XtalOpt Version 13: Multi-Objective Evolutionary Search for Novel Functional Materials

Samad Hajinazar[a], Eva Zurek[a,*]

[a]*Department of Chemistry, State University of New York at Buffalo, Buffalo, New York 14260-3000, United States*

Abstract

Version 13 of XtalOpt, an evolutionary algorithm for crystal structure prediction, is now available for download from the CPC program library or the XtalOpt website, https://xtalopt.github.io. In the new version of the XtalOpt code, a general platform for multi-objective global optimization is implemented. This functionality is designed to facilitate the search for (meta)stable phases of functional materials through minimization of the enthalpy of a crystalline system coupled with the simultaneous optimization of any desired properties that are specified by the user. The code is also able to perform a constrained search by filtering the parent pool of structures based on a user-specified feature, while optimizing multiple objectives. Here, we present the implementation and various technical details, and we provide a brief overview of additional improvements that have been introduced in the new version of XtalOpt.

Keywords: Evolutionary structure prediction, Multi-objective global optimization, Prediction of functional materials.

NEW VERSION PROGRAM SUMMARY

Program Title: XtalOpt
CPC Library link to program files: (to be added by Technical Editor)
Developer's repository link: https://github.com/xtalopt/XtalOpt
Code Ocean capsule: (to be added by Technical Editor)
Licensing provisions: 3-clause/BSD.
Programming language: C++.
Journal reference of previous version: Computer Physics Communications 237 (2019) 274–275.
Does the new version supersede the previous version?: Yes.
*Reasons for the new version:** Implementation of a multi-objective evolutionary search within the XtalOpt program package.
Summary of revisions: Implemented a general user-friendly multi-objective search capability, made various improvements to user interface and functionalities, performed bug fixes.
Nature of problem: The XtalOpt algorithm is designed to search for (meta)stable crystal structures, optionally with specific functionalities – a grand challenge in computational materials science, chemistry and physics.
Solution method: A generalized scalar fitness function, where a set of user-specified objectives contribute to the fitness value for candidate structures, is implemented within XtalOpt. This generalized fitness biases the search towards the discovery of (meta)stable phases with structural motifs that are key for the desired characteristics. As a result, the evolutionary search explores regions of the energy landscape of higher relevance in terms of target properties.

1. Introduction

The computational prediction of novel materials has recently come to the fore. Prolific examples include GNoME's purported discovery of over 2 million crystalline compounds whose energies fall below the currently-known convex hulls [1], and the machine-accelerated high-throughput identification of ambient-pressure superconductors [2, 3]. In this manuscript we describe a general method, an example of multi-objective optimization, that can be used to predict novel materials with specific functionalities. This method, implemented within the XtalOpt program package, can be paired with any external optimizer of crystalline lattices, along with any program that can estimate a property of a given material or output a descriptor that serves as a proxy for the desired functionality. Therefore, this new methodology can be used for the discovery of functional materials, but, notably, it can target any property that can be computed through the user's choice of theoretical approach including the predictive models such as

*Corresponding author.
E-mail address: ezurek@buffalo.edu

those generated in the above examples [1, 2, 3] .

In the last two decades, global optimization (GO) algorithms, typically interfaced with density functional theory (DFT) optimizers, have become an integral part of the materials-scientist's toolbox to find approximate solutions for the global minimum in the potential energy surface (PES). The basic techniques include random search [4], simulated annealing [5], metadynamics [6], minima [7] and basin [8] hopping, particle swarm optimization [9], and evolutionary algorithms [10, 11, 12, 13]. Until recently [14, 15], these GO strategies have been used to minimize the 0 K energy or enthalpy of a chemical system, and finite temperature terms have been ignored due to their computational expense. While some of these algorithms are more suited to a global exploration of the PES, others sample local regions and are therefore more likely to find metastable structures. Numerous manuscripts have reviewed these methods, their successes, failures, and more [16, 17, 18, 19, 20].

Despite the role that GO algorithms have played in aiding the characterization of synthesized materials and predicting novel materials for synthesis [21], their focus on finding the minimum in the PES can be a limitation. Indeed, most of the organic compounds that we know are metastable, and in the solid state various synthesis techniques for accessing metastable phases are actively being developed [22]. The intense interest in metastable compounds stems from their potential uses in a wide variety of technological applications including as superconductors, superhard or refractory materials, thermoelectrics, photovoltaics, and multiferroics. Simply put: GO algorithms may fail to predict a phase with desirable characteristics because it is metastable.

One way that researchers have worked around this limitation is to perform GO searches that locally optimized each structure, but the fitness was calculated using the property of interest rather than the enthalpy. This strategy has been adopted for the design and prediction of phases with desired electronic structures [23, 24], as well as super-dense [25], super-hard [26] and high dielectric [27, 28] materials. Other techniques that have been suggested include encoding the desired properties into the structural information and evolving them during the search process [29], or removing structures with undesirable geometric characteristics from the breeding pool by artificially assigning them a high enthalpy [30].

A more natural integration of the desired properties in the GO process can be achieved by simultaneous optimization of all target objectives. This can be addressed using the family of "multi-objective" global optimization (MOGO) algorithms [31, 32]. Since it is unlikely that the optimal solutions for two (or more) objectives are the same, typical MOGO algorithms aim to find a set of solutions that offer the best trade-off between various objectives. A common approach, the family of Pareto-based algorithms, searches for a set of solution candidates (known as Pareto optimal solutions) such that for any solution no single objective can be improved without worsening the others. Another common technique, belonging to the category of decomposition-based methods, is the scalarizing approach wherein a scalar fitness measure is employed to represent the optimality of the candidate solutions. This approach, effectively, transforms the MOGO problem into a single-objective problem tractable by traditional GO algorithms.

Various MOGO algorithms have been utilized for materials discovery [33]. Most notable are several studies dedicated to the prediction of new functional materials through implementations of Bayesian global optimization [34, 35, 36], and evolutionary and differential evolution algorithms [37, 38, 39, 40, 41, 42, 43]. The focus of this manuscript is on the family of evolutionary search (ES) algorithms, which are iterative stochastic approaches to GO. Among the MOGO approaches, in particular, the multi-objective evolutionary search (MOES) algorithms [44, 45, 46] are popular because they are inherently population-based and straightforward to implement.

The previous version of the XtalOpt code [47, 16] included an implementation of the MOES algorithm to search for (meta)stable phases of hard materials. In this scheme, the Vickers hardness, obtained from the AFLOW machine learning (ML) estimated shear modulus [48], was used in combination with the enthalpy to evaluate the fitness of candidate structures. In addition to finding numerous superhard carbon polymorphs, a superhard-superconducting phase was found [49].

In this work, we present a general implementation of the XtalOpt MOES algorithm. The new implementation is designed to (i) accommodate an arbitrary number of target objectives, (ii) allow the user to introduce any desired objective as long as it can be represented with a numerical value, and (iii) offer a general and easy-to-setup interface that can be used with any external code. Further, the XtalOpt MOES can perform various types of optimizations (minimization or maximization) depending on the target objective and can facilitate a constrained search by filtering the pool of structures

based on user-specified criteria. This functionality is available in both the command-line interface (CLI) and the graphical user interface (GUI) modes of XtalOpt.

This article describes the MOES functionality recently implemented in XtalOpt. Section 2 introduces the generalized fitness function: it describes the main input parameters of the MOES and details the properties of the required external codes. Sections 3 and 4 illustrate how to set up a search in the CLI and GUI versions of XtalOpt, respectively, while Section 5 provides examples of scripts that can calculate objectives for either version. The constrained search functionality is introduced in Section 6, and Section 7 describes the changes in the (legacy) hardness optimization. The MOES output and general error handling is outlined in Section 8. Finally, Section 9 lists a series of options and functionalities, independent of the MOES, that have been added to the latest version of the XtalOpt code.

2. Multi-objective search

2.1. Generalized fitness function

A typical ES [17] workflow begins with (i) generating a population of candidate structures, (ii) locally relaxing these structures, and (iii) assigning a fitness to a structure based on its enthalpy. From the pool of possible parents (iv) a structure is chosen randomly, but with a probability related to its fitness, to (v) generate a new child structure via applying variations of bio-inspired genetic operators, which include single parent distortions (mutations) or two-parent cut-and-splice (breeding). Steps (ii)-(v) are repeated until a pre-defined stopping criterion is satisfied.

In a single-objective ES, such as the initial implementation of XtalOpt, the fitness, f_s, is assigned to each candidate structure s via:

$$f_s = \frac{H_{\max} - H_s}{H_{\max} - H_{\min}} \tag{1}$$

where H_s is the enthalpy of structure s, and $H_{\max}$ and $H_{\min}$ are the maximum and minimum enthalpies, respectively, of the relaxed structures. The calculated fitness values fall between 0 and 1.0, and the (user defined number of) structures with the highest fitness are selected to be part of the breeding pool. The fitness of the structures comprising the breeding pool is first normalized so its sum is equal to unity, then employed to determine a probability for each structure. The breeding pool candidates are sorted according to the probabilities, normalized such that the lowest enthalpy crystal is assigned a probability of 1.0. A structure is chosen to be a parent if its probability is above that of a random number that falls between 0 and 1. This procedure gives a relatively higher chance for procreation to those candidates that are more "fit", i.e., have a lower enthalpy.

The MOES, however, needs a different measure for evaluating the suitability of candidate structures to be parents for the next generation. Since an ES is typically followed by high-accuracy local optimization of the best candidates, we found it sufficient to resort to the scalarizing technique by using a generalized scalar function that merges all objectives to obtain a single measure of fitness for each structure [50]. Assuming that $\{X\}$ and $\{Y\}$ represent sets of objectives to be minimized and maximized, respectively, the generalized fitness for the s^{th} structure can be obtained as:

$$f_s = \sum_i w_X^i \left(\frac{X_{max}^i - X_s^i}{X_{max}^i - X_{min}^i} \right) + \sum_j w_Y^j \left(\frac{Y_s^j - Y_{min}^j}{Y_{max}^j - Y_{min}^j} \right), \tag{2}$$

where $\{X_s\}$ and $\{Y_s\}$ are the values of the corresponding objectives calculated for the s^{th} structure, and $\{w\}$ is the set of weights associated with the objectives chosen to reflect their relative significance.

Given a total weight of 1.0 for all objectives, the above fitness measure will be normalized to $[0, 1.0]$, and the MOES is converted to a single-objective search for which the standard ES workflow can be followed. Figure 1 illustrates schematically the workflow of the MOES implementation within XtalOpt. Following a local optimization, the structural coordinates are used to calculate the desired objectives, whose values are employed to obtain the generalized fitness function for choosing the next parent(s). Otherwise, the workflow resembles that of a single-objective ES.

In practice, enthalpy is always set to be one of the objectives to be minimized. While this is not necessary in our implementation, inclusion of enthalpy enables the search for low lying local minima that potentially feature the target properties, though they may not have the lowest enthalpy. This procedure effectively increases the chance of finding "metastable" phases with the desired properties. Relying on the target property alone may result in the identification of phases with too-high-enthalpies that can never be synthesized, or those located in extremely shallow potential wells whose barriers can be easily overcome at ambient temperatures.

For utilizing the XtalOpt MOES, the user must specify

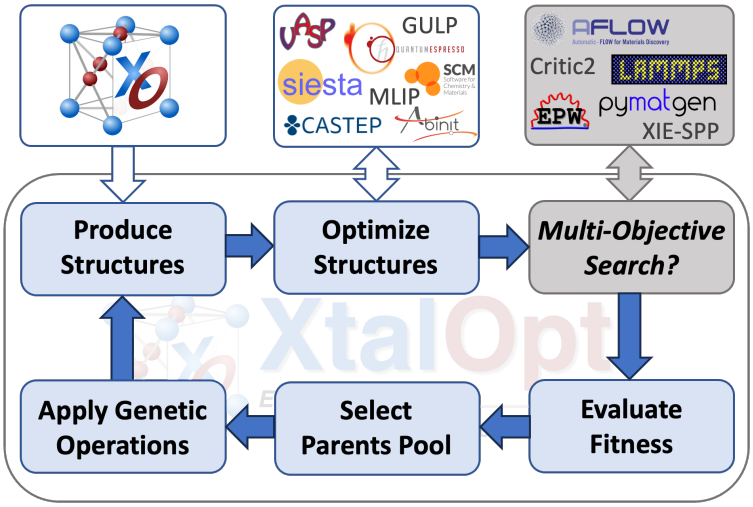

Figure 1: Workflow of the XtalOpt MOES (Multi-Objective Evolutionary Search) algorithm. After producing the initial set of structures, local relaxations can be performed via various first-principles approaches or by using interatomic potentials. The locally optimized structures are then passed on to external code(s) introduced by the user for calculating desired properties (a few are shown as examples). Subsequently, the fitness function is evaluated for all structures and the parent pool is selected accordingly. New structures are generated by applying various evolutionary operations to the structures chosen from the parent pool.

a (potentially external) code to compute the target property for each objective considered in the search, and provide the corresponding weight for the fitness calculation (Equation 2), as described in the next section.

2.2. *User-defined objectives*

For any property employed as an objective in the MOES, there should be an (executable) user-provided script or code that (i) calculates that property from the coordinates of a structure generated during the course of a regular XtalOpt search, and (ii) produces an output file with a single number (integer or double) as the value of the desired objective. The script should be available either in the local path (for a local run) or on the cluster access path (for a remote queue). XtalOpt will call this script automatically and will read and use its output for the MOES; either for determining a structure's fitness for procreation or for filtering the parent pool (see Section 6).

After each local relaxation is performed using any of the total energy calculation methods available in XtalOpt, a structure file is generated. This file is named `output.POSCAR`, irrespective of the external optimizer, and it is written in the format used by the Vienna Ab initio Simulation Package (VASP) [51, 52]. The user-provided script should be able to read and employ this file (or convert it to another structural data format if needed) to perform the intended calculations and produce the required output file, which should be a text file with the numerical value of the corresponding objective written as the first entry of the first line of the file.

The aforementioned script can simply contain a sequence of commands that use the `output.POSCAR` file and call some program to produce a result. However, if the intended calculations are computationally demanding, it may be preferable if the script generates a cluster job file and submits the job to the computational cluster. If input data not present in the output of the structural relaxation is required by the script (e.g., a file containing the *ab initio* charge density, the density of states at the Fermi level, etc.), commands to generate this data should be present in the script. Alternatively, the required values or files can be obtained by adding the appropriate entries to the XtalOpt job template used in the structure search. Samples scripts are provided in Section 5.

2.3. *Multi-objective search parameters*

To invoke the MOES functionality the user should provide the following information to the XtalOpt code for every desired objective:

- **optimization type**: instructions on how to use the value of a calculated objective in determining a structure's fitness. XtalOpt can minimize or maximize an objective, filter the parent pool according to user-defined criteria (see Section 6), or maximize the AFLOW-ML hardness (see Section 7).

- **path to the user-defined script**: the full path to the script that retrieves or calculates the desired property corresponding to the introduced objective. The script is automatically run by the XtalOpt code for each locally relaxed structure.
- **script's output filename**: the name of the file generated by the script that contains the calculated result of the corresponding objective.
- **optimization weight**: a number between 0.0 and 1.0 used as the weight for the corresponding objective in the fitness calculation. The total weight of all objectives should not exceed 1.0, and the weight for minimizing the enthalpy is calculated by XtalOpt as: "*1.0 - total weight of the objectives*".

Any objective explicitly assigned a weight of 0.0 will be calculated, but will not affect the optimization. Moreover, if the sum of the weights explicitly assigned equals 1.0, the enthalpy is assigned a weight of 0.0, i.e., the fitness is determined only based on the values calculated for the other objectives. If the total weight of the introduced objectives exceeds 1.0, XtalOpt will quit after producing an error message.

3. Multi-objective run in the XtalOpt CLI

In the MOES run in the CLI mode, for each user-defined objective a line should be added to the XtalOpt input file that starts with the keyword *objective*. This line includes the above-mentioned information for the objective and, generally, has the following format:

```
objective = "optimization_type"
    "/path/script" "script_output_filename"
    "weight"
```

It should be noted that in the CLI mode of XtalOpt:

1. Possible options for the "**optimization_type**" are "minimization", "maximization", "hardness", and "filtration", as introduced above. This field is not case-sensitive, and only the first three letters are important in identifying the optimization type by XtalOpt (i.e., "min", "max", "har", and "fil").
2. Providing the "**script_output_filename**" is *optional*. If this is not specified, the default will be `objective#.out` in which "#" is the number of the objectives in the order that they appear in the XtalOpt input file (excluding the "hardness" objective), e.g., `objective1.out`, `objective2.out`, etc.
3. Specifying the "**weight**" for the objective is *optional*, as well. If this field is not given for a number of objectives, it will be calculated. Specifically, if any weight is provided for any of the objectives, it will be subtracted from 1.0 and the remaining value will be divided between the enthalpy and objectives that don't have a specified weight.

Let us now provide some examples of acceptable input files. For a calculation that aims to minimize the volume per atom and maximize the electronic band-gap, the following lines can be added to the XtalOpt input file:

```
objective = min  /path/vol.sh  vol.dat  0.2
objective = max  /path/gap.sh  gap.dat  0.2
```

In this case, `vol.sh` and `gap.sh` are two executable scripts (either in a local or remote location, depending on the run) that use the `output.POSCAR` file to calculate the volume per atom and the band-gap, respectively. XtalOpt expects the calculated values to be written by these scripts to the `vol.dat` and `gap.dat` files in the structure's directory. Since the weight for both objectives is 0.2, the remaining weight of 0.6 will be assigned to the enthalpy.

The following example illustrates the flexibility of the input entries:

```
objective = min  /path/vol.sh  vol.dat
objective = max  /path/gap.sh  0.2
```

Since the weight for the volume objective is not specified, the remaining total weight of 0.8 will be divided equally between enthalpy and volume per atom, and since the output filename for the band-gap calculation is not given, XtalOpt will expect a file named `objective2.out` (as this is the second objective in the list of objectives) to be present with the corresponding value.

4. Multi-objective runs in the XtalOpt GUI

We now describe how the XtalOpt GUI has been modified to reflect the new options that are available in the MOES run. In the **Search Settings** tab the AFLOW-ML hardness related entries have been removed (green box in Figure 2(a)), and a new **Multiobjective Search** tab is introduced (red outline). In the **Multiobjective Search** tab (Figure 2(b)), all entries relevant to a MOES run (described in Section 2.3) can be entered in the correspond-

ing fields by (1) choosing the MOES run type from a drop-down menu, (2) specifying the weight associated with a particular objective, entering the (3) name and full path to the user-provided script and (4) output file names, and (5) specifying how XtalOpt should handle a structure that fails a filtration objective (as discussed in Section 6). It should be noted that any weight left as zero in the GUI input will result in the corresponding objective being calculated without affecting the optimization. Finally, clicking the button "Add objective" adds the defined objective to the list of objectives for the run.

The aforementioned MOES run specifications in the CLI mode apply to the GUI case, as well:

- The types of MOES runs are "maximization", "minimization", "filtration", and "hardness"; and setting up an AFLOW-ML hardness calculation only requires its weight to be specified (Figure 2(c)),
- No more than one instance of "hardness" objective is considered in each run; if more than one is present, only the corresponding weight (if entered differently) will be updated,
- Desired objectives can be arbitrarily added or removed before the run is started. However, once the search begins, the only MOES-related parameter that can be altered is the one instructing the code how to handle the structures discarded by a "filtration" objective (as detailed in Section 6).

Further, in the XtalOpt GUI MOES implementation,

- Common errors in the input parameters (e.g., leaving script or file name fields empty, spaces in text entries, total weight exceeding 1.0), result in an error message from the code (Figure 2(d)),
- In the **Progress** tab, the status of a structure that has successfully finished local relaxations changes to "Calculating objectives...", after which the status changes to "Optimized", "ObjectiveDismiss", or "ObjectiveFail" according to the calculation results (Figure 2(e)),
- In the **Plot** tab, once a MOES run is started, the list of introduced objectives appears among the available options for the x and y axes of the trend plots, as well as the list of labeling symbols (Figure 2(f)).

5. Examples of user-defined scripts

In this section we provide an example of a MOES-compatible script that can be employed with both the CLI and GUI versions. We choose a simple example, wherein the goal is to minimize the enthalpy, while simultaneously maximizing a structure's space group number calculated from the VASP format `output.POSCAR` structure file. This can be done via a simple Python script, e.g., `/path/spg.py`, where the Atomic Simulation Environment (ASE) [53] is utilized to resolve the space group of the structure as,

```
import ase.io as io
from ase import Atoms
from ase.spacegroup import get_spacegroup
s=io.read('output.POSCAR', index ='-1',
    format='vasp')
print(get_spacegroup(s, symprec=1e-3).no)
```

The output of this short Python code can be used via a simple executable bash script, e.g., `/path/spg.sh`,

```
#!/bin/bash
/path/python /path/spg.py > spg.dat
```

to produce a `spg.dat` file containing the space group number of the structure, which is readable by XtalOpt for this objective. In the XtalOpt input file for the CLI mode, this can be then introduced as:

```
objective = max  /path/spg.sh  spg.dat
```

along with the desired weight (or leaving the weight unspecified for XtalOpt to adjust it).

Alternatively, if the user desires to submit this calculation to a computational cluster, the executable script `/path/spg_queue.sh` in its most basic form can be written as:

```
#!/bin/bash

cat > fspg.slurm << EOF
#!/bin/bash
#SBATCH --nodes=1 --ntasks-per-node=1
#SBATCH --job-name=fspg
#SBATCH --output=fspg.out --error=fspg.err
#SBATCH --time=00:05:00
#SBATCH --cluster=slurm

##===== main task: calculating the objective
/path/python /paht/spg.py > spg.dat
##=====
```

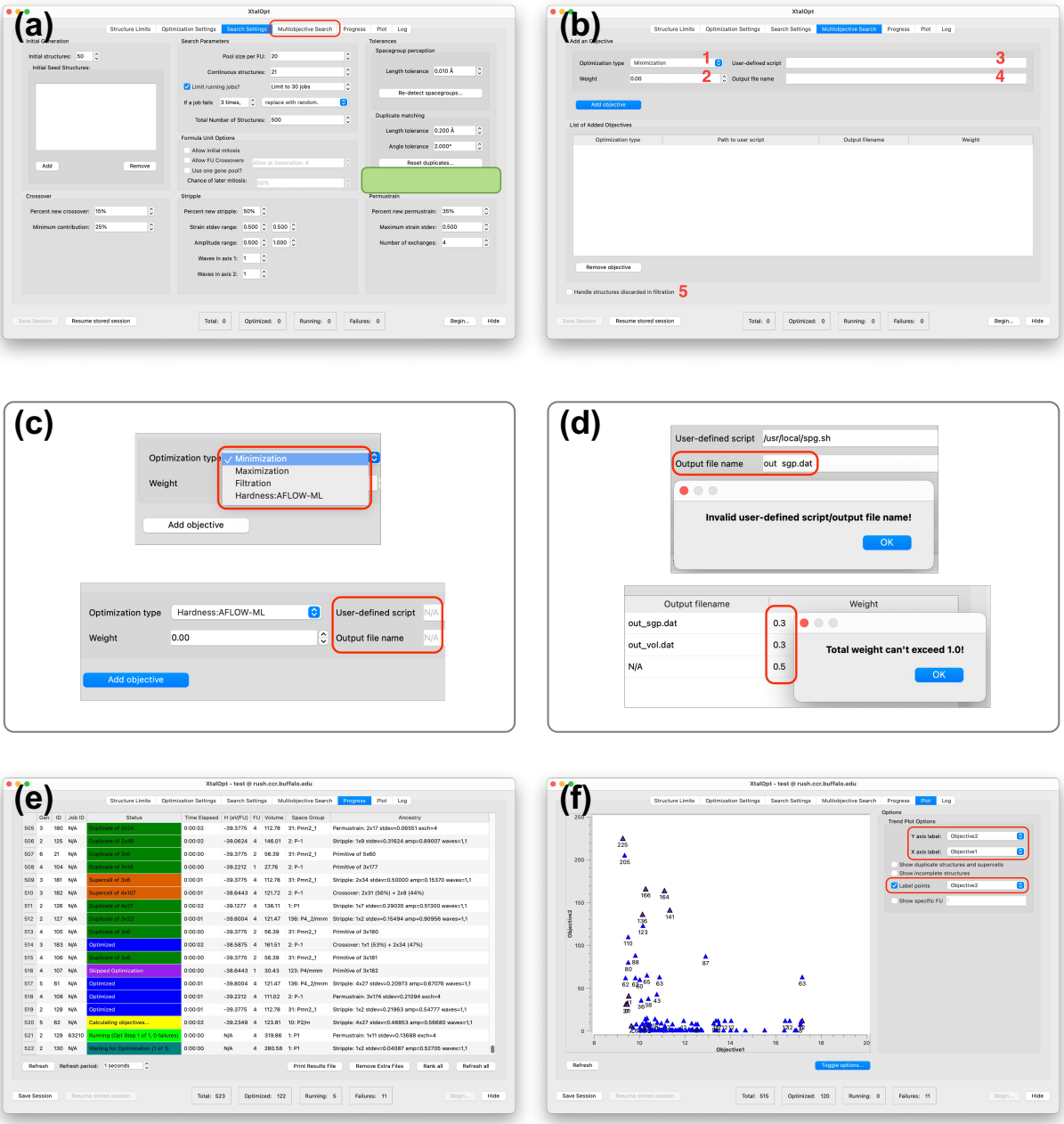

Figure 2: Screenshots from a MOES run in the XtalOpt GUI mode. (a) The **Search Settings** tab no longer includes AFLOW-ML hardness entries (green oval), and a new **Multiobjective Search** tab (b) is added. (c) Drop-down menu showing the multi-objective search type options, including for an AFLOW-ML hardness calculation. (d) Pop-ups showing errors that arise from non-acceptable text entries, and entering weights that sum to a value larger than one. (e) The **Progress** tab with new MOES-related status. (f) The **Plot** tab in a MOES run with user-defined objectives as labels and tags.

```
EOF

sbatch fspg.slurm
```

This particular executable script writes a job submission script for the slurm cluster to disk (`fspg.slurm` file, which includes everything between the lines containing the "EOF" keywords) and then submits this job (with "`sbatch fspg.slurm`") to the cluster. The job submission script (`fspg.slurm` file) includes an introductory part (job- and cluster-related settings) and the core task, just like a usual job submission script. It contains the previous simple script for calculating the space group number (which is enclosed between "`##=====`" comment lines for clarity). The latter script can be employed in conjunction with the MOES to optimize the space group number just as in the previous example, only, this time each calculation is submitted to the cluster instead of running through a simple executable bash script.

While the examples provided in Section 3 and Section 5 illustrate the structure of the input file that must be present when the CLI version is employed and the files required to calculate the property of interest, the objectives chosen are somewhat artificial. Objectives such as the calculated density of states at the Fermi level, band-gap, Vickers or Knoops Hardness, superconducting critical temperature, zT figure of merit and more could be chosen, as desired by the user. A follow up publication will focus on application of the MOES-XtalOpt methodology for identifying structures with desired properties.

6. Filtering structures: Constrained search

The MOES implementation within the XtalOpt code can also facilitate a *constrained ES*. Besides maximizing or minimizing a particular objective, XtalOpt can optionally filter the relaxed structures based on user defined properties. The constrained search prevents crystals deemed unsuitable from entering the parent pool, hence promoting or prohibiting the propagation of a specific genetic characteristic. A similar approach was employed in Ref. [30], except that structures were excluded from the parent pool by artificially setting their enthalpy to an unphysically high value. The filtration technique introduced here has a similar effect, but without the need of modifying the enthalpy.

To utilize the filtration functionality, similar to the "minimization" and "maximization" features, the user should provide a script that marks the structures to keep or discard based on the intended property. Structures that are marked for discarding will not be allowed in the parents' pool, although they will remain in the set of generated structures. An example of the relevant entry in the XtalOpt input file for the CLI mode is:

```
objective = fil  /path/script  out_file
```

The workflow differs compared to when an objective is minimized or maximized by XtalOpt in the following ways. Since the objective is not meant for optimization, regardless of the user-defined weight, the objective's weight will be automatically set to 0.0. Moreover, the numerical value written to the output file by the script should be either 0 (instructing XtalOpt to discard the structure) or 1 (instructing XtalOpt to keep it). A structure that fails the filtration step, by default, will be removed from the parents' pool. However, the user can optionally instruct XtalOpt for further handling of a dismissed structure by adding the text

```
objectivesReDo = true       # default is false
```

to the input file in the CLI mode. In this case, XtalOpt proceeds depending on the value of the *jobFailAction* flag in the input file. If the *jobFailAction* is set to "replaceWithRandom" (default value) or "replaceWithOffspring" the failed structure is replaced with a new structure generated randomly or by applying evolutionary operations, respectively. On the other hand, if *jobFailAction* is set to "kill" or "keepTrying", no further action is taken. If the user instructions result in replacing the failed structure with a new one, it will then be submitted for local optimization and the subsequent calculation of objectives, including the filtration objective. It should be noted that this procedure will be performed at most once for a failed structure, i.e., no more than one replacement will be attempted for a structure that is marked to be dismissed in filtration.

The same can be achieved in the GUI mode by checking the "Handle structures discarded in filtration" box in the **Multiobjective Search** tab; where the appropriate follow-up action will be taken according to the "If a job fails" entry in the **Search Settings** tab and similar to the above workflow discussed for the CLI case.

7. AFLOW-ML hardness

Previously, maximizing the AFLOW-ML hardness was invoked with the *calculateHardness* flag in the CLI

XtalOpt version (or, setting relevant entries in the GUI). In the current version the Vickers hardness is obtained by introducing the "hardness" objective. In the CLI mode, this can be performed by adding an objective with the "hardness" optimization type and, optionally, providing a corresponding weight, i.e.,

```
objective = hardness "weight"
```

Therefore, AFLOW-ML hardness calculations are now treated as a user-defined objective. It should be noted that the script name and the output file name inputs do not need to be provided for a "hardness" objective, and while an arbitrary number of objectives with "maximization", "minimization", and "filtration" type can be introduced, no more than one "hardness" objective can be added by the user. If there is more than one entry for hardness calculations in the XtalOpt input file, only the last one will be considered by the code (i.e., the weight for hardness calculation will be that of the last entry).

An objective of the "hardness" type performs the hardness optimization by obtaining the relevant data from AFLOW-ML through internal functions of the XtalOpt code. This is a legacy code and will be disabled in future releases of XtalOpt to avoid compatibility issues. Instead of using this type of objective, a script to facilitate AFLOW-ML hardness optimization, similar to any regular "maximization" objective, is strongly encouraged. In this case, an example of a script that can be used to retrieve the required information from the AFLOW-ML platform is presented in Appendix A.

8. Error handling and outputs

The external script (or the code it is called by) may fail to operate properly or fail to produce the output in a format that is readable by XtalOpt. In general, XtalOpt considers the output file to be correct and contain a valid value only if the "first entry" in the "first line" of the file is a numerical value. Otherwise, e.g., the file is not produced, is empty, or its first entry is not a legitimate numerical value, the calculation will be marked as failed, and the structure will not be considered as a candidate to enter the breeding pool.

Generally, the search settings (including the MOES settings) are written to the `xtaloptSettings.log` and `xtalopt.state` files, which can be used to verify the initialization of the search. In the CLI mode the `xtalopt-runtime-options.txt` file, which includes the parameters that can be modified during the search, is also produced. Among the MOES-related entries only the *objectivesReDo* flag is output to this file, and is allowed to be changed once the run is started.

For each structure, besides the corresponding output files generated by the scripts, a summary of the objective-related info (overall status of its calculations and their value) is written to the `structure.state` file.

Finally, just as with any XtalOpt run, the live status is available in the `results.txt` file, which summarizes the ranking of structures generated during the course of the ES. In the case of MOES runs, the status of a structure that has successfully been locally optimized changes to "ObjectiveCalculation". Once the objectives are obtained, the status changes to "Optimized", "ObjectiveDismiss", or "ObjectiveFail" depending on whether the calculations finished successfully, the structure was discarded during filtration, or the calculations failed, respectively. Moreover, the `results.txt` file contains an extra column for the calculated values of each objective introduced by the user.

9. Miscellaneous

In the new release of the XtalOpt code, a number of options are implemented to address special situations the user might encounter. These options are briefly introduced in the following.

9.1. Scaled volume limits (CLI and GUI mode)

An important step in conducting a successful ES is the specification of reasonable minimum and maximum volumes for the unit cells. The previous versions of XtalOpt allowed the user to either explicitly specify these limits, or to introduce a fixed value for the generated unit cells' volume. Now, a new option in XtalOpt can facilitate a reasonable guess.

In the CLI mode, the user can optionally specify the pair of flags:

```
volumeScaleMin = ####
volumeScaleMax = ####
```

with the corresponding values being real numbers greater than zero (e.g., 0.8 and 1.2 for the minimum and maximum values, respectively). If these flags are properly provided, XtalOpt first calculates the total volume of spheres of van der Waals radii for all atoms in the formula unit. Then, it multiplies that total volume by the scaling factors to obtain the minimum and maximum volumes. One can check the final calculated

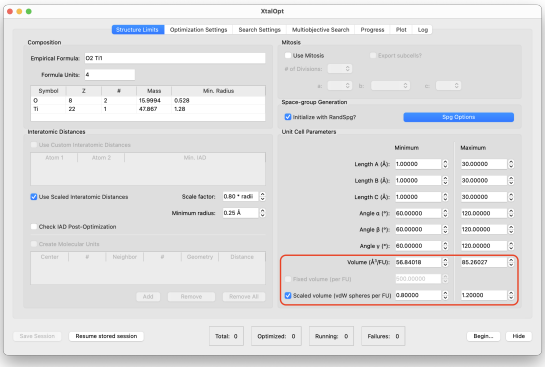

Figure 3: Screenshot of the "Structure Limits" tab in the new XtalOpt GUI. The red circle highlights the settings used calculated the acceptable volume range from the sum of van der Waals spheres multiplied by a scaling factor.

values in the run output (i.e., the `xtalopt.state` or `xtalopt-runtime-options.txt` files).

The GUI mode includes this option in the **Structure Limits** tab, where the user can set the scaling factors and access a live update of calculated minimum and maximum volume limits while adjusting the scaling factors (Figure 3).

9.2. Running XtalOpt *locally on a cluster (CLI mode)*

Often when lengthy ES runs are performed, the XtalOpt code is executed on the cluster where the jobs are being submitted (i.e., running XtalOpt locally while submitting the jobs to a queue). As the jobs are being submitted to the cluster, a remote queue interface should be specified in the XtalOpt input (e.g., slurm, pbs, etc.). This, however, requires a ssh connection to the cluster itself, which might or might not be allowed, depending on the ssh configuration of the user's account. For these types of XtalOpt runs, in the CLI mode, the user can add the following flag to the input file and run the code as a regular remote run:

```
localQueue = true      # default is false
```

9.3. Termination of the XtalOpt *run (CLI mode)*

In a regular XtalOpt run, and once the maximum number of structures (specified by the user) is generated, the user can resume the run by increasing this maximum number (among other run-time flags that can be changed). This functionality requires the code to not quit automatically, and the user needs to terminate the application manually after the desired output is obtained. There are, however, situations where it is desirable for the code to exit after producing a specified number of structures (e.g., scripting a series of XtalOpt runs over multiple directories, such that each run should begin after the previous one is finished). For a run in the CLI mode, the code can be instructed to exit after producing the specified number of structures by adding the following flag to the input file:

```
softExit = true       # default is false
```

or by setting its value to true in the run-time setting file `xtalopt-runtime-options.txt` during the run. With this flag set to true, the code quits after all running (and pending) jobs are finished and the output files are updated.

On the other hand, and at any moment during a run, the user can force the XtalOpt process (hence, the run) to quit immediately by adding the following line to the run-time settings file (`xtalopt-runtime-options.txt`),

```
hardExit = true
```

It should be noted that the *hardExit* flag terminates the XtalOpt running process regardless of any running or pending jobs and without updating the output files, and this is only a run-time option and the presence of the flag in the input file is ignored by XtalOpt.

9.4. New options for the VASP optimizer (CLI and GUI modes)

Recent versions of the VASP code allow for training and using ML interatomic potentials. The new version of XtalOpt supports the output (i.e., `OUTCAR` file) generated by structural optimizations performed using VASP ML force fields.

In previous releases of XtalOpt, when the VASP optimizer was used, a `POTCAR` file was required for each element type present in the system. In the new version, it is possible to provide only a single `POTCAR` file for a multi-element system. This option is especially useful when XtalOpt is interfaced with an external code that is not explicitly supported (e.g., an arbitrary optimizer, which is scripted to produce VASP format output files).

In the GUI, introducing a single `POTCAR` file for the system can be accomplished by setting the path as:

```
%fileContents:/path/to/system/potcar%
```

and in the CLI mode this can be done by introducing a POTCAR file of the "system" type in the input file, i.e.,

```
potcarFile system = /path_to/potcar
```

It should be noted that as XtalOpt arranges the chemical elements in alphabetical order, individual POTCAR files should be combined in the same order to produce the correct results. Moreover, if a "system" POTCAR is introduced, other entries of *potcarFile* flag in the XtalOpt input file will be ignored by the code.

9.5. *Run-time log file and debug options (CLI and GUI modes)*

In the previous versions of XtalOpt, the code output a comprehensive list of messages regarding the progress of the run in the CLI mode, while only a subset of this information (i.e., key updates about the status of the run) were available in the **Log** tab for a run in the GUI mode. In the new XtalOpt release, compilation with the

```
-DXTALOPT_DEBUG=ON
```

configuration option saves a detailed list of output messages to the log file `xtaloptDebug.log` in the local working directory for both the CLI and GUI modes.

Further, if the code is compiled with the

```
-DMOES_DEBUG=ON
```

configuration option, running a MOES produces a set of output messages regarding the calculation of objectives and the generalized fitness function. These lines, starting with the keyword NOTE, contain information useful in monitoring the sanity of the calculations and MOES run.

10. Conclusions

Herein, we describe developments to the latest version of the XtalOpt evolutionary algorithm that make it possible to perform a multi-objective global optimization (MOGO) search for materials with a desired functionality. The resulting multi-objective evolutionary search (MOES) employs a weighted linear sum of functions for each introduced objective, and it belongs to the category of decomposition-based MOGO techniques. This makes it possible for the user to (optionally) minimize the energy or enthalpy of a crystalline lattice, while at the same time optimizing (minimizing or maximizing) any arbitrary feature or objective for which a numerical value can be obtained, optionally by a code other than the one employed for structural relaxation. A further option to filter structures with undesirable characteristics from the breeding pool is also implemented.

The MOES implementations in both the graphical user interface (GUI) and command line interface (CLI) versions are described, and example input files for the latter are provided. Examples of user defined scripts, which can be used to call the external codes either locally or using a job submission script that is sent to the computational cluster, are also given. Finally, a number of miscellaneous fixes, regarding options for: choosing the unit cell volumes, CLI-mode initialization and termination, run-time and log file generation, and for the VASP optimizer are described.

The MOES implemented in XtalOpt will be useful for the computational discovery of novel materials with a wide range of functionalities. Work is underway in developing descriptors, workflows and user-defined scripts for the prediction of superconductors, electrides, crystalline lattices with user-defined geometric features and more.

Acknowledgement

We acknowledge the U.S. National Science Foundation (DMR-2136038 and DMR-2119065) for financial support, and the Center for Computational Research at SUNY Buffalo for computational support (http://hdl.handle.net/10477/79221).

Appendix A. Retrieving AFLOW-ML data

The data within the AFLOW database has been used to train a number of ML models, for example for obtaining structure-dependent band-gaps (or metal/insulator classification), bulk and shear moduli, (constant volume or pressure) heat capacity, Debye temperature, thermal expansion coefficient, and unit cell energy [54]. Given a POSCAR with the VASP format, the following script [55] can be used to obtain the AFLOW-ML data, to be used in a MOES for one of the aforementioned objectives

```
#!/usr/bin/python3
import json, sys, os
from time import sleep
from urllib.parse import urlencode
from urllib.request import urlopen
from urllib.request import Request
from urllib.error import HTTPError
```

```
SERVER="http://aflow.org"
API="/API/aflow-ml"
MODEL="plmf"
poscar=open('POSCAR', 'r').read()
encoded_data = urlencode({'file':
    poscar,}).encode('utf-8')
url = SERVER + API + "/" + MODEL +
    "/prediction"
request_task = Request(url, encoded_data)
task = urlopen(request_task).read()
task_json = json.loads(task.decode('utf-8'))
results_endpoint =
    task_json["results_endpoint"]
results_url = SERVER + API + results_endpoint
incomplete = True
while incomplete:
  request_results = Request(results_url)
  results = urlopen(request_results).read()
  results_json = json.loads(results)
  if results_json["status"] == 'PENDING':
    sleep(10)
    continue
  elif results_json["status"] == 'STARTED':
    sleep(10)
    continue
  elif results_json["status"] == 'FAILURE':
    print("Error: prediction failure")
    incomplete = False
  elif results_json["status"] == 'SUCCESS':
    print("Successful prediction")
    print(results_json)
    incomplete = False
```

It should be noted that AFLOW-ML models are also available for a series of chemical-formula-only-dependent properties, e.g., vibrational free energies and entropies [56] and the superconducting critical temperatures [57]. However, the outputs of these models are not relevant to the MOES implementation within XtalOpt, which uses a fixed chemical composition during the run.

References

[1] A. Merchant, S. Batzner, S. S. Schoenholz, M. Aykol, G. Cheon, E. D. Cubuk, Scaling deep learning for materials discovery, Nature 624 (7990) (2023) 80–85. doi:10.1038/s41586-023-06735-9.

[2] A. Sanna, T. F. Cerqueira, Y. W. Fang, I. Errea, A. Ludwig, M. A. Marques, Prediction of ambient pressure conventional superconductivity above 80 K in hydride compounds, npj Comput. Mater. 10 (1) (2024) 44. doi:10.1038/s41524-024-01214-9.

[3] K. Dolui, L. J. Conway, C. Heil, T. A. Strobel, R. P. Prasankumar, C. J. Pickard, Feasible Route to High-Temperature Ambient-Pressure Hydride Superconductivity, Phys. Rev. Lett. 132 (16) (2024) 166001. doi:10.1103/PhysRevLett.132.166001.

[4] C. J. Pickard, R. J. Needs, Structures at high pressure from random searching, Phys. status solidi 246 (3) (2009) 536–540. doi:10.1002/pssb.200880546.

[5] S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi, Optimization by Simulated Annealing, Science (80-.). 220 (4598) (1983) 671–680. doi:10.1126/science.220.4598.671.

[6] R. Martoňák, A. Laio, M. Parrinello, Predicting Crystal Structures: The Parrinello-Rahman Method Revisited, Phys. Rev. Lett. 90 (7) (2003) 075503. doi:10.1103/PhysRevLett.90.075503.

[7] S. Goedecker, Minima Hopping: An Efficient Search Method for the Global Minimum of the Potential Energy Surface of Complex Molecular Systems, J. Chem. Phys. 120 (21) (2004) 9911–9917. doi:10.1063/1.1724816.

[8] D. J. Wales, J. P. K. Doye, Global optimization by basin-hopping and the lowest energy structures of Lennard-Jones clusters containing up to 110 atoms, J. Phys. Chem. A 101 (28) (1997) 5111–5116. doi:10.1021/jp970984n.

[9] Y. Wang, J. Lv, L. Zhu, Y. Ma, Crystal structure prediction via particle-swarm optimization, Phys. Rev. B 82 (9) (2010) 094116. doi:10.1103/PhysRevB.82.094116.

[10] A. R. Oganov, C. W. Glass, Crystal structure prediction using ab initio evolutionary techniques: Principles and applications, J. Chem. Phys. 124 (24) (2006) 244704. doi:10.1063/1.2210932.

[11] D. C. Lonie, E. Zurek, XtalOpt: An open-source evolutionary algorithm for crystal structure prediction, Comput. Phys. Commun. 182 (2) (2011) 372–387. doi:10.1016/j.cpc.2010.07.048.

[12] S. Hajinazar, A. Thorn, E. D. Sandoval, S. Kharabadze, A. N. Kolmogorov, MAISE: Construction of neural network interatomic models and evolutionary structure optimization, Comput. Phys. Commun. 259 (2021) 107679. doi:10.1016/j.cpc.2020.107679.

[13] W. W. Tipton, R. G. Hennig, A grand canonical genetic algorithm for the prediction of multi-component phase diagrams and testing of empirical potentials, J. Phys. Condens. Matter 25 (49) (2013) 495401. doi:10.1088/0953-8984/25/49/495401.

[14] I. A. Kruglov, A. V. Yanilkin, Y. Propad, A. B. Mazitov, P. Rachitskii, A. R. Oganov, Crystal structure prediction at finite temperatures, npj Comput. Mater. 9 (1) (2023) 197. doi:10.1038/s41524-023-01120-6.

[15] P. T. Salzbrenner, S. H. Joo, L. J. Conway, P. I. C. Cooke, B. Zhu, M. P. Matraszek, W. C. Witt, C. J. Pickard, Developments and further applications of ephemeral data derived potentials, J. Chem. Phys. 159 (14) (2023) 144801. doi:10.1063/5.0158710.

[16] Z. Falls, P. Avery, X. Wang, K. P. Hilleke, E. Zurek, The XtalOpt Evolutionary Algorithm for Crystal Structure Prediction, J. Phys. Chem. C 125 (3) (2021) 1601–1620. doi:10.1021/acs.jpcc.0c09531.

[17] E. Zurek, Discovering New Materials via A Priori Crystal Structure Prediction, in: A. L. Parrill, K. B. Lipkowitz (Eds.), Rev. Comput. Chem., Vol. 29, John Wiley \& Sons, Inc., Hoboken, New Jersey, 2016, pp. 274–326. doi:10.1002/9781119148739.ch5.

[18] L. J. Conway, C. J. Pickard, A. Hermann, First principles crystal structure prediction, in: Compr. Inorg. Chem. III, Elsevier, 2023, pp. 393–420. doi:10.1016/B978-0-12-823144-9.00173-4.

[19] A. R. Oganov, C. J. Pickard, Q. Zhu, R. J. Needs, Structure Prediction Drives Materials Discovery, Nat. Rev. Mater. 4 (5) (2019) 331–348. doi:https://doi.org/10.1038/s41578-019-0101-8.

[20] Y. Wang, J. Lv, P. Gao, Y. Ma, Crystal Structure Prediction

via Efficient Sampling of the Potential Energy Surface, Acc. Chem. Res. 55 (15) (2022) 2068–2076. doi:10.1021/acs.accounts.2c00243.

[21] E. Zurek, W. Grochala, Predicting Crystal Structures and Properties of Matter under Extreme Conditions via Quantum Mechanics: The Pressure is on, Phys. Chem. Chem. Phys. 17 (5) (2015) 2917–2934. doi:10.1039/c4cp04445b.

[22] A. Parija, G. R. Waetzig, J. L. Andrews, S. Banerjee, Traversing Energy Landscapes Away from Equilibrium: Strategies for Accessing and Utilizing Metastable Phase Space, J. Phys. Chem. C 122 (45) (2018) 25709–25728. doi:10.1021/acs.jpcc.8b04622.

[23] A. Franceschetti, A. Zunger, The inverse band-structure problem of finding an atomic configuration with given electronic properties, Nature 402 (6757) (1999) 60–63. doi:10.1038/46995.

[24] S. V. Dudiy, A. Zunger, Searching for Alloy Configurations with Target Physical Properties: Impurity Design via a Genetic Algorithm Inverse Band Structure Approach, Phys. Rev. Lett. 97 (4) (2006) 046401. doi:10.1103/PhysRevLett.97.046401.

[25] Q. Zhu, A. R. Oganov, M. A. Salvadó, P. Pertierra, A. O. Lyakhov, Denser than diamond: Ab initio search for superdense carbon allotropes, Phys. Rev. B 83 (19) (2011) 193410. doi:10.1103/PhysRevB.83.193410.

[26] A. O. Lyakhov, A. R. Oganov, Evolutionary search for superhard materials: Methodology and applications to forms of carbon and TiO2, Phys. Rev. B 84 (9) (2011) 092103. doi:10.1103/PhysRevB.84.092103.

[27] Q. Zeng, A. R. Oganov, A. O. Lyakhov, C. Xie, X. Zhang, J. Zhang, Q. Zhu, B. Wei, I. Grigorenko, L. Zhang, L. Cheng, Evolutionary search for new high- k dielectric materials: methodology and applications to hafnia-based oxides, Acta Crystallogr. Sect. C Struct. Chem. 70 (2) (2014) 76–84. doi:10.1107/S2053229613027861.

[28] J. Qu, D. Zagaceta, W. Zhang, Q. Zhu, High dielectric ternary oxides from crystal structure prediction and high-throughput screening, Sci. Data 7 (1) (2020) 81. doi:10.1038/s41597-020-0418-6.

[29] E. J. Higgins, P. J. Hasnip, M. I. J. Probert, Simultaneous Prediction of the Magnetic and Crystal Structure of Materials Using a Genetic Algorithm, Crystals 9 (9) (2019) 439. doi:10.3390/cryst9090439.

[30] B. Wang, K. P. Hilleke, S. Hajinazar, G. Frapper, E. Zurek, Structurally Constrained Evolutionary Algorithm for the Discovery and Design of Metastable Phases, J. Chem. Theory Comput. 19 (21) (2023) 7960–7971. doi:10.1021/acs.jctc.3c00594.

[31] I. Giagkiozis, P. Fleming, Methods for multi-objective optimization: An analysis, Inf. Sci. (Ny). 293 (2015) 338–350. doi:10.1016/j.ins.2014.08.071.

[32] N. Gunantara, A review of multi-objective optimization: Methods and its applications, Cogent Eng. 5 (1) (2018) 1502242. doi:10.1080/23311916.2018.1502242.

[33] T. W. Liao, G. Li, Metaheuristic-based inverse design of materials – A survey, J. Mater. 6 (2) (2020) 414–430. doi:10.1016/j.jmat.2020.02.011.

[34] A. Solomou, G. Zhao, S. Boluki, J. K. Joy, X. Qian, I. Karaman, R. Arróyave, D. C. Lagoudas, Multi-objective Bayesian materials discovery: Application on the discovery of precipitation strengthened NiTi shape memory alloys through micromechanical modeling, Mater. Des. 160 (2018) 810–827. doi:10.1016/j.matdes.2018.10.014.

[35] A. M. Gopakumar, P. V. Balachandran, D. Xue, J. E. Gubernatis, T. Lookman, Multi-objective Optimization for Materials Discovery via Adaptive Design, Sci. Rep. 8 (1) (2018) 3738. doi:10.1038/s41598-018-21936-3.

[36] D. Khatamsaz, B. Vela, P. Singh, D. D. Johnson, D. Allaire, R. Arróyave, Multi-objective materials bayesian optimization with active learning of design constraints: Design of ductile refractory multi-principal-element alloys, Acta Mater. 236 (2022). doi:10.1016/j.actamat.2022.118133.

[37] L. Yang, D. Robin, F. Sannibale, C. Steier, W. Wan, Global optimization of the magnetic lattice using genetic algorithms, Conf. Proc. C 0806233 (2008) THPC033.

[38] H.-Z. Chen, Y.-Y. Zhang, X. Gong, H. Xiang, Predicting New TiO 2 Phases with Low Band Gaps by a Multiobjective Global Optimization Approach, J. Phys. Chem. C 118 (5) (2014) 2333–2337. doi:10.1021/jp411437f.

[39] Y.-Y. Zhang, W. Gao, S. Chen, H. Xiang, X.-G. Gong, Inverse design of materials by multi-objective differential evolution, Comput. Mater. Sci. 98 (2015) 51–55. doi:10.1016/j.commatsci.2014.10.054.

[40] M. Núñez-Valdez, Z. Allahyari, T. Fan, A. R. Oganov, Efficient technique for computational design of thermoelectric materials, Comput. Phys. Commun. 222 (2018) 152–157. doi:https://doi.org/10.1016/j.cpc.2017.10.001.

[41] J. J. Maldonis, Z. Xu, Z. Song, M. Yu, T. Mayeshiba, D. Morgan, P. M. Voyles, StructOpt: A modular materials structure optimization suite incorporating experimental data and simulated energies, Comput. Mater. Sci. 160 (2019) 1–8. doi:10.1016/j.commatsci.2018.12.052.

[42] C. Y. Cheng, J. E. Campbell, G. M. Day, Evolutionary chemical space exploration for functional materials: computational organic semiconductor discovery, Chem. Sci. 11 (19) (2020) 4922–4933. doi:10.1039/D0SC00554A.

[43] J. Meng, M. Abbasi, Y. Dong, C. Carlos, X. Wang, J. Hwang, D. Morgan, Experimentally informed structure optimization of amorphous TiO 2 films grown by atomic layer deposition, Nanoscale 15 (2) (2023) 718–729. doi:10.1039/D2NR03614B.

[44] J. Horn, N. Nafpliotis, D. Goldberg, A niched Pareto genetic algorithm for multiobjective optimization, in: Proc. First IEEE Conf. Evol. Comput. IEEE World Congr. Comput. Intell., Vol. 1, IEEE, 1994, pp. 82–87. doi:10.1109/ICEC.1994.350037.

[45] N. Srinivas, K. Deb, Muiltiobjective Optimization Using Nondominated Sorting in Genetic Algorithms, Evol. Comput. 2 (3) (1994) 221–248. doi:10.1162/evco.1994.2.3.221.

[46] K. Deb, Multi-objective Optimisation Using Evolutionary Algorithms: An Introduction, in: L. Wang, A. H. C. Ng, K. Deb (Eds.), Multi-objective Evol. Optim. Prod. Des. Manuf., Springer London, London, 2011, pp. 3–34. doi:10.1007/978-0-85729-652-8_1.

[47] P. Avery, C. Toher, S. Curtarolo, E. Zurek, XtalOpt Version r12: An open-source evolutionary algorithm for crystal structure prediction, Comput. Phys. Commun. 237 (2019) 274–275. doi:10.1016/j.cpc.2018.11.016.

[48] P. Avery, X. Wang, C. Oses, E. Gossett, D. M. Proserpio, C. Toher, S. Curtarolo, E. Zurek, Predicting superhard materials via a machine learning informed evolutionary structure search, npj Comput. Mater. 5 (1) (2019) 89. doi:10.1038/s41524-019-0226-8.

[49] X. Wang, D. M. Proserpio, C. Oses, C. Toher, S. Curtarolo, E. Zurek, The Microscopic Diamond Anvil Cell: Stabilization of Superhard, Superconducting Carbon Allotropes at Ambient Pressure, Angew. Chemie Int. Ed. 61 (32) (2022). doi:10.1002/anie.202205129.

[50] M. T. M. Emmerich, A. H. Deutz, A tutorial on multiobjective optimization: fundamentals and evolutionary methods, Nat. Comput. 17 (3) (2018) 585–609. doi:10.1007/s11047-018-9685-y.

[51] G. Kresse, J. Hafner, Ab initio molecular dynamics for liquid metals, Phys. Rev. B 47 (1) (1993) 558–561. doi:10.1103/PhysRevB.47.558.

[52] G. Kresse, J. Furthmüller, Efficient iterative schemes for ab initio total-energy calculations using a plane-wave basis set, Phys. Rev. B 54 (16) (1996) 11169–11186. doi:10.1103/PhysRevB.54.11169.

[53] A. Hjorth Larsen, J. Jørgen Mortensen, J. Blomqvist, I. E. Castelli, R. Christensen, M. Dułak, J. Friis, M. N. Groves, B. Hammer, C. Hargus, E. D. Hermes, P. C. Jennings, P. Bjerre Jensen, J. Kermode, J. R. Kitchin, E. Leonhard Kolsbjerg, J. Kubal, K. Kaasbjerg, S. Lysgaard, J. Bergmann Maronsson, T. Maxson, T. Olsen, L. Pastewka, A. Peterson, C. Rostgaard, J. Schiøtz, O. Schütt, M. Strange, K. S. Thygesen, T. Vegge, L. Vilhelmsen, M. Walter, Z. Zeng, K. W. Jacobsen, The atomic simulation environment—a Python library for working with atoms, J. Phys. Condens. Matter 29 (27) (2017) 273002. doi:10.1088/1361-648X/aa680e.

[54] O. Isayev, C. Oses, C. Toher, E. Gossett, S. Curtarolo, A. Tropsha, Universal Fragment Descriptors for Predicting Properties of Inorganic Crystals, Nat. Commun. 8 (2017) 15679. doi:10.1038/ncomms15679.

[55] E. Gossett, C. Toher, C. Oses, O. Isayev, F. Legrain, F. Rose, E. Zurek, J. Carrete, N. Mingo, A. Tropsha, S. Curtarolo, AFLOW-ML: A RESTful API for machine-learning predictions of materials properties, Comput. Mater. Sci. 152 (2018) 134–145. doi:10.1016/j.commatsci.2018.03.075.

[56] F. Legrain, J. Carrete, A. van Roekeghem, S. Curtarolo, N. Mingo, How Chemical Composition Alone Can Predict Vibrational Free Energies and Entropies of Solids, Chem. Mater. 29 (15) (2017) 6220–6227. doi:10.1021/acs.chemmater.7b00789.

[57] V. Stanev, C. Oses, A. G. Kusne, E. Rodriguez, J. Paglione, S. Curtarolo, I. Takeuchi, Machine learning modeling of superconducting critical temperature, npj Comput. Mater. 4 (1) (2018) 29. doi:10.1038/s41524-018-0085-8.