

Interactive exploration of high-dimensional phase diagrams

A. van de Walle,^{1,2} H. Chen,¹ H. Liu,¹ C. Nataraj,¹ S. Samanta,¹ S. Zhu¹ and R. Arroyave³

1 — School of Engineering, Brown University, Providence, RI 02912, USA.

2 — Corresponding author, email: avdw@alum.mit.edu

3 — Department of Materials Science & Engineering, Texas A&M University, College Station, TX 77843, USA.

April 25, 2022

Abstract

High-dimensional thermodynamic phase stability databases are becoming increasingly common due to the convergence of three recent trends: (i) the widespread interest in so-called “high-entropy” alloys, (ii) the availability of high-throughput computational assessments of phase stability in broad composition spaces and (iii) the ongoing development of ever-increasingly broad, multicomponent, multiphase CALPHAD databases. Although automated computational tools can readily process such high-dimensional data, scientists are often unable to visualize the relevant phase relations, an ability that is crucial to gaining an intuitive understanding of the stability constraints governing materials design. The present work addresses this need by providing algorithms that enable the interactive exploration of phase equilibria in high-dimensional spaces. These algorithms concentrate the complex nonlinear nonsmooth optimization needed into a preprocessing step that generates a large number of high-dimensional yet elementary graphical primitives. These primitives can then be cross-sectioned to yield 3-dimensional views in a computationally efficient manner that enables an interactive exploration of high-dimensional spaces. All of these operations are highly parallelizable, thus facilitating scaling of this method to large data sets.

1 Introduction

Materials scientists and engineers are well-acquainted with phase diagram handbooks that have guided materials design for decades, but such resources fall short when the number of components is too large for the phase diagram to be effectively represented on a 2-dimensional medium. This paper presents software tools that enable the interactive visualization and exploration of phase equilibria in high-dimensional spaces, to provide a unique window into high-dimensional thermodynamic phase stability databases.

Such databases are becoming increasingly common due to the convergence of three recent trends. First, the concepts of high-entropy alloys or multiple principal component alloys (e.g. [1, 2]) have stimulated the exploration of a wide range of complex alloy chemistries and these efforts are generating high-dimensional phase stability data as a by-product.

Second, the development of high-throughput computational methods that enable the large-scale discovery of stable and metastable phases produces, by design, large amounts of high-dimensional phase stability data. While initial efforts concentrated on the exploration of ordered phase energetics at absolute zero [3, 4, 5, 6, 7], these efforts are being extended beyond stoichiometric compounds to yield more complete free energy models that feature composition- and temperature-dependence [8, 9, 10, 11, 12].

Finally, the CALPHAD (CALculation of PHase Diagram) community has been steadily expanding the coverage of complex chemistries by developing increasingly broad multicomponent multiphase CALPHAD databases. These efforts have been ongoing for decades and the amount of available data is vast and fuels a materials research and development ecosystem. These developments have taken place both in industry, via proprietary database development [13, 14, 15], and in the open scientific literature [16, 17, 18, 19, 20]. Efforts are under way to consolidate these data into centralized meta-databases [21, 22].

In all of the above cases, the data takes the form of composition- and temperature- dependent free energies and obtaining a phase diagram involves finding the mixture of phases and their respective compositions such

that the free energy is minimized under given imposed conditions (e.g. overall composition, temperature and pressure).

The determination of equilibrium phase boundaries from free energy models is a problem that has received considerable attention and suitable algorithms have been implemented in numerous commercial ([13, 14, 15]) and open-source ([23, 24, 25]) software. Still, this task involves a constrained nonlinear optimization problem (with possible multiple local optima) with inequality constraints that are potentially binding and where solutions could depend non-smoothly on the input conditions. As CALPHAD practitioners are well-aware, these features make it difficult to devise algorithms that are both rapid and reliable. Efficient boundary tracing techniques do exist ([26, 27, 28]) but require a starting point equilibrium that often needs to be found by more extensive calculations. The possibility of multiple local optima prompts the need for brute-force linear or grid searches that, while effective, are considerably slower and are less suitable for an interactive software tool.

To address this, we propose to carry out the expensive computations of the thermodynamic equilibria as a preprocessing step taking place before the interactive visualization. We rely on sampling schemes to ensure better scaling to high dimensions and facilitate parallelization. Once the high-dimensional phase boundaries have been determined and expressed as simple geometric primitives, the interactive visualization step only involves simple linear operations that can efficiently be implemented to ensure a real-time feedback to user input.

In the following sections, we first describe our algorithms, along with their rationale, before providing a few examples inspired by high entropy alloy design.

2 Methods

Let us first define some useful concepts. In a phase diagram at constant temperature and pressure, if there are e elements, there are $e - 1$ compositional degrees of freedom and the phase diagram has dimension $n = e - 1$. To treat all compositions symmetrically, they are traditionally represented in a Gibbs triangle when $e = 3$. We generalize this to a Gibbs simplex (a simplex in $n = e - 1$ dimensions is a polytope with $n + 1 = e$ vertices). When temperature is considered, another dimension orthogonal to the Gibbs simplex is added, so $n = e$, and the phase diagram is then contained within a hyper-prism with a “base” that is a simplex with e vertices. One can proceed similarly to include pressure, although we do not consider this here.

The phase boundaries are, in general, $(n - 1)$ -dimensional curved hyper-surfaces, which will be hereafter called *manifolds*. We represent these manifolds by points interconnected to form a mesh. The basic building block of that mesh is a $(n - 1)$ -dimensional simplex (For instance in 3 dimensions, surfaces can meshed by triangles, i.e. 2-dimensional simplexes). When $n > 3$, we wish to perform a 3-dimensional cross-section of the phase diagram for plotting purposes. The remaining dimensions (orthogonal to the 3-dimensional cross-section hyper-plane) can be accessed as the user interactively changes the hyper-plane of the cross-section. (We use the term hyper-plane rather than hyper-volume to emphasize that it is lower dimensional than the full phase diagram.)

Let us now outline our computational approach, which is also illustrated in Figure 1 in a low-dimensional setting for clarity. Each step will be detailed further below.

1. **Random sampling.** First, the temperature-composition space is randomly sampled from a uniform distribution.
2. **Equilibria calculations.** At every imposed condition (overall composition and temperature), the thermodynamic equilibrium is calculated. All the calculations yielding a single-phase equilibrium are discarded while those yielding a multi-phase equilibrium are kept (miscibility gaps are considered multi-phase equilibria as well).
3. **Phase boundary classification.** The remaining data points provide, for each phase, a set of points that samples its phase boundary in temperature-composition space. Points from different equilibria but associated with the same phase are grouped. Within a group, the data points are placed in sub-groups based on which other phase(s) they are in equilibrium with.

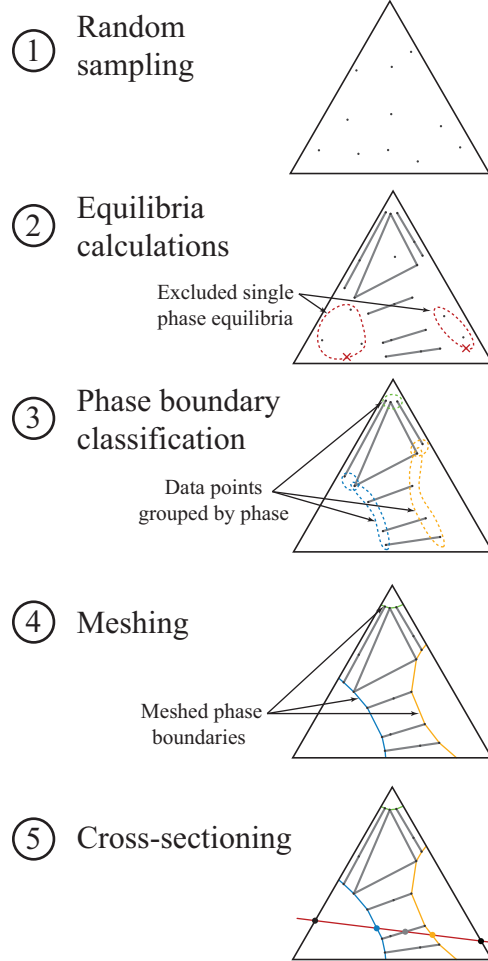


Figure 1: Computational approach for the determination of low-dimensional cross-sections of high-dimensional phase diagrams. For plotting purposes, the “high” dimension is 2, while the “low” dimension is 1 and an isothermal section is considered. Step numbers correspond to those described in the main text. In Step 1, composition points are drawn uniformly at random within the Gibbs triangle. In Step 2, phase equilibria associated with each imposed overall composition are calculated and those yielding a single phase (encircled by a dotted line) equilibrium are discarded. In Step 3, end points of the tie-lines are grouped (as shown by dotted lines) by phase (distinguished by colors) and sub-grouped based on the phase they are in equilibrium with. In Step 4, each subgroup is meshed and the resulting meshes are re-grouped by phase (indicated by different colors). In Step 5, the cross section (along the red line) of each simplex (here shown as line segments) constituting the meshes is calculated.

4. **Meshing.** Within each subgroup, the sets of points can be meshed to form polygonal hyper-surfaces. The meshes for each subgroup are then re-grouped with those associated with the same phase. Of course, in an n -dimensional space, these are actually $(n - 1)$ -dimensional manifolds decomposed into n -point simplexes.
5. **Cross-sectioning.** 3-dimensional cross-sections of these manifolds are then computed. The problem of calculating a cross section of a manifold decomposed into simplexes is a linear convex programming problem that can be implemented in a computationally efficient and reliable fashion.

Random sampling (Step 1) uniformly on the Gibbs simplex can be accomplished in a number of ways. For an e -component system, the simplest way is to draw compositions x_i for $i = 1, \dots, e - 1$ each from a uniform distribution on $[0, 1]$ and to reject trial draws summing to greater than 1 (the last composition x_e is then determined by $x_e = 1 - \sum_{i=1}^{e-1} x_i$).¹ An efficient algorithm that does not involve rejections is to draw e random numbers from independent exponential distributions and normalize them to sum to 1. (The use of an exponential distribution is critical in this scheme, as other distributions would not yield a uniform distribution on the Gibbs simplex.) It is also possible to replace purely random sampling by deterministic *quasi-random* or *minimum discrepancy* sequences with improved sampling properties (in terms of avoiding unnecessarily close points while maintaining probabilistic validity). A detailed discussion of the implementations and of the relative merits of the above schemes can be found in [29]. Other popular uniform sampling schemes, such as Latin hypercube sampling [30], may be difficult to adapt to a Gibbs simplex geometry.

Random sampling (or its deterministic alternatives) provides a desirable way to sample high-dimensional spaces due to favorable scaling as the dimension of the space increases as well as due to the ability to better control the computational cost. In contrast, in grid-type sampling, one has very little choice in the number of sample points: the jump in the number of points between two consecutive grid sizes can be very large, leaving the user the unenviable choice between a too coarse grid and a very computationally demanding calculation. With random sampling, the densest computationally feasible sampling can always be selected. One can simply stop point generation when an adequate sampling has been achieved without having to plan in advance how many samples will have to be drawn. As an added benefit, random sampling also ensures that the probability of accidentally missing a phase is proportional to the volume it, and its associated multi-phase equilibria, occupy in temperature-composition space [31]. The probability of missing a phase also decays exponentially to zero as the number of sample points is increased. Grid-based methods do not possess these advantages.

During the equilibria calculations (Step 2), the random sampling approach offers the advantage that the expensive thermodynamic equilibria calculations can be very easily parallelized without requiring the different computing threads to coordinate their actions, in contrast to boundary-following approaches. The latter are also difficult to generalize to arbitrary dimensions and, in fact, to our knowledge, these methods have so far not been used to generate phase boundary manifolds in high dimensions.

Once a large number of equilibria have been calculated, the single-phase equilibria are discarded (as they provide no information regarding the phase boundaries) while the multi-phase equilibria are classified (in Step 3) in terms of which phases take part in each equilibrium. For instance, if there are three phases (α, β, γ) in the phase diagram, the equilibria are classified in 4 groups: $(\alpha, \beta), (\beta, \gamma), (\alpha, \gamma), (\alpha, \beta, \gamma)$, which we shall denote by the subscripts 1 to 4. Next, for each group, we extract the 9 manifolds corresponding to each phase boundary, namely: $\alpha_1, \beta_1, \beta_2, \gamma_2, \alpha_3, \gamma_3, \alpha_4, \beta_4, \gamma_4$. The tie-triangle data-points (and higher-order equilibria, if any) are then combined with the appropriate group: $\alpha_1 \cup \alpha_4, \beta_1 \cup \beta_4, \beta_2 \cup \beta_4, \gamma_2 \cup \gamma_4, \alpha_3 \cup \alpha_4, \gamma_3 \cup \gamma_4$ to ensure each manifold has edges that just touch the adjacent manifold. Each of these group will be meshed separately. The rationale for this classification is that all points within one group sample a smooth (i.e. continuously differentiable) and connected manifold. In contrast, equilibria that involve different phases are either disconnected in composition-temperature phase, or, when they are connected, there will typically be a kink (i.e. discontinuous derivative) at the junction. The classification thus ensures that the phase diagram is decomposed into smooth objects.²

¹This scheme produces genuine uniform sampling because (i) a subregion of a uniformly sampled region is also uniformly sampled and (ii) a nondegenerate linear transformation preserves uniform sampling.

²This statement assumes that the underlying thermodynamic engine uses different labels for the phases on either side of a higher-order phase transition.

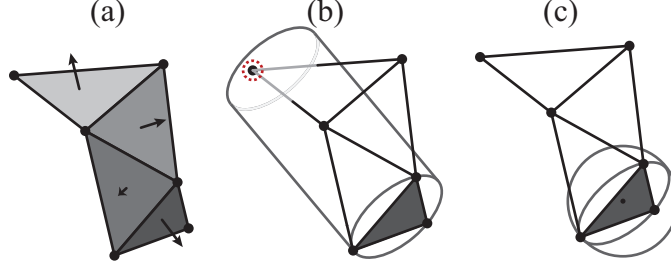


Figure 2: Meshing a curved surface. The mesh intended to be generated is shown in Panel (a), with triangle normals shown to convey curvature information. Panel (b) shows that a straightforward application of the Delaunay triangulation criterion for a flat surface incorrectly flags an invalid mesh due to the point circled in red falling within the circumscribed cylinder. This problem is avoided if one instead uses a circumscribed sphere whose center lies on the current facet, as shown in Panel (c).

The meshing step (Step 4) is not as straightforward as it may appear to be at first. For a set of points on a flat two-dimensional plane, Delaunay triangulation [32] is a standard method to construct suitable connecting triangles. The geometric condition that such a construction must satisfy is simply that the circumscribed circle around each triangle must not contain any other points other than the triangle’s vertices. This condition can be straightforwardly used to devise a constructive algorithm: given a segment between two points, search for a third point such that the resulting circumscribed circle satisfies the condition and forms a triangle. The algorithm is then applied to the newly created segments, etc. This algorithm straightforwardly generalizes to n dimensions: one simply finds simplexes connecting $n + 1$ points such that the circumscribed hyper-sphere does not contain any other points. Implementations of this algorithm are readily available [33].

Our task is more complex, however: the meshing we seek is not generated from points in a flat space. We need to mesh a $(n - 1)$ -dimensional curved manifold embedded in an n -dimensional space.³ Software packages able to handle both hyper-surface curvature and spaces of general dimension appear to not yet be available, to the best of the authors’ knowledge. A direct application of one step of Delaunay “simplexization” might find multiple points to add to the mesh or no points at all, due to the curvature of this manifold. The case of multiple points can be easily addressed by a tie-breaking rule (e.g. proximity to the previously meshed point). However, the case of no suitable points, illustrated in Figure 2(b), demands a modification to the algorithm. Instead of checking if no other points belong to a $(n - 1)$ -dimensional sphere (extended to a n -dimensional cylinder, as shown in Figure 2(b)), one checks if no other points belong to an n -dimensional hyper-sphere circumscribing the n points of the new candidate $(n - 1)$ -dimensional simplex with the constraint that the hyper-sphere center lies along the plane of that $(n - 1)$ -dimensional simplex, as shown in Figure 2(c). This modified algorithm is then applicable to curved manifolds, as long as the manifold’s radius of curvature does not approach the radius of the circumscribed hyper-spheres. For a continuously differentiable hyper-surface, the latter condition is always eventually satisfied if the sampling points are sufficiently close. Fortunately, phase boundaries are typically continuously differentiable, except at points where phases appear or disappear.

This latter observation is what motivates grouping together phase boundary points that share the same combination of phases in equilibrium. This ensures that the resulting manifold to be meshed is smooth. Once each group of points has been meshed separately, they can be combined into a single mesh to create a continuous hyper-surface with possible kinks where there are changes in the combination of phases that are in equilibrium.

The meshing step can be implemented in a parallel architecture as follows. Each of the N instances of the code keeps track of the mesh constructed so far and looks for simplex faces that are at the boundary of the manifold meshed so far. Each of these faces is assigned a numerical index i . Instance number t of the code works on the face with smallest i such that $i \bmod N = t$. Each instance spends most of its time trying to find a neighboring point that satisfies the circumscribed hyper-sphere criterion. When an instance finds such a

³Note that this task cannot be accomplished by performing the n -dimensional Delaunay triangulation and keeping all the $(n - 1)$ -dimensional faces that belong to only one n -dimensional simplex. This would give a mesh for the *convex hull* of the boundary points while, in general, phase boundaries need not be convex.

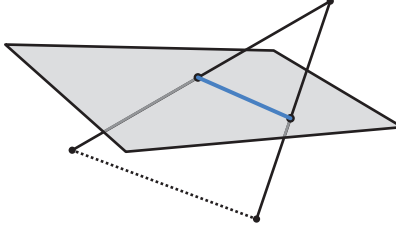


Figure 3: Calculating the cross-section of a simplex. Example of a 2-dimensional cross-section ($c = 2$) of a 2-dimensional simplex ($n - 1 = 2$) embedded in a 3-dimensional space ($n = 3$). One picks every subset of $n - c + 1 = 2$ vertices from the simplex and checks whether the $(n - c)$ -dimensional simplex (here, segments, since $n - c = 1$) generated by convex combinations of these vertices intersects the cross-section plane (in grey). Here, two of the segments meet this criterion while one (dotted) does not. The two successful intersections yield two points in the plane that can be meshed by $(c - 1)$ -dimensional simplex(es) (here shown as a thick blue segment, since $c - 1 = 1$).

point, it shares the information with all other instances and all of them update their internal mesh. Only the point index and face index need to be shared, so the communication overhead is kept to a minimum. In addition, the number of faces on the boundary of the mesh is large during most of the calculations (except towards the beginning and the end of the meshing process), so the potential for parallelization is large as well.

Thanks to the fact that the meshing step represents the phase boundary manifolds as a union of simplexes, the process of computing a cross-section of the phase diagram (Step 5) reduces to computing, many times, the cross-section of a simplex. The c -dimensional cross section of an $(n - 1)$ -dimensional simplex could, in principle, be calculated by solving a generic linear programming problem. However, this task has a considerably more specific structure which can be exploited to devise a more efficient algorithm.

The vertices of a c -dimensional cross section of an $(n - 1)$ -dimensional simplex can be obtained by considering, in turn, any subset of $n - c + 1$ of these vertices. For each subset of vertices, one computes which weighted average of these vertices yields a point that lies along the cross-section hyper-plane. If the weights are all positive, then this point is a vertex of the cross-section but, otherwise, this point should be discarded. This algorithm is illustrated in Figure 3 for $n = 3$ and $c = 2$ and is more formally described in Appendix A. Note that there is no guarantee that the cross section of a simplex is itself a $(c - 1)$ -dimensional simplex, although it is at least guaranteed to be convex. For each simplex cross-section, the resulting points can be trivially meshed because they lie on a flat surface and there are typically very few of them. It is important to emphasize that the cross-section operation essentially involves repeatedly solving linear systems of equations, a task that is very easy to vectorize, parallelize or perform via Graphical Processing Units (GPU). Another efficiency consideration is that only a small fraction of the high-dimensional simplexes is being cut through for a given visualized cross-section. These simplexes relevant for visualization can be very quickly identified by simply looking at the pattern of signs of the vertices' coordinates (once all coordinates have been transformed so that the cross-section hyperplane crosses the origin and contains the first c Cartesian axes). This implies that the method scales well with the dimension of the high-dimensional space.

Phase diagrams for more than 2 components traditionally include tie-lines to clarify which phases are in equilibrium. For a generic 3-dimensional cross-section in an n -dimensional ($n > 3$) space, the probability that a tie-line lies exactly in the subspace of the chosen cross-section is negligible. Although this would suggest that tie-lines should not even be plotted in this context, this is not entirely satisfying. It would be useful to visualize how close the cross-section is to lining up with a given tie-line, as this could help guide the user towards cross-sections that are more informative or easier to interpret. To this effect, we propose to represent tie-lines by hyper-ellipsoids elongated along the direction of the tie-line and narrowed along directions perpendicular to it (see Figure 4). This has the effect of smoothly interpolating between a small sphere (when the tie-line is perpendicular to the cross-section) and an elongated ellipsoid resembling a conventional tie-line (when the tie-line is parallel to the cross-section). This lets the user gauge how close the tie-line is to being co-planar with the selected cross-section. A similar idea could be used to represent tie-triangles (and, more generally, tie-simplexes), using hyper-ellipsoids whose long principal axes lie in the

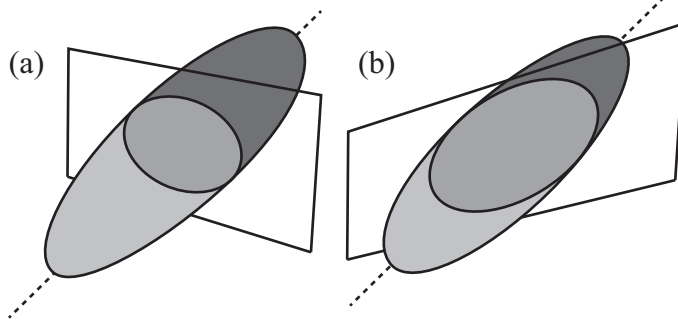


Figure 4: Tie-line representation by a hyper-ellipsoid and its cross-section. Each hyper-ellipsoid (here shown as 3-dimensional ellipsoid) is elongated along the direction of the tie-line (shown as a dashed line) and narrowed perpendicular to it. When the tie-line is almost perpendicular to the cross-section (here 2-dimensional), it appears close to a circle and, when the tie-line is almost parallel to the cross-section, it appears as an ellipse.

hyper-plane of the tie-simplex and whose short principal axes are perpendicular to it. For rendering efficiency purposes, these hyper-ellipsoids are triangulated into simplexes.

3 Implementation

The above algorithms have been implemented within the Alloy Theoretic Automated Toolkit (ATAT) [34, 35]. The main command for preprocessing is `ocplotpd` and takes the form of a Perl script that calls OpenCalphad [36] to perform the thermodynamic equilibrium calculations and various ATAT commands (implemented in C++) that perform the meshing (`simplexize` command) or generate coordinate axes and labels (`mkaxes` command). The script `ocplotpd` can spawn multiple instances of OpenCalphad to take advantage of multiple cores while the `simplexize` command can exploit parallelization via MPI.

The command `ocplotpd` takes as an input a thermodynamic database in the standard TDB format and produces, as an output, one of the following:

1. 2- or 3-dimensional output suitable for viewing with `gnuplot`
2. 3-dimensional output in standard `vtk` format [37] suitable for viewing with `ParaView` [38].
3. n -dimensional output in the form of simplex-meshed manifolds in ATAT’s “`nd`” format (which stands for “ n -dimensional”).

The outputs in the forms of item 1 and 2 above are already being used to generate the graphical output for Thermodynamic Database Database (TDBDB) [22], while the output in the form of item 3 is the main novel contribution of this article. Files in the “`nd`” format can be viewed with the ATAT command `ndviewer`, which generates 3-dimensional cross-sections interactively while allowing the user to move and rotate in n dimensions. This code is implemented in C++, with graphical aspects handled using OpenGL via the GLUT library. The parts of the code implementing linear algebra operations can be linked with BLAS and LAPACK, for which GPU-aware implementations are readily available.

4 Applications

As a first example, we consider the well-known Co-Cr-Fe-Ni-V high-entropy alloy, for which a thermodynamic database from experimental data has been recently developed [39]. We compute this system’s 4-dimensional phase diagram at 1500K and view various 3-dimensional cross-sections.

Let us first provide some statistics that convey an order of magnitude of the computations involved. Thermodynamic equilibrium calculations were performed at 100,000 sample points, yielding about 50,000 points

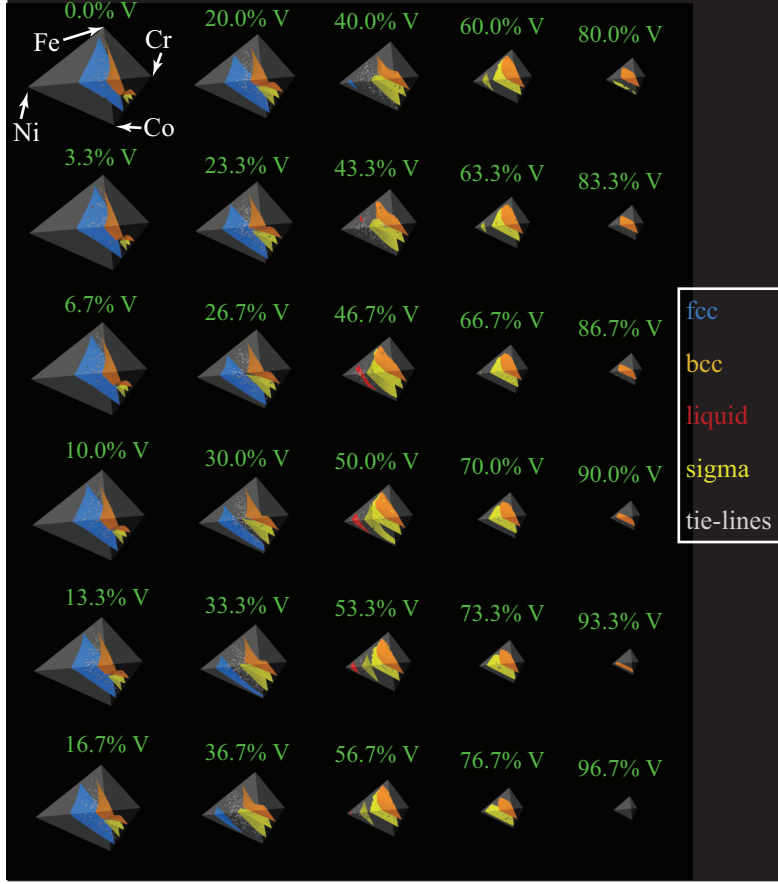


Figure 5: Example of interactive exploration of the Co-Cr-Fe-Ni-V phase diagram (1500K isothermal section). Surfaces indicate phase boundaries, color coded by phase, while tie-lines are shown in white. The image sequence starts at the top left with the Gibbs tetrahedron associated with the Co-Cr-Fe-Ni system and the remaining image sequence shows (from top to bottom and then left to right) cross-sections of constant V concentration with V content gradually going from 0 to 100 at. % in successive images.

on the phase boundaries. We kept about 10% of the calculated tie-lines for plotting purposes, yielding about 2400 tie-lines. The meshed boundary points resulted in about 320,000 simplexes while the representation of the tie-lines added about 20,000 simplexes.

The calculations were performed on a single 24-core node (Intel e5-2670, Skylake architecture). The parallelized thermodynamic calculations took about 10 min of wall clock time while the meshing took about 30 min of wall clock time. Once this pre-processing step was completed, the phase diagram could be viewed interactively at about 5 frames per second on a mid-range laptop (1.90GHz Intel i7-8650U CPU without discrete graphic card). These software tools are still undergoing significant efficiency improvements, however, and the timings we report here are likely to continuously improve (for instance, our current implementation of the viewer does not exploit multi-threading or the availability of multiple cores).

A typical output is shown in Figure 5, where the cross-section hyper-plane is slowly moved from the 0 at. % V to 100 at. % V. This type of exploration is useful to identify regions of composition space where the detrimental σ phase does not form.

As a second example, we consider the promising Cr-Mo-Nb-V-W high-entropy alloy system [40]. In this example, the thermodynamic Calphad model is generated from high-throughput ab initio calculations, following the method described in [8] and the parameters listed in Appendix B. We focus on the thermodynamics of the bcc phase and obtain its metastable phase diagram. This exercise enables us to find which regions of composition space are free of miscibility gaps, so that synthesizing a bcc alloy that is at least

metastable would be a possibility. Of course, other phases based on other crystal structures could further reduce the set of feasible alloys if one wishes to require a strict thermodynamic equilibrium, but we leave the investigation of this possibility to future communications.

The calculation process involved the computation of 70 Special Quasirandom Structures [41, 42] (with unit cell sizes ranging from 32 to 48 atoms) spanning the full composition range of the 5-component alloy. The resulting random alloy formation energies were then combined with short-range-order contributions based on the Cluster Variation Method [43], as described in [8], and used to build a Calphad model.

Thermodynamic equilibrium calculations were performed at 100,000 sample points, yielding about 60,000 points on the phase boundaries. We kept about 10% of the calculated tie-lines for plotting purposes, yielding about 5000 tie-lines (many of which are actually the boundaries of tie-triangles). The meshed boundary points resulted in about 360,000 simplexes, while the representation of the tie-lines added about 45,000 simplexes.

In this example, we can explore how the shape of the miscibility gaps changes along different cross-sections. In panel (a) of Figure 6, we clearly see, from the tie-lines and phase boundaries, that 50 at. % W alloy with similar V and Nb content phase separates into a V-rich and a Nb-rich alloy while the addition of Cr to this alloy does not lead to the precipitation of a Cr-rich phase. In contrast, Panel (g) of Figure 6 also shows a miscibility gap in another Cr-poor portion of the phase diagram, but the tie-lines there actually point to a phase separation that involves Cr-rich phases. The figure also shows the gradual transition between these two situations, and this type of behavior would be very difficult to investigate without an interactive tool such as the one proposed herein.

5 Conclusion

This paper has introduced software tools that enable the interactive visualization and exploration of phase equilibria in high-dimensional spaces, in an effort to bring the traditional handbooks of phase diagrams into the next century and meet the needs of the increasing community of researchers relying on high-dimensional phase stability data. In analogy with the echographies used in medicine, where interactive 2-dimensional cross-sections enable viewing of a 3-dimensional body, our software tools enable interactive 3-dimensional cross-sections that facilitate the exploration of higher-dimensional spaces. We have presented here a snapshot of the current status of these tools — they are being continuously improved in terms of performance and usability.

In addition to various algorithmic innovations, our contribution is to observe that the complex process of rendering high-dimensional phase diagrams can be broken down into (i) a computationally intensive pre-processing step handling all the complex, nonlinear and nonsmooth operations that can be performed in advance and (ii) an interactive visualization step in which only simple linear operations on elementary graphical primitives have to be carried out. Such operations can be easily vectorized, parallelized or performed by Graphical Processing Units (GPU).

The interactive high-dimensional viewer presented herein is actually agnostic regarding the type of data to be viewed. Hence, a by-product of our efforts is to provide general tools to view general high-dimensional scientific data through cross-sections.

Acknowledgements

This research was supported by NSF under grant DMR-2001411, by ARPA-e under grant DE-AR0001427 and by Brown University through the use of the facilities at its Center for Computation and Visualization. This work uses the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by National Science Foundation grant number ACI-1548562.

Conflict of interest

On behalf of all authors, the corresponding author states that there is no conflict of interest.

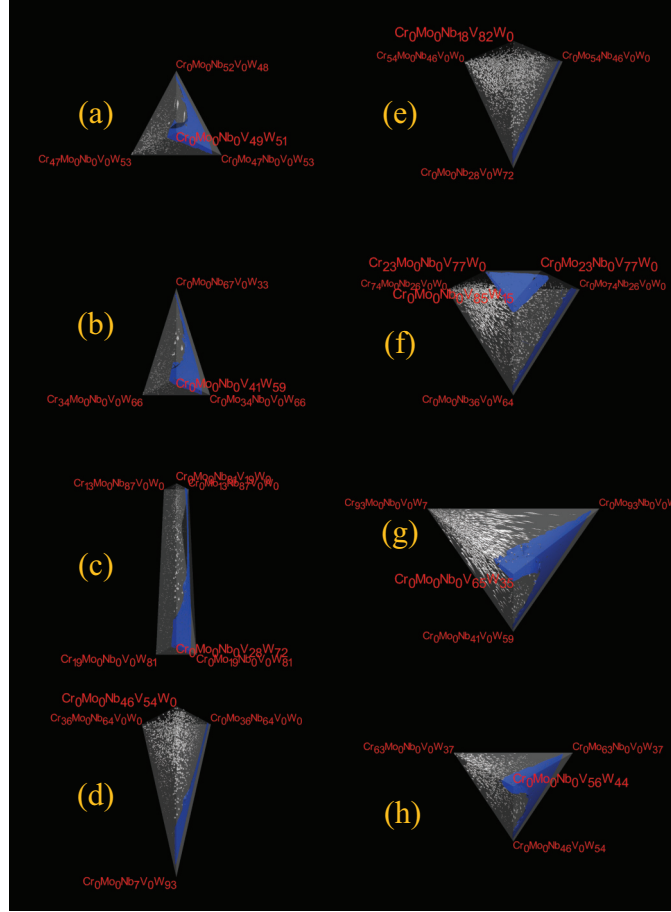


Figure 6: Example of interactive exploration of the Cr-Mo-Nb-V-W metastable bcc phase diagram. Blue surfaces indicate the bcc phase boundary while tie-lines are shown in white. Compositional coordinates of the vertices are indicated in red. The image sequence starts at the top left with a cross-section at 50 at. % W. The remaining image sequence shows (from top to bottom and then left to right) a rotation along the V-W hyper-plane.

References

- [1] E.P. George, D. Raabe, R.O. Ritchie, *Nature Reviews Materials* **4**, 515 (2019)
- [2] J.W. Yeh, S.K. Chen, S.J. Lin, J.Y. Gan, T.S. Chin, T.T. Shun, C.H. Tsau, S.Y. Chang, *Advanced Engineering Materials* **6**, 299 (2004)
- [3] D. Morgan, G. Ceder, S. Curtarolo, *Meas. Sci. & Tech.* **16**, 296 (2005)
- [4] A. Jain, G. Hautier, C.J. Moore, S.P. Ong, C.C. Fischer, T. Mueller, K.A. Persson, G. Ceder, *Comp. Mater. Sci.* **50**, 2295 (2011)
- [5] Y. Wang, S. Curtarolo, C. Jiang, R. Arroyave, T. Wang, G. Ceder, L.Q. Chen, Z.K. Liu, *Calphad* **28**(1), 79 (2004)
- [6] J.E. Saal, S. Kirklin, M. Aykol, B. Meredig, C. Wolverton, *JOM* **65**, 1501 (2013)
- [7] C.E. Calderon, J.J. Plata, C. Toher, C. Oses, O. Levy, M. Fornari, A. Natan, M.J. Mehl, G. Hart, M.B. Nardelli, S. Curtarolo, *Comput. Mater. Sci.* **108**, 233 (2015)
- [8] A. van de Walle, R. Sun, Q.J. Hong, S. Kadkhodaei, *Calphad* **58**, 70 (2017)
- [9] A. van de Walle, J. Sabisch, A.M. Minor, M.D. Asta, *Journal of Materials Research* **34**, 3296 (2019)
- [10] A. van de Walle, M. Asta, *MRS Bull.* **44**, 252 (2019)
- [11] S. Samanta, A. van de Walle, *Calphad* **74**, 102306 (2021)
- [12] S. Zhu, A. van de Walle, *J. of Phase Equilibria* **42**, 315 (2021)
- [13] C. Bale, E. B  lisle, P. Chartrand, S. Decterov, G. Eriksson, A. Gheribi, K. Hack, I.H. Jung, Y.B. Kang, J. Melan  on, A. Pelton, S. Petersen, C. Robelin, J. Sangster, P. Spencer, M.A.V. Ende, *Calphad* **54**, 35 (2016)
- [14] W. Cao, S.L. Chen, F. Zhang, K. Wu, Y. Yang, Y. Chang, R. Schmid-Fetzer, W. Oates, *Calphad* **33**, 328 (2009)
- [15] J.O. Andersson, T. Helander, L. H  glund, P.F. Shi, B. Sundman, *Calphad* **26**, 273 (2002)
- [16] Z.K. Liu, *J. Phase Equilib. Diff.* **30**, 517 (2009)
- [17] Y. Wang, S. Shang, L.Q. Chen, Z.K. Liu, *JOM - J. Min. Met. Mat. S.* **65**, 1533 (2013)
- [18] C. Campbell, U. Kattner, Z.K. Liu, *Scripta Materialia* **70**, 7 (2014)
- [19] T. Hickel, U.R. Kattner, S.G. Fries, *Phys. Status Solidi B* **251**, 9 (2014)
- [20] C.E. Campbell, U.R. Kattner, Z.K. Liu, *Integrating Materials* **3**, 12 (2014)
- [21] NIST materials data repository. URL <https://materialsdata.nist.gov/>
- [22] A. van de Walle, C. Nataraj, Z.K. Liu, *Calphad* **61**, 173 (2018)
- [23] B. Sundman, U.R. Kattner, C. Sigli, M. Stratmann, R.L. Tellier, M. Palumbo, S.G. Fries, *Computational Materials Science* **125**, 188 (2016)
- [24] R.A. Otis, Z.K. Liu, *J. Open Res. Softw.* **5**, 1 (2017)
- [25] H. Lukas, E. Henig, B. Zimmermann, *Calphad* **1**, 225 (1977)
- [26] Q. Du, M. Wells, *Computational Materials Science* **50**, 3153 (2011)
- [27] H. Gaye, C. Lupis, *Metallurgical Transactions A* **6**, 1057 (1975)

- [28] A. van de Walle, M.D. Asta, *Model. Simul. Mater. Sc.* **10**, 521 (2002)
- [29] R. Otis, M. Emelianenko, Z.K. Liu, *Comput. Mater. Sci.* **130**, 282 (2017)
- [30] M.D. McKay, W.J.C. R. J. Beckman, *Technometrics* **21**, 239 (1979)
- [31] A. van de Walle, Q.J. Hong, *J. Phase Equilib. Diff.* **40**, 170 (2019)
- [32] A. Okabe, B.B. abd K., Sugihara, *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams* (Wiley, New York, 1992)
- [33] The CGAL Project, *CGAL User and Reference Manual*, 5.3.1 edn. (CGAL Editorial Board, 2021). URL <https://doc.cgal.org/5.3.1/Manual/packages.html>
- [34] A. van de Walle, M.D. Asta, G. Ceder, *Calphad* **26**, 539 (2002)
- [35] A. van de Walle, *Calphad* **33**, 266 (2009)
- [36] B. Sundman, U.R. Kattner, M. Palumbo, S.G. Fries, *Integrating Materials and Manufacturing Innovation* **4**, 1 (2015)
- [37] L.S. Avila, U. Ayachit, S. Barré, et al., *The VTK User's Guide* (Kitware, USA, 2010). URL <https://www.vtk.org/vtk-users-guide/>
- [38] U. Ayachit, *The ParaView Guide: A Parallel Visualization Application* (Kitware, USA, 2015)
- [39] W.M. Choi, Y.H. Jo, D.G. Kim, S.S. Sohn, S. Lee, B.J. Lee, *Calphad* **66**, 101624 (2019)
- [40] T. Kirk, B. Vela, S. Mehalic, K. Youssef, R. Arróyave, *Scripta Materialia* **208**, 114336 (2022)
- [41] A. Zunger, S.H. Wei, L.G. Ferreira, J.E. Bernard, *Phys. Rev. Lett.* **65**(3), 353 (1990)
- [42] A. van de Walle, P. Tiwary, M.M. de Jong, D.L. Olmsted, M.D. Asta, A. Dick, D. Shin, Y. Wang, L.Q. Chen, Z.K. Liu, *Calphad* **42**, 13 (2013)
- [43] R. Kikuchi, *Phys. Rev.* **81**, 988 (1951)
- [44] G. Kresse, J. Furthmüller, *Comp. Mater. Sci.* **6**, 15 (1996)
- [45] G. Kresse, D. Joubert, *Phys. Rev. B* **59**, 1758 (1999)
- [46] P.E. Blöchl, *Phys. Rev. B* **50**, 17953 (1994)
- [47] J.P. Perdew, K. Burke, M. Ernzerhof, *Phys. Rev. Lett.* **77**, 3865 (1996)
- [48] A. van de Walle, G. Ceder, *J. Phase Equilib.* **23**, 348 (2002)
- [49] P.E. Blöchl, O. Jepsen, O.K. Andersen, *Phys. Rev. B* **49**, 16223 (1994)

A Appendix: Cross-section calculation algorithm

This section describes in more detail how to compute a c -dimensional cross section of an $(n - 1)$ -dimensional simplex.

An $(n - 1)$ -dimensional simplex can be defined as the set of all convex combinations of its n vertices (a *convex combination* is a linear combination with positive weights that sum to one). Similarly, its $(n - s)$ -dimensional facets (for some $s > 1$) are defined by convex combination of $n - s + 1$ of its vertices. (To fix the ideas, 2-dimensional facets correspond to the usual facets, while the 1-dimensional facets correspond to edges.)

These $(n - s)$ -dimensional facets are mapped onto lower dimensional objects by the cross-section operation. Our task is to find the s such that the $(n - s)$ -dimensional facets will be mapped to single points by

a c -dimensional cross-section operation. These points then correspond to the vertices of the cross-section. These vertices, once connected by a $(c - 1)$ -dimensional mesh, yield the cross-section of interest.

A c -dimensional cross-section is defined by $n - c$ linear constraints and we thus need $n - c$ degrees of freedom to be able to uniquely satisfy these constraints (i.e. obtain a single point). A convex combination of $n - s + 1$ vertices of the simplex has $n - s$ degrees of freedom (because the weights must sum to one), and we conclude that $s = c$.

The algorithm is then simply:

1. Consider every group of $n - c + 1$ vertices out of the n vertices of the original simplex.
2. For each group, solve for which linear combination (with weights summing to one) of these vertices yields a point p within the given c -dimensional cross-section hyper-plane.
3. If the weights are all positive, place the point p in the list of vertices.
4. Mesh the resulting vertices to form a $(c - 1)$ -dimensional manifold (which is a flat manifold bounded by a convex boundary).

B Ab Initio Calculation Parameters

The ab initio calculations were performed with the VASP code [44, 45], which implements the Projector Augmented Wave (PAW) [46]. The pseudopotentials used had the following number of electrons included as valence: Cr: 6, Mo: 6, Nb: 11, V: 5, W: 6. The PBE [47] exchange-correlation functional was used. A kinetic energy cutoff of 300 eV and k -point mesh of density of at least 8000 points per reciprocal atom was automatically generated [48] for each structure and Fermi level smearing of 0.1 eV was performed using the Methfessel-Paxton scheme of order 1. All relaxation calculations allowed all cell parameters and atomic coordinates to vary. They were followed by a static run, where Brillouin zone integration were performed using the tetrahedron method with Blöchl corrections [49].

The Calphad model was built using Special Quasirandom Structure (SQS) structural energies on composition grid corresponding to “level 4” in `sqs2tdb` [8], this including compositions of the form A, $A_{1/2}B_{1/2}$, $A_{3/4}B_{1/4}$, $A_{1/3}B_{1/3}C_{1/3}$, $A_{1/2}B_{1/4}C_{1/4}$ with A,B,C standing for any 3-element subset of the system’s 5 elements. The interaction coefficients of the Calphad model include polynomials of binary and ternary compositions up to order 3. The cross-validation score of the fit was 11 meV. Short-range-order corrections were included.