Collisions (Unstable Nets) and Crystal Chemistry". In: *Acta Cryst. A* 69.6 (2013), pp. 535–542. ISSN: 0108-7673. DOI: 10.1107/s0108767313020655.

[46] B. Delaunay. "Neue Darstellung der geometrischen Kristallographie". In: *Z. Kristallogr. Cryst. Mater.* 84.1-6 (1933), pp. 109–149. ISSN: 2196-7105. DOI: 10.1524/zkri.1933.84.1.109.

[47] P. Niggli, W. Wien, and F. Harms. *Handbuch Der Experimentalphysik*. Leipzig: Akademische Verlagsgesellschaft, 1928.

[48] A. Santoro and A. Mighell. "Determination of Reduced Cells". In: *Acta Cryst. A* 26.1 (1970), pp. 124–127. ISSN: 0567-7394. DOI: 10.1107/s0567739470000177.

[49] A. Togo and I. Tanaka. "Spglib: A Software Library for Crystal Symmetry Search". In: *arXiv preprint arXiv:1808.01590* (2018). arXiv: 1808.01590.

[50] R. W. Grosse-Kunstleve, N. K. Sauter, N. W. Moriarty, and P. D. Adams. "The Computational Crystallography Toolbox: Crystallographic Algorithms in a Reusable Software Framework". In: *J. Appl. Cryst.* 35.1 (2002), pp. 126–136. ISSN: 0021-8898. DOI: 10.1107/s0021889801017824.

[51] G. Havas, B. S. Majewski, and K. R. Matthews. "Extended Gcd and Hermite Normal Form Algorithms via Lattice Basis Reduction". In: *Exp. Math.* 7.2 (1998), pp. 125–136. ISSN: 1058-6458. DOI: 10.1080/10586458.1998.10504362.

[52] E. Ren, P. Guilbaud, and F.-X. Coudert. "High-Throughput Computational Screening of Nanoporous Materials in Targeted Applications". In: *arXiv preprint arXiv:2202.09886* (2022). arXiv: 2202.09886.

[53] N. Castel and F.-X. Coudert. "Atomistic Models of Amorphous Metal–Organic Frameworks". In: *J. Phys. Chem. C* (2022). DOI: 10.1021/acs.jpcc.2c01091.

[54] C. Baerlocher and L. B. McCusker. *Database of Zeolite Structures*. https://europe.iza-structure.org/IZA-SC/ftc_table.php.

[55] Y. G. Chung, E. Haldoupis, B. J. Bucior, M. Haranczyk, S. Lee, H. Zhang, K. D. Vogiatzis, M. Milisavljevic, S. Ling, J. S. Camp, B. Slater, J. I. Siepmann, D. S. Sholl, and R. Q. Snurr. "Advances, Updates, and Analytics for the Computation-Ready, Experimental

Metal–Organic Framework Database: CoRE MOF 2019". In: *J. Chem. Eng. Data* 64.12 (2019), pp. 5985–5998. ISSN: 0021-9568. DOI: 10.1021/acs.jced.9b00835.

[56] C. Zheng, Y. Li, and J. Yu. "Database of Open-Framework Aluminophosphate Structures". In: *Sci. Data* 7.1 (2020). ISSN: 2052-4463. DOI: 10.1038/s41597-020-0452-4.

[57] J. V. Smith. "Topochemistry of Zeolites and Related Materials. 1. Topology and Geometry". In: *Chem. Rev.* 88.1 (1988), pp. 149–182. ISSN: 0009-2665. DOI: 10.1021/cr00083a008.

[58] M. Treacy, I. Rivin, E. Balkovsky, K. Randall, and M. Foster. "Enumeration of Periodic Tetrahedral Frameworks. II. Polynodal Graphs". In: *Micropor. Mesopor. Mater.* 74.1-3 (2004), pp. 121–132. ISSN: 1387-1811. DOI: 10.1016/j.micromeso.2004.06.013.

[59] R. Pophale, P. A. Cheeseman, and M. W. Deem. "A Database of New Zeolite-like Materials". In: *Phys. Chem. Chem. Phys.* 13.27 (2011), pp. 12407–12412. DOI: 10.1039/c0cp02255a.

[60] S. Gražulis, D. Chateigner, R. T. Downs, A. F. T. Yokochi, M. Quirós, L. Lutterotti, E. Manakova, J. Butkus, P. Moeck, and A. Le Bail. "Crystallography Open Database – an Open-Access Collection of Crystal Structures". In: *J. Appl. Cryst.* 42.4 (Aug. 2009), pp. 726–729. DOI: 10.1107/s0021889809016690.

[61] C. R. Groom, I. J. Bruno, M. P. Lightfoot, and S. C. Ward. "The Cambridge Structural Database". In: *Acta Cryst. B* 72.2 (2016), pp. 171–179. DOI: 10.1107/s2052520616003954.

[62] A. Di Lella, N. Desbiens, A. Boutin, I. Demachy, P. Ungerer, J.-P. Bellat, and A. H. Fuchs. "Molecular simulation studies of water physisorption in zeolites". In: *Physical Chemistry Chemical Physics* 8.46 (2006), pp. 5396–5406.

[63] J. Hunger, I. A. Beta, H. Böhlig, C. Ling, H. Jobic, and B. Hunger. "Adsorption structures of water in NaX studied by DRIFT spectroscopy and neutron powder diffraction". In: *The Journal of Physical Chemistry B* 110.1 (2006), pp. 342–353.

[64] S. E. Boulfelfel, J. M. Findley, H. Fang, A. S. Daou, P. I. Ravikovitch, and D. S. Sholl. "A Transferable Force Field for Predicting Adsorption and Diffusion of Small Molecules in Alkali Metal Exchanged Zeolites with Coupled Cluster Accuracy". In: *The Journal of Physical Chemistry C* 125.48 (2021), pp. 26832–26846.

[65] T. Marqueño, D. Santamaria-Perez, J. Ruiz-Fuertes, R. Chuliá-Jordán, J. L. Jordá, F. Rey, C. McGuire, A. Kavner, S. MacLeod, D. Daisenberger, C. Popescu, P. Rodriguez-Hernandez, and A. Muñoz. "An Ultrahigh CO2-Loaded Silicalite-1 Zeolite: Structural Stability and Physical Properties at High Pressures and Temperatures". In: *Inorganic Chemistry* 57.11 (2018). PMID: 29737842, pp. 6447–6455. DOI: 10.1021/acs.inorgchem.8b00523. eprint: https://doi.org/10.1021/acs.inorgchem.8b00523.

[66] D.-Y. Peng and D. B. Robinson. "A new two-constant equation of state". In: *Industrial & Engineering Chemistry Fundamentals* 15.1 (1976), pp. 59–64.

[67] O. Kunz and W. Wagner. "The GERG-2008 wide-range equation of state for natural gases and other mixtures: an expansion of GERG-2004". In: *Journal of chemical & engineering data* 57.11 (2012), pp. 3032–3091.

[68] P. J. Walker, H.-W. Yew, and A. Riedemann. "Clapeyron.jl: an extensible, open-source fluid thermodynamics toolkit". In: *Industrial & Engineering Chemistry Research* 61.20 (2022), pp. 7130–7153.

[69] K. M. Jablonka, D. Ongari, and B. Smit. "Applicability of tail corrections in the molecular simulations of porous materials". In: *Journal of chemical theory and computation* 15.10 (2019), pp. 5635–5641.

[70] H. Fang, A. Kulkarni, P. Kamakoti, R. Awati, P. I. Ravikovitch, and D. S. Sholl. "Identification of high-CO2-capacity cationic zeolites by accurate computational screening". In: *Chemistry of Materials* 28.11 (2016), pp. 3887–3896.

[71] S. H. Mousavi, K. Chen, J. Yao, A. Zavabeti, J. Z. Liu, and G. K. Li. "Screening of alkali metal-exchanged zeolites for nitrogen/methane separation". In: *Langmuir* 39.3 (2023), pp. 1277–1287.

[72] L.-C. Lin, A. H. Berger, R. L. Martin, J. Kim, J. A. Swisher, K. Jariwala, C. H. Rycroft, A. S. Bhown, M. W. Deem, M. Haranczyk, et al. "In silico screening of carbon-capture materials". In: *Nature materials* 11.7 (2012), pp. 633–641.

[73] S. Brandani, E. Mangano, and L. Sarkisov. "Net, excess and absolute adsorption and adsorption of helium". In: *Adsorption* 22 (2016), pp. 261–276.

[74] J. Rouquerol, F. Rouquerol, P. Llewellyn, G. Maurin, and K. Sing. *Adsorption by powders and porous solids: principles, methodology and applications*. Academic press, 2013.

[75] K. Sing, D. Everett, R. Haul, L. Moscou, R. Pierotti, J. Rouquerol, and T. Siemieniewska. "International union of pure and applied chemistry physical chemistry division commission on colloid and surface chemistry including catalysis. vol. 57, issue 4". In: *Pure Appl Chem* (1985), pp. 603–619.

[76] N. Ayawei, A. N. Ebelegi, and D. Wankasi. "Modelling and interpretation of adsorption isotherms". In: *Journal of chemistry* 2017.1 (2017), p. 3039817.

[77] P. Iacomi and P. L. Llewellyn. "pyGAPS: a Python-based framework for adsorption isotherm processing and material characterisation". In: *Adsorption* 25.8 (2019), pp. 1533–1542.

RÉSUMÉ EN FRANÇAIS

Introduction

RÉSUMÉ

La séparation et purification des gaz est un enjeu industriel majeur, qui nécessite l'emploi d'adsorbants spécifiques à certaines espèces moléculaires. Afin d'accélérer le développements de nouveaux adsorbants d'intérêt, il peut être utile d'avoir recours à des méthodes de simulations numériques permettant d'identifier des candidats potentiels.

Ces méthodes restent cependant coûteuses en temps et en énergie. Pour tenter d'améliorer leurs performances, cette thèse propose et explore plusieurs avancées méthodologiques sur les différentes étapes de la simulation. En particulier, de nouveaux algorithmes et outils sont présentés pour l'identification de la topologie des matériaux cristallins, pour localiser les cations dans les charpentes zéolitiques, et pour déterminer la capacité d'adsorption d'un gaz dans un matériau poreux. Les résultats de simulation alimentent une base de donnée qui est aussi exploitée pour proposer une prédiction approximative mais ultra-rapide d'isothermes d'adsorption, ouvrant la voie à des stratégies de criblage de matériaux nanoporeux à haute performance.

MOTS CLÉS

simulation moléculaire, matériaux nanoporeux, adsorption, topologie cristalline, zéolites

ABSTRACT

Gas separation and purification is a crucial industrial issue, which is technologically addressed by fine-tuning the gas specificity of adsorbents. However, the experimental development of new adsorbents is a costly process, hence the need for preliminary computer simulations to test only the most promising candidates.

These simulation techniques remain costly in themselves. In order to improve their performance, this thesis proposes and explores new methodological options across the different steps of the simulation. In particular, new algorithms and tools are presented to tackle the identification of the topology of crystalline materials, the location of cations in zeolitic frameworks, and the adsorption capacity of a gas in porous materials. The simulation results are then used to feed a database, which allows using a statistical approach to provide coarse but ultra-fast predictions of adsorption isotherms, paving the way for high-performance screening stratgies of nanoporous materials.

KEYWORDS

molecular simulation, nanoporous materials, adsorption, crystal topology, zeolites