Five Degree-of-Freedom Property Interpolation of Arbitrary Grain Boundaries via Voronoi Fundamental Zone Octonion Framework

Sterling G. Baird[a], Eric R. Homer[a], David T. Fullwood[a], Oliver K. Johnson[a,*]

[a]*Department of Mechanical Engineering, Brigham Young University, Provo, UT 84602, USA*

Abstract

In this work we introduce the Voronoi Fundamental Zone octonion (VFZO) interpolation framework for grain boundary (GB) structure-property models and surrogates. The VFZO framework offers an advantage over other five degree-of-freedom (5DOF) based property interpolation methods because it is constructed as a Voronoi Fundamental Zone (VFZ) point set in a Riemannian manifold, for which directly computed Euclidean and arc length distances significantly reduce computation time compared with other approaches. This increased efficiency facilitates the use of significantly more input data and therefore leads to a reduction of the interpolation error. As a demonstration, we present grain boundary energy (GBE) interpolation results for a non-smooth validation function (Bulatov et al. [1]) and a simulated bi-crystal dataset for Fe (Kim et al. [2]). Four interpolation methods built on this framework — barycentric interpolation, Gaussian process regression (GPR) or Kriging, inverse-distance weighting (IDW), and nearest neighbor (NN) interpolation — are presented for the validation function and compared against a constant, average model (root mean square error (RMSE) $\simeq 0.1283\,\mathrm{J\,m^{-2}}$), resulting in RMSE values of 0.0238, 0.0218, 0.0356, and $0.0445\,\mathrm{J\,m^{-2}}$, respectively, for 50 000 randomly sampled input GBs and evaluated for 10 000 randomly sampled prediction GBs. A vectorized, parallelized, MATLAB interpolation function (`interp5DOF.m`) and related routines is made available in our VFZO repository (`github.com/sgbaird-5dof/interp`). A GPR mixture model is developed and applied for the Fe simulation dataset, yielding a simulation-based interpolation function for Fe GBE with quantified uncertainty. The VFZO framework offers advantages for computing distances between GBs, estimating property values for arbitrary GBs, and modeling surrogates of computationally expensive 5DOF functions and simulations.

Keywords: Grain Boundary, Structure-Property Model, Interpolation, Octonion, Energy

1. Introduction

In this work, we present a new method for interpolation and prediction of grain boundary (GB) properties from a set of measured/calculated values. Our approach, called the Voronoi Fundamental Zone octonion (VFZO) framework is highly efficient, and thereby facilitates the use of large data sets to enhance prediction accuracy.

1.1. Previous Work

In previous work, a number of strategies have been developed for predicting five degree-of-freedom (5DOF) grain boundary (GB) properties from experimental or simulated data. Binning and gradient descent were used to produce a 5DOF grain boundary energy distribution (GBED) in nickel [3], yttria [4], and copper [5] based on experimentally characterized 3D microstructures.

Restrepo et al. [6] used an artificial neural network (ANN) and approximately 17 000 and 51 000 Fe bicrystal simulations from Kim et al. [7] as train-

*Corresponding author.

Email address: `ojohnson@byu.edu` (O.K. Johnson).

ing and validation data, respectively, to achieve mean absolute errors (MAEs) of 0.0486 J m^{-2} and approximately 0.09 J m^{-2} in the best fitted ANNs for randomly selected and special GBs, respectively.

Recently, a new GB representation, grain boundary octonions (GBOs), was reported [8] and tested [9]. The GBO representation is valuable for a number of applications. Most relevant to the present work is the resulting distance metric. The GBO distance metric offers an advantage over other metrics in that it "correctly determines the angular distances between GBs with a common normal or misorientation" and "closely approximates the geodesic metric on $SO(3) \times SO(3)$ *for all grain boundary pairs* while maintaining the ability to be analytically minimized with respect to the $U(1)$ symmetry" [8]. In this context, Francis et al. [8] derived octonion Spherical Linear Interpolation (oSLERP) and provided examples showing that oSLERP produces smooth, minimum distance paths through GB character space between two arbitrary GBs.

Laplacian kernel regression (LKR) (a type of inverse-distance weighting (IDW)) involving scaled pairwise distance matrices was later used with GBOs to predict properties of arbitrary GBs from a set of known values [9]. Using k-fold cross validation (kFCV) with $k = 10$ for 388 Ni grain boundary energy (GBE) simulations [10] and an optimized scaling parameter, an root mean square error (RMSE) of 0.0977 J m^{-2} was obtained. Due to computation time of pairwise distance matrices, this approach is currently "limited to datasets with several thousand or fewer" GBs [9].

1.2. *Voronoi Fundamental Zone Octonion Framework*

The VFZO interpolation framework introduced in this work offers an advantage over other methods because it is defined as a Voronoi Fundamental Zone (VFZ) point set in a Riemannian manifold. This advantage is manifest in the ability to triangulate a mesh using standard routines (e.g. quickhull [11]) and interpolate using barycentric coordinates or machine learning methods such as Gaussian process regression (GPR). Building on previous work on GBOs [8, 9], we create a VFZ point set by obtaining a set of octonions minimized with respect to Euclidean distance and an arbitrary reference octonion after considering all symmetrically equivalent octonions (SEOs). Because GBOs are guaranteed to reside on the surface of a hypersphere [8] (a type of Riemannian manifold) a point set which locally resembles Euclidean space is the result (Section 2.1.3). Below we mention a few benefits and applications of this approach, after which we provide the detailed description of the method (Section 2) and numerical test results (Section 3).

1.3. *Benefits of Voronoi Fundamental Zone Octonion Framework*

1.3.1. *Distance Calculations*

Directly computed, scaled Euclidean and arc length distances in the VFZO framework approximate the original octonion distance by Francis et al. [8], and the calculation speed is even higher than explicit GBO distance calculations using the original octonion distance. For example, 50 000 GBOs can by symmetrized into VFZOs in approximately 76 seconds using 6 cores (e.g. via VFZO repository function `get_octpairs.m`), and the corresponding $50\,000 \times 50\,000$ pairwise-distance matrix can be computed in approximately 10 seconds using the built-in MATLAB *Statistics and Machine Learning Toolbox* function `pdist.m`, giving a total runtime of approximately 86 seconds. Compared to the original octonion metric distance calculations [9] in the Fortran-based EMSoft package [12], this represents an improvement in computational speed by five orders of magnitude using our MATLAB implementation [13] which we refer to as the VFZO repository.[1]

This significant speed up stems from the fact that in the VFZO framework SEOs only need to be considered once per GB, $O(L)$, rather than once per distance calculation, $O(L^2)$, and that SEOs only need to be considered once in a GB pair, $O(N_p^2)$,

[1]Improvement per distance calculation per core of the VFZO repository is about 4×10^5 relative to the EMSoft [12] metric of 26 minutes using 8 cores for a 388 x 388 pairwise distance matrix. This EMSoft timing information is directly reported in [9].

rather than for every combination between the two GBs, $O(N_p^4)$. The SEO computation complexity is thus $O(N_p^2 L)$, a significant improvement compared with the original SEO complexity of $O(N_p^4 L^2)$ [9], where N_p is the cardinality of the crystallographic point group ($N_p = 24$ for $m\bar{3}m$ face-centered cubic (FCC) point group) and L is the number of GBs.

Empirically, to compute a pairwise-distance matrix for $L = 50\,000$ GBs using the VFZO repository [13], the full $O(N_p^2 L)$ symmetrization operations take about $76 \times 6 = 456$ seconds of CPU time, whereas the subsequent pairwise-distance computation is $O_{\text{pd}}(L^2)$ and takes approximately 10 seconds for a $50\,000 \times 50\,000$ matrix, despite $N_p^2 L \ll L^2$. In other words, $O(N_p^2 L) + O_{\text{pd}}(L^2) \approx O(N_p^2 L)$ because the operation of considering an SEO is computationally much more expensive than the direct (Euclidean) pairwise distance computation in this work. Because Euclidean distances are employed—which can be computed faster than trigonometric inverse functions—and built-in, vectorized MATLAB functions are utilized where possible, there is a further speed enhancement in the VFZO approach.

1.3.2. Interpolation Error

The ability to use significantly more input data lends itself to lower interpolation error. This will be demonstrated by GBE interpolation results for a non-smooth validation function in Section 3. Four interpolation methods built on the VFZO framework — barycentric (Section 2.3.1), GPR or Kriging (Section 2.3.2), IDW (Section 2.3.3), nearest neighbor (NN) (Section 2.3.4) — will be presented and compared against a constant, average model (Section 3.1). To facilitate easy application of the presented method, a vectorized, parallelized, MATLAB implementation, `interp5DOF.m`, is made available in the VFZO repository [13] with similar input/output structure to that of built-in MATLAB interpolation functions (e.g. `scatteredInterpolant()`, `griddatan()`). A typical function call is as follows: `ypred = interp5DOF(qm,nA,y,qm2,nA2,method)`. The argument `y` is a vector of known property values corresponding to the GBs defined by (`qm`,`nA`), which respectively denote pairs of GB misorientation quaternions and boundary plane (BP) normals. The result, `ypred`, is a vector of predicted/interpolated property values corresponding to the prediction GBs defined by (`qm2`,`nA2`).

Internally, these are converted to octonions and interpolation is performed using the selected `method`. The methods used in this work are `'pbary'`, `'gpr'`, `'idw'`, and `'nn'`, corresponding to planar barycentric, GPR, IDW, and NN interpolation, respectively. A placeholder template with instructions for implementing additional interpolation schemes is also provided in `interp5DOF.m`. Misorientation quaternions are represented in the active sense[2]:

$$q_m = {q_A}^{-1} q_B \tag{1}$$

where q_m, q_A, and q_B represent the misorientation quaternion, orientation quaternion of grain A in the sample frame, and orientation quaternion of grain B in the sample frame, respectively. The $^{-1}$ operator denotes a unit quaternion inverse (identical to conjugation of a unit quaternion). Quaternion multiplication is given by equation 23 of [14] with $P = 1$. BP unit normals are expressed pointing away from grain A and in the reference frame of grain A (i.e. the outward-pointing normal convention). See [8] and `five2oct.m` [13] for a treatment of conversions to octonion coordinates.

1.3.3. Applications

The VFZO framework offers a great advantage in estimating property values for arbitrary GBs based on experimental or simulated data such as energy, mobility, and diffusivity. The framework can also enable efficient surrogate modeling of computationally expensive 5DOF functions and simulations such as in evaluation of the Bulatov Reed Kumar (BRK) function [1] for use in anisotropic grain growth simulations. In other words, one can evaluate the VFZO surrogate model in a fraction of the time of the true property model, thereby facilitating larger scale iterative simulations, which require repetitive evaluation of a computationally expensive structure-property model.

[2]The passive convention is used in [8]

2. Methods

The core operations of the VFZO framework are (i) the generation of, and (ii) mapping of points into, a VFZ, and (iii) distance calculations within the VFZ. Rather than defining a 5DOF GB fundamental zone (FZ) using linear inequalities[3] which is the traditional approach that has been applied to other crystallographic spaces (e.g. the orientation [15] and misorientation [15, 16] FZs), we employ a numerical strategy to define what we will refer to as a VFZ.

The construction of the VFZ dramatically reduces the computational burden of pairwise distance calculations. The mechanism by which this reduction is achieved can best be illustrated with an example. Let o_1 and o_2 denote two GBs represented as GBOs. To perform a traditional distance calculation it is necessary to compare all SEOs of o_1 to all of the SEOs of o_2 and take the smallest distance. If N_p is the cardinality of the crystallographic point group, this single minimum distance calculation requires a total of N_p^4 SEOs to be considered. If one desires to compute a pairwise distance matrix between L GBs, all SEOs of each GB are compared against all SEOs of all others so that the total number of SEO computations will be $O(N_p^4 L^2)$.

In contrast, for a single distance calculation using the VFZO framework, o_1 and o_2 are first mapped into the VFZ, and then only a single distance calculation is required between them. Mapping o_1 into the VFZ requires comparison of all N_p^2 SEOs of o_1 with a fixed reference GB in the interior of the VFZ; and likewise for o_2. Consequently, a single distance calculation between o_1 and o_2 under the VFZO framework requires $2N_p^2$ SEO computations. If one desires to compute a pairwise distance matrix between L GBs, the total computational cost will be[4] $O(N_p^2 L)$, which represents a dramatic reduction compared to the traditional approach (Table 5). A summary of the differences between the two approaches is provided in Table 1.

In this section we describe our methods for each of the VFZO framework operations (i)-(iii) named above. We then describe our implementation of 4 different interpolation schemes for prediction of GB properties based on the VFZO approach.

2.1. The Voronoi Fundamental Zone Octonion Framework

2.1.1. Defining the Voronoi Fundamental Zone

To define a VFZ, an arbitrary, fixed, low-symmetry reference GBO is chosen (o_{ref}) and the VFZ is formally defined as the region of $\mathbb{S}^7$ (the unit 7-sphere in 8 dimensions) closer to o_{ref} than any of its symmetric images. However, use of the VFZ does not require its explicit construction. Rather, practical calculations require only the selection of the single point o_{ref} (which completes the definition of the VFZ), followed by mapping of query points into the VFZ by comparison of their SEOs with o_{ref}.

To illustrate the process of mapping points into the VFZ, we describe a 3D Cartesian analogue (Figure 1) to a 7D Cartesian non-degenerate (i.e. U(1) degeneracy removed) representation of a VFZ. A set of 500 points ($p_i, i \in [1, 500]$) randomly scattered on the surface of the 2-sphere comprise the data (red points) (a). A random point, p_{ref}, also on the surface of the 2-sphere, is chosen as the reference point (white circle). In this illustration, O_h or $m\bar{3}m$ point group rotations are used as symmetry operators, S_j, $j \in [1, N_p]$, where N_p is the cardinality as before and $N_p = 24$ for the O_h point group. For each data point, 24 symmetrically equivalent representations ($p_{i,j}^{\text{sym}} = S_j(p_i)$, $j \in [1, 24]$) are produced by applying each of the relevant symmetry operators. After calculating the Euclidean distance between p_{ref} and $p_{i,j}^{\text{sym}}$, the point (p_i^*) closest to p_{ref} is chosen and retained as the unique representative of $p_{i,j}^{\text{sym}}$. As illustrated in Figure 1(a), the projected

[3]If desired, linear inequalities can be obtained for a VFZ by determining a Voronoi tessellation's junction points (similar to what is shown in Figure 1 by e.g. `voronoin()`), transforming to 6D Cartesian coordinates via a singular value decomposition (SVD) transformation (Appendix A) and defining the bounded region by e.g. MATLAB FEX function `vert2lcon.m`.

[4]See Section 1.3.1 for a detailed explanation of why this is *not* $O(N_p^2 L^2)$.

Table 1: Comparison between Voronoi Fundamental Zone octonion and traditional octonion frameworks. *6D Cartesian representation used only for mesh triangulation efficiency in barycentric interpolation and *7D Cartesian representation only required for barycentric interpolation. 7D Cartesian representation is also implemented (though not required) for GPR, NN, and IDW. For pairwise distance complexity, N_p is the symmetry cardinality ($N_p = 24$ for $m\bar{3}m$ FCC point group) and L is the number of GBs.

Property	Traditional	This Work
Symmetrizing Distance	Octonion	Euclidean
Dimensionality	8D Cartesian	6*/7*/8D Cartesian
Bounded by FZ	No	Yes
Pairwise Distance Complexity	$O(N_p^4 L^2)$	$O(N_p^2 L)$
Rotation Convention	Passive	Active

points p_i^* (dark blue points) all fall in the VFZ without ever having to construct or define it explicitly, we call this group of projected points a *VFZ point set*. Note also that there is only one p_i^* in the VFZ for each $p_{i,j}^{\mathrm{sym}}$ (see Figure 1(b)).

To calculate the distance between a given octonion, and the reference octonion, we employ the standard 8D Euclidean distance

$$d_{\mathrm{E}}(o_A, o_B) = \left(\sum_{k=1}^{8} (o_{A,k} - o_{B,k})^2 \right)^{1/2} \qquad (2)$$

where $o_{X,k}$ is the k-the element of octonion o_X, and o_X is normalized. Euclidean distance is an approximation to the true geodesic arc length on $\mathbb{S}^7$, which is given by

$$d_{\mathrm{S}}(o_A, o_B) = \cos^{-1}(o_A \cdot o_B) \qquad (3)$$

where $\cdot$ is the dot product, $\cos^{-1}$ is the inverse cosine operator, and o_A and o_B are each normalized. In [8], the original octonion distance metric can be defined by

$$d_{\Omega}(o_A, o_B) = 2\cos^{-1}(o_A \cdot o_B) \qquad (4)$$

where o_A and o_B are each normalized, where d_Ω can be seen to be simply twice the geodesic arc length: $d_\Omega = 2d_{\mathrm{S}}$. The definition of d_Ω has certain aesthetic benefits in that it mirrors the definition of a misorientation angle, ω_{AB}, between two crystal orientations in the quaternion parameterization: $\omega_{AB} = 2\cos^{-1}(q_A \cdot q_B)$.

Our choice to use d_{E} instead of d_{S} or d_Ω is motivated by the fact that it enables the use of standard algorithms, for a variety of operations, that require or assume Euclidean distances. In addition to enabling us to leverage the machinery of efficient and established algorithms, this choice can be justified by the following observations:

- The minimum Euclidean distance SEO will be the same as the minimum arc length distance SEO because d_{S} is a monotonically increasing function of d_{E}, for $d_{\mathrm{S}}(d_{\mathrm{E}}) \in [0, \pi]$ (Figure S1).
- For the FCC point group symmetry ($m\bar{3}m$) the portion of $\mathbb{S}^7$ subtended by the VFZ is sufficiently small that the approximation $d_{\mathrm{E}} \simeq d_{\mathrm{S}}$ holds to very high accuracy[5] as shown in Figure S1.
- Calculation of d_{E} does not require the use of any inverse trigonometric functions and is about 23 % faster than calculation of d_{S} or d_Ω.

2.1.2. Mapping GBs to the Voronoi Fundamental Zone

As described above in the 3D analogy, with a reference GBO chosen (o_{ref}), and consequently the VFZ defined (Section 2.1.1), a GBO is mapped into the VFZ by finding among its SEOs the one that is closest to o_{ref} according to d_{E} (Eq. (2)). This is

[5]This is true when calculating the actual distance between two points in the VFZ without regard to symmetric images. When calculating the minimum distance between SEOs of two points, there are additional considerations that must be attended to as discussed in detail in Section 2.1.3.

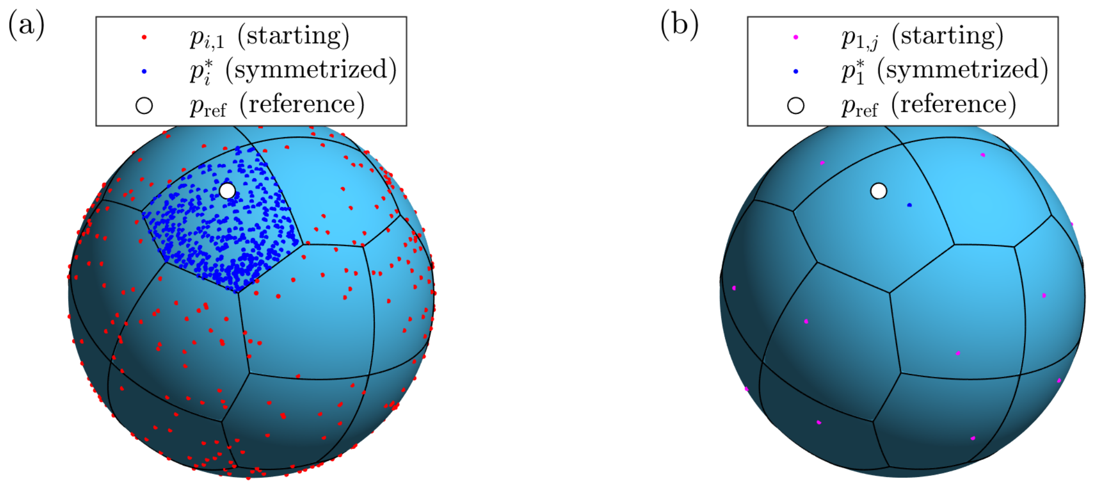

Figure 1: (a) 3D Cartesian analogue to a non-degenerate 7D Cartesian representation of U(1)-symmetrized GBOs and VFZOs (VFZOs are inherently U(1)-symmetrized) which demonstrates the symmetrization of many points relative to a fixed, reference point (white circle). This produces a 3D Cartesian VFZ point set (dark blue points). (b) To further illustrate, a single input point (magenta points) is symmetrized (dark blue point) relative to a fixed, reference point (white circle) demonstrating that only one symmetrized point is found within the borders (black) of each of the Voronoi cells (light blue). The Voronoi tessellation is defined by the symmetric images of the reference point, and the spherical Voronoi diagram for this illustration is constructed using a modified version of [17].

performed for all input and prediction points with respect to o_{ref}, and the result is a VFZ point set.

2.1.3. Distance Calculations in the Voronoi Fundamental Zone

Euclidean distances are an accurate approximation of arc length distances in a VFZ because the difference between the two metrics for the maximum pairwise distance ($pd_{max} \simeq 60°$) in a VFZ is small as shown in Figure S1. However, when compared with the traditional octonion distance [8], due to the presence of low-symmetry GBs near the exterior of a VFZ, some GB pairs will exhibit larger Euclidean or arc length distances than is truly representative (Figure 2a). In other words, moving "past" the low-symmetry border of a VFZ will result in an instantaneous relocation to a possibly distant point in the VFZ that in reality is highly correlated. VFZO Euclidean, hyperspherical arc length, and octonion distances are computed via VFZO repository function `GBdist4.m` which is used in the symmetrization function `get_octpairs.m`.

This is a limitation of the VFZO framework which generates a VFZ with low-symmetry GBs at the borders in contrast to typical FZs [18, 19]. While defining a FZ with high-symmetry GBs at the borders (especially mirror-symmetry GBs) will certainly increase interpolation accuracy, the favorable interpolation results presented in this work are obtained because overestimation is infrequent within a small correlation length (e.g. 10° [20], for which many NNs fall within for a 50 000 VFZO set, see Figure 5b), and underestimation is non-existent within numerical precision.

Overestimation imposes an arbitrary "sparseness" of data within a local region of influence common to the interpolation methods in this work, whereas underestimation would give erroneous high correlations between uncorrelated GBs. Because only overestimation relative to traditional octonion distances exist in this work (Figure 2), it is expected that large errors will occur infrequently, as shown in Figure 6, and that overall errors will remain low as shown in Figure 7, and Tables 2 and 3.

While distance calculations are subject to these infrequent overestimates, they are largely immaterial for interpolation. This is because all of the interpolation methods in this work involve a region of influence that is small, so that if the distance to a NN is overestimated it simply does not contribute to the interpolation. Consequently the accuracy of the interpolation is not significantly impacted by infrequent distance overestimates, and excellent results can be achieved without addressing this limitation. However, if even greater accuracy is desired it can be obtained for a relatively minor cost by considering multiple VFZs.

We find that taking the minimum distance among several VFZO sets defined by separate reference octonions leads to better correlation between the Euclidean approximation and the traditional octonion metric (Figure 2). Additionally, Figure 3 shows that the error between scaled Euclidean distance and the traditional octonion metric decreases rapidly as the number of ensemble VFZO sets increases. This confirms that employing a small ensemble of VFZO sets results in significant improvement to the Euclidean distance approximation (Figures 2 and 3) of the traditional octonion metric. However, as already mentioned, improvements to interpolation results are expected to be less significant since they are already robust to occasional distance overestimates.

Incorporating ensemble VFZO sets with interpolation methods increases overall accuracy at a computational cost (e.g. approximately 10x using 10 ensembled VFZO sets). We expect these modest overall accuracy improvements occur because GBE predictions near the exterior of the VFZ where data may be arbitrarily sparse are improved. Ensemble interpolation results as a function of ensemble size and parity plots for mean, median, minimum, and maximum homogenization functions are shown in Figure S3 and Figure S2, respectively. Further details of ensemble interpolation are given in Section S2.

2.2. Generating Random Voronoi Fundamental Zone Octonions

In addition to the 3 core operations of the VFZO framework described in Section 2.1, it will be necessary for our tests, and useful for other applications, to be able to generate points in the VFZ. We briefly explain here our process for accomplishing this.

First, random GBOs are formed by taking random misorientation quaternion (`qm`) and BP normal (`nA`) pairs. Random misorientation quaternions are obtained via cubochoric sampling [21] (`get_cubo.m`) and random BP vectors are sampled from a multivariate Gaussian distribution ($\mu = 0$, $\sigma = 1$) in $\mathbb{R}^3$ and normalized[6]. After this, they are converted to GBOs via VFZO repository function `five2oct.m`. The VFZO repository function `get_five.m` returns the result of these several operations (see also VFZO repository function `get_ocubo.m` for generating random GBOs directly).

These (`qm`,`nA`) pairs are then converted to an octonion representation, `o`, using VFZO repository function `o=five2oct(qm,nA)` and symmetrized via `osym=get_octpairs(o)`. A default reference octonion[7] is used for these calculations, unless specified by the user. The conventions used for `qm` and `nA` are given in Section 1.3.2.

For the present work we use this procedure to generate random VFZO sets containing between 100 to 50 000 VFZOs. The average NN distance (over approximately 70 trials) of such sets ranges between $(10.7175 \pm 0.3684)^\circ$ and $(2.6479 \pm 0.2254)^\circ$, respectively. Figure 4 illustrates how the VFZO average NN distance varies with the cardinality of the set (i.e. number of random VFZOs in the set).

For a specific 50 000 VFZO set, the NN octonion distance is $(2.870\,90 \pm 0.691\,12)^\circ$ (Figure 5a) while the average 100-th NN distance is within 10° (Figure 5b). This indicates that, on average, prediction VFZOs fall within a typical GB correlation length (10° [20]) of input VFZOs in large set sizes.

2.3. Interpolation in the Voronoi Fundamental Zone Octonion Framework

With the VFZO framework established, it is possible to define interpolation schemes over the VFZ to predict the properties of new GBs from the known properties of other GBs. For one application of interest to us, it is necessary to evaluate

[6]Methods for uniform sampling of points on a sphere can be obtained by methods such as those described in https://mathworld.wolfram.com/SpherePointPicking.html.

[7]This is generated by `get_ocubo.m` using a random number generator seed of 10. We expect that `five2oct.m` combined with `get_five.m` will generate near identical statistical properties to `get_ocubo.m` which is supported by a visual comparison of pairwise distance histograms (not shown in this work).

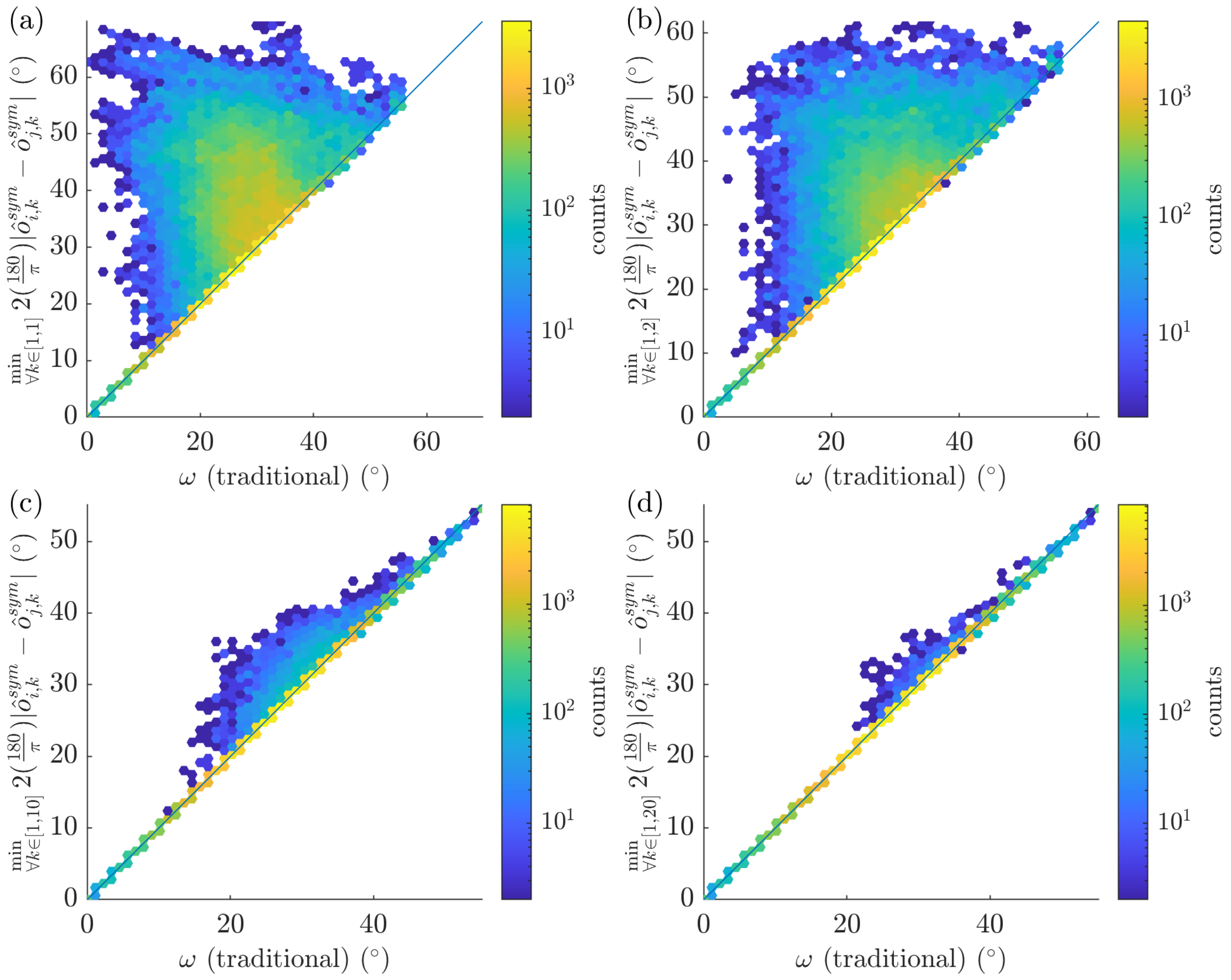

Figure 2: Hexagonally binned parity plots of pairwise distances of 388 Ni bicrystals [10]. Euclidean distance approximation is converted to octonions ($x_{i,j,k} = 2\left(\frac{180}{\pi}\right)|\hat{o}^{\text{sym}}_{i,k} - \hat{o}^{\text{sym}}_{j,k}|$) for comparison with the traditional octonion metric [9]. The minimum distance among an ensemble of VFZO sets ($\min_{\forall k\in[1,k_{max}]} x_{i,j,k}$) is used for (a) 1, (b) 2, (c) 10, and (d) 20 VFZO sets. As the number of VFZO sets increases, the correlation between the Euclidean distance and the traditional octonion distance improves.

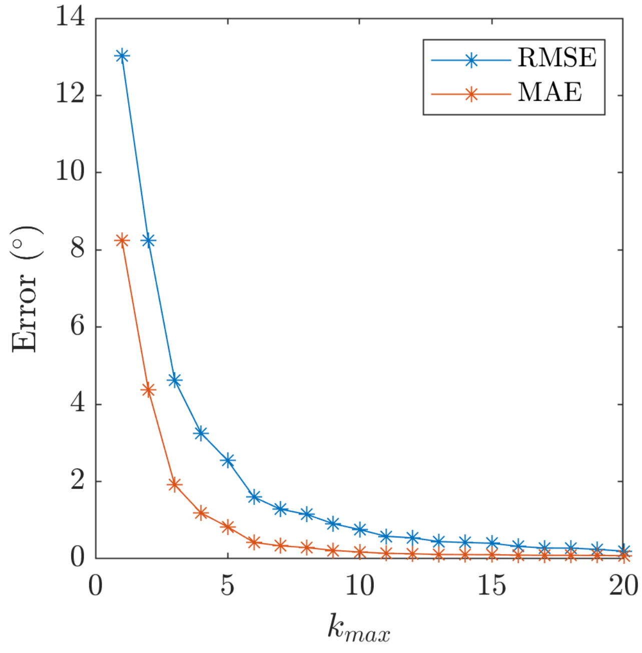

Figure 3: RMSE and MAE of pairwise distance errors for 388 Ni bicrystals [10] of scaled Euclidean distance approximation relative to the traditional octonion metric [9] (compare with Figure 2). The minimum distance among an ensemble of VFZO sets ($\min_{\forall k \in [1, k_{max}]} x_{i,j,k}$, where $x_{i,j,k}$ is the scaled Euclidean distance) is taken, iteratively adding consecutive sets up to $k_{max} = 20$. As the number of VFZO sets increases, RMSE and MAE between the scaled Euclidean distance approximation and the traditional octonion distance decreases.

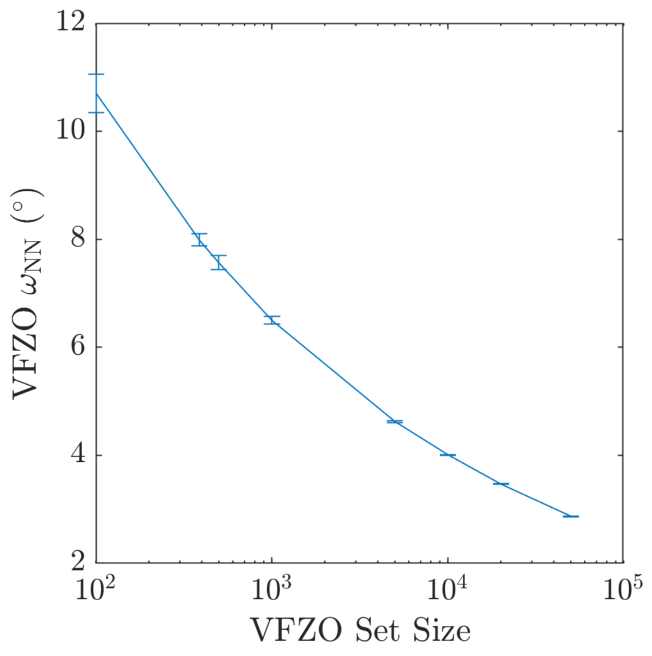

Figure 4: NN VFZO (ω_{NN}) distances (°) versus VFZO set size out of 70-80 random VFZO sets per set size.

multiple different functions over a fixed set of input and prediction GBOs. In this section we first present a barycentric interpolation method that we have developed to efficiently accomplish this task by pre-computing the interpolation weights (which remain fixed only when the function being evaluated changes). We then present adaptations of three other interpolation methods (GPR, IDW, and NN) that are useful for other applications. We recommend the GPR interpolation method for the VFZO framework for most applications because it provides the best combination of accuracy and speed (Section 3).

2.3.1. Barycentric Interpolation

Barycentric coordinates are a type of homogeneous coordinate system that reference a prediction point within a simplex [22] or convex polytope [22–24] based on "masses" or weights at the vertices, which can be negative. The prediction point is assumed to be the barycenter (center of mass) of the simplex or convex polytope, and weights at the vertices necessary to make this assumption true are determined. We utilize rigid SVD transformations and a standard triangulation algorithm (quickhull

[11] via `delaunayn()` in VFZO repository function `sphconvhulln.m`) to define a simplicial mesh (Appendix A.1). We then use barycentric weights (i.e. coordinates) for computing intersections of a point within a simplicial facet (Appendix A.2) and for interpolation (Appendix A.3) [22]. A detailed explanation of the process is provided in Appendix A. The barycentric interpolation method is invoked in `interp5DOF.m` by setting the `method` argument to `'pbary'`.

2.3.2. Gaussian Process Regression

GPR or Kriging uses the notion of similarity between points to fit Gaussian processes (random variables) to data based on prior information and provides uncertainty information in addition to interpolated or inferred values. For a general treatment of GPR, see [25]. We use MATLAB's built-in function, `fitrgp()`, with all default parameters (MATLAB R2020b was used for the Fe simulation dataset, all other results employed MATLAB R2019b) except that `PredictMethod = 'fic'` was used regardless of the number of input points, and we assume a Euclidean approximation of the VFZ (see Section 2.1.3 and Figure S1). A slower, more accurate, and more memory-intensive `PredictMethod = 'exact'` parameter choice is also available (Section 3.2). The GPR interpolation method is invoked in `interp5DOF.m` by setting the `method` argument to `'gpr'`.

2.3.3. Inverse-distance Weighting Interpolation

IDW interpolation applies a weighted average to points within a neighborhood of a query point to obtain an interpolated value. `interp5DOF.m` implements a simple IDW approach based on [26]. A default radius of influence of $r = \sqrt{2}\mu$ is used, where μ represents the mean NN distance, and where octonion distance is approximated by the Euclidean distance or 2-norm (see Section 2.1.3, and Figure S1). NN interpolation (Section 2.3.4) is used for a given query point when there are no input points in the radius of influence. The IDW interpolation method is invoked in `interp5DOF.m` by setting the `method` argument to `'idw'`.

2.3.4. Nearest Neighbor Interpolation

NN interpolation takes the nearest input point relative to a query point and assigns the value of the NN input point to the query point. This is implemented via the built-in MATLAB function `dsearchn()` using a Euclidean approximation of octonion distance (see Section 2.1.3, and Figure S1). The NN interpolation method is invoked in `interp5DOF.m` by setting the `method` argument to `'nn'`.

3. Results and Discussion

To illustrate the utility of the VFZO framework for one application, namely interpolation, we compare the (i) accuracy (Section 3.1), and (ii) efficiency (Section 3.2) of the four previously described interpolation methods implemented over the VFZ with each other and with existing methods from the literature (see Section 1). We then demonstrate VFZO GPR interpolation applied to a large Fe bicrystal simulation dataset [2] (Section 3.3) which provides both GBE and GBE uncertainty predictions.

For these tests, we use the 5DOF GB energy function developed by Bulatov, Reed and Kumar [1] as a validation function which we refer to as the BRK function. For a given trial run, an input VFZO set, `o`, and a prediction VFZO set, `o2`, are randomly generated according to Section 2.2. The validation function is evaluated at each of these points and the values are stored in the vectors `y` and `ytrue`, respectively. `o` and `o2` are SVD transformed together into a 7D Cartesian representation, which is an optional step for GPR, NN, and IDW, but not for barycentric interpolation. Finally, we use `interp5DOF.m` [13] to predict the function value at the prediction points, which are stored in the vector `ypred`. We repeat this process for each of the interpolation methods approximately 10 times per set size, and compare the predictions, `ypred`, to the true values, `ytrue`.

3.1. Interpolation Accuracy

Figure 6 provides hexagonally binned parity plots (via VFZO repository function `parityplot.m`

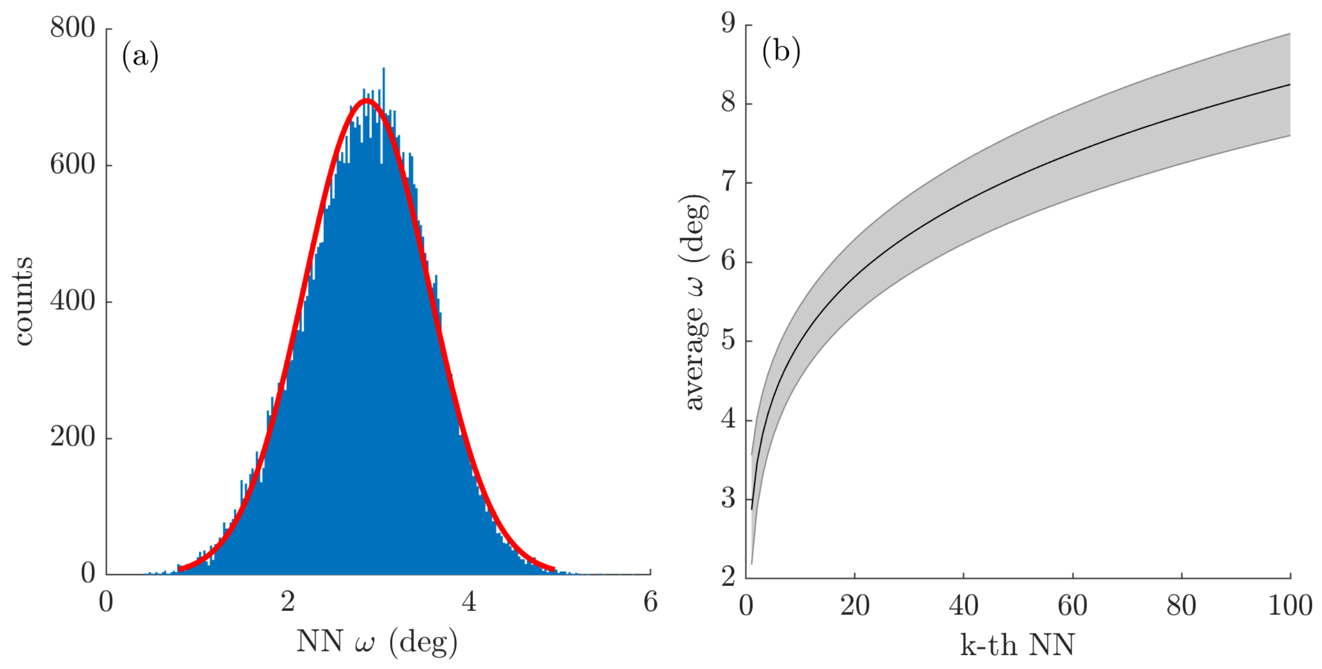

Figure 5: (a) Histogram of NN octonion distances (ω) in a VFZO set of 50 000 points. The average NN distance was $(2.870\,90 \pm 0.691\,12)^\circ$. (b) The average k-th nearest neighbor distances demonstrate that many nearest neighbors fall within a tight tolerance (less then 10°) out of approximately 10 trial runs.

which depends on a modified version of [27]) for each of the four interpolation methods.

All of the methods permit successful interpolation, and the highest density region in all cases falls squarely on the parity line. The GPR and barycentric results show a slight asymmetry such that low energy values are overpredicted more often than they are underpredicted. The width of the point clouds provides a qualitative indication of the dispersion in the prediction errors, and the color indicates the frequency of errors of a given magnitude. As can be seen, the vast majority of errors are very small (the highest density—yellow region—is concentrated on the line of parity). Quantitative measures of the overall accuracy are presented for RMSE (Table 3) and MAE (Table 2) along with results of prior work where available. Error metrics are obtained via VFZO repository function `get_errmetrics.m`.

To aid in objective interpretation of the error metrics, comparison is made to a constant valued control model, whose value is chosen to be the average of `y` (approximately $1.16\,\mathrm{J\,m^{-2}}$ in the limit of `ninputpts` $\to \infty$) resulting in RMSE and MAE values of approximately 0.1283 and $0.0955\,\mathrm{J\,m^{-2}}$. Because the true models differed for other articles, equivalent errors obtained by a constant valued control model (mean of input GBE values) were also included for prior work in Tables 2 and 3. This comparison with the relevant constant-valued function gives a sense of the complexity and variability of the validation function employed in each work and allows for a more objective comparison between differing works. For example, the RMSE for the constant function compared to the validation function employed for the ANN interpolation method in [6] is $0.0854\,\mathrm{J\,m^{-2}}$; in contrast, the RMSE for the constant function compared to the BRK validation function used in this work is $0.1302\,\mathrm{J\,m^{-2}}$ (see Table 3). This suggests that the BRK validation function is more complex and therefore less well approximated by a constant than the validation function used to test the ANN interpolation method in [6]. Consequently, the improved performance of the present methods is even more notable in that the validation function employed here is more difficult to interpolate.

To give further context to the results of this and

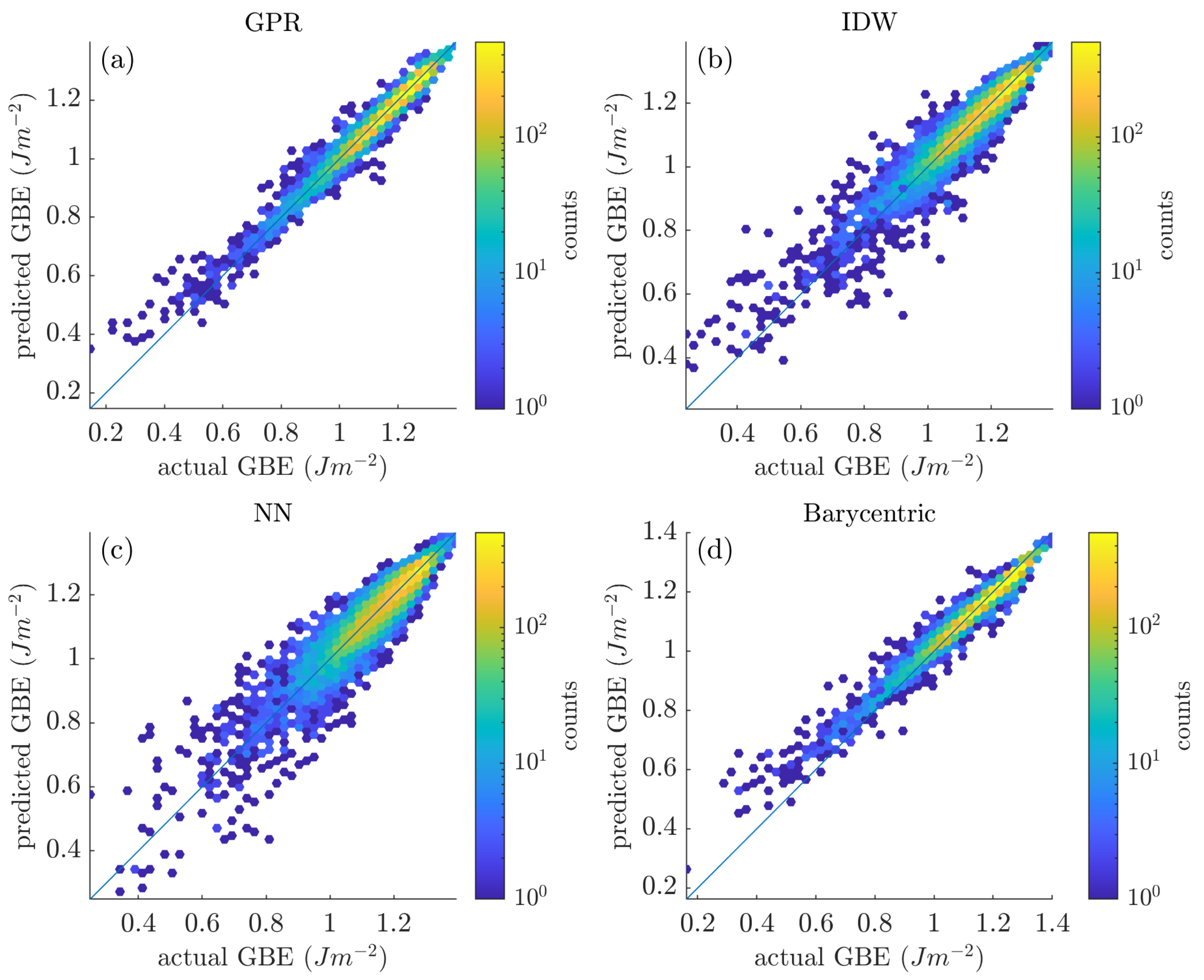

Figure 6: Hexagonally binned parity plots for 50 000 input and 10 000 prediction octonions formed via pairs of a random cubochorically sampled quaternion and a spherically sampled random boundary plane normal. Interpolation via (a) GPR, (b) IDW, (c) NN, and (d) barycentric coordinates. BRK GBE function for FCC Ni [1] was used as the test function.

Table 2: Comparison of average interpolation MAE (approximately 10 trial runs) for each interpolation method in the present work, using 50 000 points in the definition of the VFZ. Comparison to the work of other authors is also included. A constant model (Cst, Avg MAE), whose value was chosen to be the mean of the input GBE was used as a control. The last two columns represent the reduction (↓) in MAE in absolute units of $\mathrm{J\,m^{-2}}$ and % relative to the control model, respectively.

Method	# GBs	MAE $(\mathrm{J\,m^{-2}})$	Cst, Avg MAE $(\mathrm{J\,m^{-2}})$	MAE ↓ $(\mathrm{J\,m^{-2}})$	MAE ↓ (%)
GPR	50 000	0.0145	0.0955	0.081	84.8
Barycentric	50 000	0.0145	0.0955	0.081	84.8
IDW	50 000	0.0225	0.0955	0.073	76.4
NN	50 000	0.0307	0.0955	0.0648	67.9
ANN [6]	17 176	0.0486	0.0617	0.0131	21.2
[illegible]R [9]	388	0.0977	0.1752	0.0775	44.2

Table 3: Comparison of average interpolation RMSE (approximately 10 trial runs) for each interpolation method in the present work, using 50 000 points in the definition of the VFZ. Comparison to the work of other authors is also included. A constant model (Cst, Avg RMSE), whose value was chosen to be the mean of the input GBE was used as a control. The last two columns, RMSE ↓ ($\mathrm{J\,m^{-2}}$ and RMSE ↓ (%)), represent the reduction in RMSE in units of $\mathrm{J\,m^{-2}}$ and % relative to the control model, respectively.

Method	# GBs	RMSE ($\mathrm{J\,m^{-2}}$)	Cst, Avg RMSE ($\mathrm{J\,m^{-2}}$)	RMSE ↓ ($\mathrm{J\,m^{-2}}$)	RMSE ↓ (%)
GPR	50 000	0.0218	0.1283	0.1065	83
Barycentric	50 000	0.0238	0.1283	0.1045	81.4
IDW	50 000	0.0356	0.1283	0.0927	72.3
NN	50 000	0.0445	0.1283	0.0838	65.3
ANN [6]	17 176	—	0.0854	—	—
LKR [9]	388	0.0977	0.2243	0.1266	56.4

prior works, it is useful to consider what the intrinsic error is for typical GB property data. This provides an idea of the minimum possible interpolation error, since one cannot reliably *detect* lower error in the interpolation than already exists in the prediction data itself. One such estimate is furnished by the work of Shen et al. [28], who introduced a non-discretizing approach to extract relative GB energies from polycrystalline samples using the locally optimal block preconditioned conjugate gradient (LOBPCG) method. Their approach utilizes regularization imposed on triple junction (TJ) equilibrium equations and k-nearest neighbor (kNN) distances. Using 60 000 TJs (180 000 GBs) and a custom, non-smooth validation function they obtained GBE RMSE values of $0.0076\,\mathrm{J\,m^{-2}}$ and $0.0277\,\mathrm{J\,m^{-2}}$ for GBE values greater than $0.9\,\mathrm{J\,m^{-2}}$ and less than $0.9\,\mathrm{J\,m^{-2}}$, respectively. This suggests that an optimistic estimate for the error in noise-free *experimental* GBE data obtained using such a method is on the order of $0.0076\,\mathrm{J\,m^{-2}}$ to $0.0277\,\mathrm{J\,m^{-2}}$, which also serves as an estimate of the minimum achievable noise-free experimental interpolation error for any of the interpolation methods described here. Similar analysis for *simulation* data is provided in Section 3.3 and Section S4.2.

By comparing a constant-valued control model[8] to the validation function [28], we calculate a RMSE and MAE of $0.0976\,\mathrm{J\,m^{-2}}$ and $0.0466\,\mathrm{J\,m^{-2}}$, respectively. This implies that the validation model used by Shen et al. [28] is simpler[9] than the BRK validation model employed in the present work.

As shown in Tables 2 and 3, of the four interpolation methods from this work, GPR has the lowest error, both in terms of RMSE and MAE, while NN has the highest error. Compared to a constant valued control model, GPR interpolation reduced the prediction RMSE by 83 %, which outperforms all of the interpolation methods in this work with respect to accuracy, as well as those considered from the literature. After GPR the next most accurate methods are barycentric and LKR interpolation (with the order depending on whether RMSE or MAE are used as the error measure), followed by IDW, NN and then ANN. It is worth noting that the RMSE interpolation error for the GPR and barycentric methods is comparable to the minimum achievable interpolation error (the estimated error in experimental data obtained via the LOBPCG analysis mentioned previously). This is even more signifi-

[8]We use the mean of the true GBEs from their validation function to define the constant-valued control model instead of the mean of the input GBEs because the latter does not exist for polycrystalline data.

[9]Shen et al. [28] used 8 cusps of varying depths and widths based on the Read-Shockley model and unity GBE everywhere else.

cant because the BRK validation function used in the present work is more complex and difficult to interpolate than that used in [28].

The accuracy of the predictions made using the VFZ methods depends on the VFZO set size and distribution. Figure 7 compares the prediction accuracy for each of the 4 methods to the constant valued control model, as a function of the number of input VFZOs (`ninputpts`). As expected, higher density VFZO sets result in lower error, but eventually gives diminishing returns. Moreover, the standard deviations produced via multiple runs are tightly constrained and generally shrink as the VFZO set size increases.

GPR consistently gives lower error than the other three interpolation methods for all VFZO set sizes. NN interpolation produces the worst error of the four methods, but is better than a constant valued control model (i.e. average of the input GBEs) so long as `ninputpts` exceeds a few hundred input points.

It is worthwhile to note that both GPR and IDW are kernel-based in that a model parameter controls the size of the region that can influence the interpolation results. In the GPR case, this is automatically calculated via an internal fitting routine of `fitrgp()`. NN distance distributions (Figure 5) can lead to insight about correlation lengths in a given VFZO set and is used in the IDW implementation. For IDW, the radius of influence is set to $r = \sqrt{2}\mu$, where μ is the mean NN distance. It is likely that better tuning of the kernel parameters in these two methods (such as use of built-in hyperparameter optimization in the case of `fitrgp()`) could further decrease their interpolation errors.

By contrast, barycentric interpolation automatically adjusts its effective region of influence because the size of the simplices in the mesh decreases as the number of vertices increases. More uniformly distributed meshes (such as obtained via constrained optimization [29, 30]) will likely result in lower, more uniform interpolation error, especially for this simplex-based approach which can exhibit high-aspect ratio facets and non-intersections outside the bounds of the mesh (Figure S4). While the barycentric interpolation error is always higher than GPR for the considered set sizes, at 50 000 VFZOs, the errors of GPR and barycentric interpolation are nearly identical.

Predictions for each of the four interpolation methods are plotted as a function of distance along a 1D arc ($\overset{\frown}{AB}$) between two VFZOs, A and B, as shown in Figure 8. Approximate coordinates for A and B are given in Table 4, and each intermediate point between A and B resides on the surface of a hypersphere. Each model used its own set of 50 000 random input VFZOs with GBE sampled via the BRK validation function. The two VFZOs were chosen by taking the furthest apart pair out of 20 000 VFZOs which thus approximates the largest dimension of the VFZ where each endpoint is close to the true VFZ exterior. We believe this is the first[10] plot of a GB property continuously interpolated between two arbitrary GBs (i.e. neither residing entirely in a single misorientation fundamental zone (MFZ) nor a single boundary plane fundamental zone (BPFZ)). Such visualizations can naturally be extended to 2D and 3D by plotting colored points in a triangle or tetrahedron, respectively, all of which (1D, 2D, and 3D) represent small "slices" of the GB character space. Such visualizations naturally suggest the ability to determine and verify numerical derivatives or gradients of GB properties without being restricted to a GB subspace (e.g. MFZ or BPFZ) which can be a useful mathematical construct for the GB community. For example, steepest descent paths and all local GBE minima can be found and used in grain growth simulations. In such a case, use of ensembled VFZO interpolation may be necessary to prevent discontinuity artifacts when crossing the exterior of a VFZ.

3.2. Interpolation Efficiency

GPR is fast and has lower error compared to barycentric interpolation; however, the entire process has to be reevaluated (in current implementation) if the input points (i.e. VFZOs) or input prop-

[10]OSLERP results from [8] plots GB *structure* continuously between two GBs, and [9] performs cross-validation on the simulated Olmsted Ni GBs. The results in both [8] and [9] are distinct from what is presented here: a plot of continuously interpolated GBEs between two arbitrary GBs.

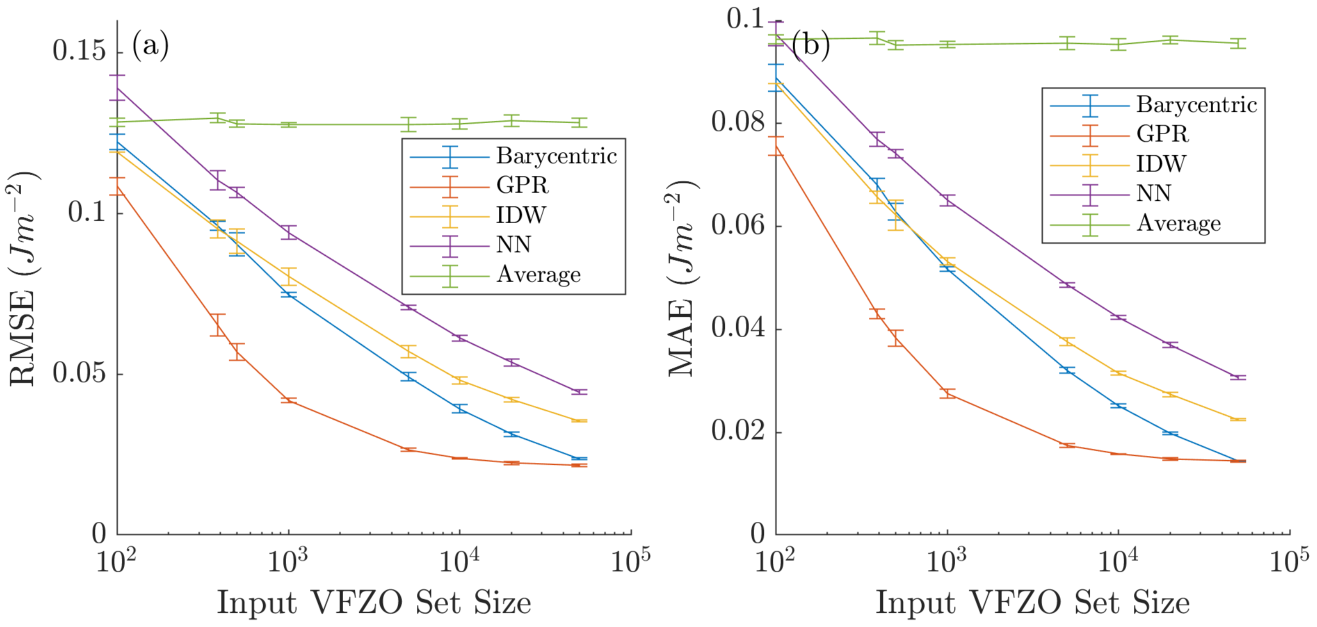

Figure 7: (a) Average RMSE and (b) average MAE vs. number of input points for (planar) barycentric (blue), GPR (orange), IDW (yellow), and NN (purple) interpolation for approximately 10 random runs with different input and prediction points. Standard deviations of approximately 10 runs are also included. Compare with approximately 0.1283 J m^{-2} and 0.0955 J m^{-2} RMSE and MAE, respectively, for a constant, average model (green) using the average of the input properties (approximately 1.16 J m^{-2}).

Table 4: Approximate coordinates of VFZOs A and B used for the interpolation in Figure 8. Individual quaternions of each octonion are given in the active sense and in the laboratory reference frame with an assumed GB normal pointing in the +z direction, also in the laboratory reference frame.

Octonion	o(1)	o(2)	o(3)	o(4)	o(5)	o(6)	o(7)	o(8)
A	0.8658	-0.4269	-0.1270	0.2280	0.2810	0.8390	-0.3852	0.2622
B	0.4684	-0.7657	-0.4100	-0.1617	-0.1483	0.8204	-0.3588	0.4198

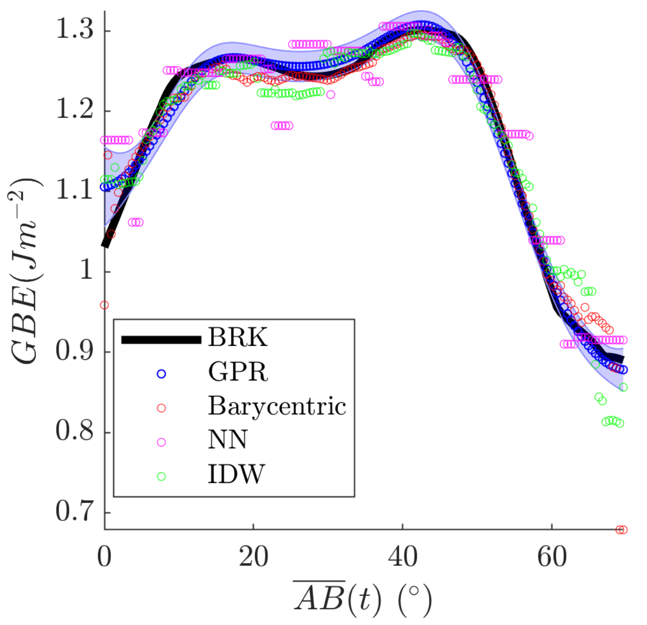

Figure 8: Predictions of GPR (blue circles), barycentric (red circles), NN (magenta circles), and IDW (green circles) as a function of distance along a 1D arc ($\overline{AB}$) between two VFZOs (A and B). The true, underlying BRK function is also shown (black line). 50 000 random input VFZOs were generated and used for each of the models. 150 equally spaced points between A and B were used as prediction points. GPR uncertainty standard deviation is plotted as shaded error band.

erty values (i.e. GBEs) change (typically referred to as predictors and responses, respectively, in the machine learning community). On the other hand, barycentric interpolation is fast if the triangulation and intersections are pre-computed and only input property values change (see `interp_bary_fast.m`), but slow if the input or prediction points change, which requires recomputing the triangulation and intersections.

Computational runtimes of the various interpolation methods are also shown (Table 5). Barycentric interpolation takes the longest, in spite of the fact that it is the only parallelized method by default (not accounted for in Table 5). In other words, since 12 cores were used to obtain these runtime results, the total runtime across all cores is much higher compared with the other methods; however, it is possible that other methods used multi-threading via built-in vectorized functions. The long computation times of barycentric interpolation result primarily from the large number of facets present in a high-dimensional mesh triangulation and the interconnectedness of facets with respect to each other.

GPR is the second-longest in terms of of runtime, but is more accurate than any of the other three methods. NN and IDW interpolation have vectorized implementations and are much simpler than the barycentric and GPR methods. Consistent with expectations, NN and IDW exhibit almost negligible runtimes; however, this is at the expense of increased error, as discussed earlier (Section 3.1). It should also be noted that barycentric interpolation has much higher memory requirements than GPR, NN, and IDW due to the need to store large matrices. If `fitrgp PredictMethod = 'exact'`, then GPR also has high memory requirements for large VFZO sets. For 50 000 input points with sufficient RAM (e.g. 128 GB) and 12 cores available, the `'exact'` method runtime is (535.1 ± 392.6) seconds.

Because the default implementation of IDW uses a radius cut-off, the distance and weight matrices can be stored as sparse objects, dramatically reducing both the storage requirements and computational complexity of this method. We expect that a kNN approach would produce similar results both in terms of runtime and error when a relatively uniform sampling of grain boundary character (GBC) is obtained.

3.3. Simulation Dataset

Interpolation results for a large database of Fe GBE simulations [2] are presented in Figure 9, where approximate coordinates for the octonions A and B in Figure 9b are given in Table 6). A GPR mixture model (Figure S5) based on a sigmoid mixing function (Figure S6) that builds on the GPR interpolation scheme discussed in this work is used to better predict low GBE values of the noisy dataset. If multiple metastable GBEs (e.g. 3-10 metastable states per distinct GB rather than a single metastable GBE) were available and/or the GBs were sampled more uniformly in 5DOF space, the GPR mixing scheme may not have been needed to improve predictive performance of low GBEs. For example, GPR in Figure 6a shows reasonable predictive performance of low GBEs, but the GBs are more uniformly sampled and the input and prediction data are noise-free (compare with the poor predictions for low GBE for this data set shown in Figure S5d).

First, the Fe simulation data is obtained from [2] rather than [7] due to a mistake in the earlier dataset file[11]. First, GBs with a GBE less than $0.01\,\mathrm{J\,m^{-2}}$ are removed to get rid of "no-boundary" GBs. Repeated GBs are then identified and removed by converting all GBs into a VFZO set (see `Kim2oct.m`) and applying `avgrepeats.m` with `avgfn='min'` to sort the repeated GBs into "degenerate sets"[12], and only the average GBE (and a single GB) within each degenerate set was retained. We estimate the intrinsic RMSE and MAE of the Fe simulation dataset to be $0.065\,29\,\mathrm{J\,m^{-2}}$ and $0.061\,90\,\mathrm{J\,m^{-2}}$, respectively. Minimum and maximum error was $-0.2625\,\mathrm{J\,m^{-2}}$ and $0.2625\,\mathrm{J\,m^{-2}}$, respectively. See Section S4.2 for further details on methods used to estimate intrinsic error of the Fe simulation dataset.

An exponential rather than a squared exponential kernel was used for the subset GPR model (Section S4) to accommodate sharper transitions to better approximate low GBEs. Further details of the GPR mixture model are given in Supplementary Information (Section S4). Because only a single metastable state was used for each GBE simulation, both the training and validation data are subject to noise, consistent with a wide lateral spread of predictions in both Figure 9 and the intrinsic error estimation (Figure S9). The Fe simulation dataset GPR mixture model gives lower RMSE ($0.055\,035\,\mathrm{J\,m^{-2}}$) and MAE ($0.039\,185\,\mathrm{J\,m^{-2}}$) than the intrinsic error estimates. This indicates that the intrinsic error itself is somewhat overestimated[13]. The fact that both model and intrinsic error metrics are relatively close and that the parity plots (Figure 9 and Figure S9b) also suggests that the model is performing well and that further improvements in the model relative to the "true" values will be "hidden", i.e. they will probably not manifest as lower RMSE or MAE nor as more tightly distributed parity plots, etc.

Given the theoretical existence of a true minimum GBE, the predictions can be assumed to have an overprediction bias relative to the true minimum. On average, we expect this overprediction bias relative to the true minimum GBE (rather than the most likely metastable state) may be on the order of a few hundred $\mathrm{mJ\,m^{-2}}$ and may vary as a function of true minimum GBE. In other words, the model provided is probably an estimate of the most likely metastable GBE rather than the true minimum GBE. This is akin to saying we're providing a model that approximates the non-equilibrium, Stillinger quenched red curve of Figure 4(c1) in [31], not the minimum GBE blue curve of the same chart. See [31] for an in-depth treatment of equilibrium and metastable GBE.

[11] We were informed of the error during an email discussion with the corresponding author of [2].

[12] A degenerate "set" is distinct from a VFZOs "set", the former of which is discussed in greater detail in Supplementary Information Section S4.2.

[13] The prediction error of a model typically cannot be less than the noise of the prediction data of a model even if the model is estimating the true prediction values with better accuracy than the noise (which is very possible and even expected with GPR models and Guassian noise in the input data).

Table 5: Comparison of average runtime (s) for 10 trials for barycentric, GPR, IDW, and NN interpolation methods for various VFZO set sizes using 12 cores. Because GPR, IDW, and NN method defaults do not use `parfor` loops but may have internal multi-core vectorization, it is unclear to what extent the number of cores affects the runtime of methods other than barycentric interpolation. VFZO symmetrization runtime was not included; however, symmetrization of 50 000 GBOs takes approximately 76 seconds on 6 cores (Intel i7-10750H, 2.6 GHz) and is a common step in every interpolation method (i.e. it is fundamental to the VFZO framework).

	Runtime (s)			
VFZO Set Size	Barycentric	GPR	IDW	NN
100	191.8 ± 19.57	0.4187 ± 0.4342	0.034 ± 0	0.0367 ± 0.0041
388	388.4 ± 18.84	0.943 ± 0.3481	0.0904 ± 0.0224	0.0705 ± 0.0129
500	455.7 ± 55.28	0.6104 ± 0.3138	0.1352 ± 0.0364	0.0724 ± 0.0051
1000	536.5 ± 35.26	1.743 ± 0.9464	0.1948 ± 0.0395	0.1203 ± 0.0184
5000	998.9 ± 54.48	5.216 ± 0.4816	0.8726 ± 0.1529	0.9277 ± 0.2418
10 000	1516 ± 56.59	5.609 ± 0.8756	1.631 ± 0.3915	0.8938 ± 0.1717
20 000	2526 ± 119.5	11.45 ± 3.29	3.191 ± 0.4752	1.275 ± 0.3423
50 000	5743 ± 361.3	13.69 ± 4.05	7.635 ± 1.872	3.817 ± 0.5884

Again, data for which multiple metastable GBEs (e.g. 3-10 repeats) are provided for each GB will likely greatly improve the performance of the GPR model in predicting either most likely metastable GBE (when all GBEs are considered) or true minimum GBE (when only the minimum GBE is considered for each GB) and may even negate the need for a GPR mixture approach. Thus, where feasible, we hope that future large-scale GB bicrystal simulation studies will report all property data for repeated trial runs rather than a single trial run or a single value from a set of trial runs. Ideally, data for the three additional microscopic degree of freedoms (DOFs) for GBs (which falls into the category of epistemic uncertainty in this work) would also be included. We believe it is likely that minimum energy paths (i.e. paths of steepest descent) in the GBE landscape depend on both macroscopic and microscopic DOFs (in total, 8DOF) and could offer a more holistic view of GB behavior that better mimics and explains experimental grain growth observations.

The model can be probed at new GBs via `gprmix.m`, which provides both predicted GBE and uncertainty standard deviation. A subject of future study in our group is to use this model for grain growth simulations.

4. Conclusion

In this work, we presented the VFZO framework for computing distances between GBs and predicting the properties of GBs from existing measurements. We found that distance calculations in the VFZO framework (via VFZO repository function `GBdist4.m`) are dramatically more computationally efficient ($O(N_p^2 L)$) than traditional methods ($O(N_p^4 L^2)$) at the expense of infrequent, large distance overestimation which can be addressed through ensemble techniques at a small computational cost (relative to traditional methods) as discussed in Section 2.1.3.

We also developed and tested a barycentric interpolation method, and adapted three other interpolation methods for use in the VFZO framework. We provide an easy-to-use, versatile implementation of our methods through an interpolation function `interp5DOF.m` written in MATLAB (`github.com/sgbaird-5dof/interp`, [13]) and many companion functions in the VFZO repository. This approach is general and applies to any crystal system (any of the 32 crystallographic point groups can be selected by the parameter `pgnum`).

Of the interpolation methods that we present in this work, GPR provided the highest accuracy predictions. It also provided higher accuracy predictions than any of the methods in the literature. The GPR interpolation errors (50 000 VFZOs) for the

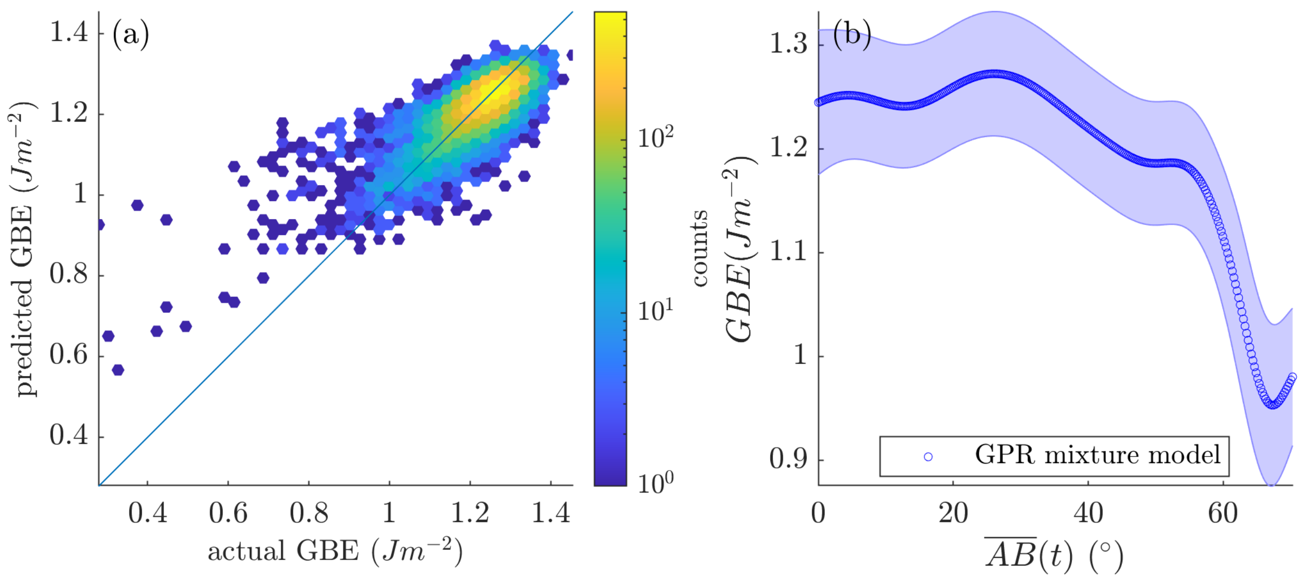

Figure 9: Interpolation results for a large Fe simulation database [2] using 46 883 input GBs and 11 721 prediction GBs in an 80%/20% split and a GPR mixture model to better approximate low GBEs. Use of a GPR mixture model predicts low GBE better than the standard GPR model (compare with Figure S5d). (a) Hexagonally binned parity plot of the GPR mixing model with RMSE and MAE of 0.055 035 $\mathrm{J\,m^{-2}}$ and 0.039 185 $\mathrm{J\,m^{-2}}$, respectively, relative to typical, constant average models of 0.0854 $\mathrm{J\,m^{-2}}$ and 0.0617 $\mathrm{J\,m^{-2}}$, respectively. (b) Predictions of GPR mixture model (blue circles) as a function of distance along a 1D arc ($\overline{AB}$) between two VFZOs (A and B).

Table 6: Approximate coordinates of VFZOs A and B used for the Fe simulation interpolation in Figure 9. Individual quaternions of each octonion are given in the laboratory reference frame with an assumed GB normal pointing in the +z direction, also in the laboratory reference frame.

Octonion	o(1)	o(2)	o(3)	o(4)	o(5)	o(6)	o(7)	o(8)
A	0.8716	-0.4124	-0.1857	0.1893	0.3146	0.8359	-0.3815	0.2382
B	0.4391	-0.7856	-0.4142	-0.1360	-0.1376	0.8082	-0.3705	0.4366

BRK validation model are about 2.4 times the intrinsic error that would be expected from noise-free reconstruction of polycrystalline data via LOBPCG [28] (180 000 GBs) with their (simpler) validation model. Moreover, the interpolation errors for a Fe simulation dataset are on par with the intrinsic errors of the dataset itself (Section S4.2). While IDW, and NN interpolation, have the fastest computation times, they also have higher interpolation error. Consequently, we recommend the GPR interpolation method for the VFZO framework for most applications because it provides the best combination of accuracy and speed and handles input noise; however, the other methods can meet niche needs. For example, barycentric interpolation enables rapid and accurate predictions when the function to be evaluated changes, but the input and prediction GBs remain fixed.

We anticipate that the VFZO framework and corresponding implementation will benefit numerous applications related to GB structure and properties, including facilitating GB structure-property model development, enabling efficient surrogate modeling of GB properties, and larger scale iterative simulations that require repetitive evaluation of computationally expensive structure-property models.

Acknowledgement

The authors thank Ian Chesser, Toby Francis, Victoria Baird, and Brandon Snow for useful discussions. The material presented here is based upon work supported by the National Science Foundation under Grant No. 1610077.

Appendix A Detailed Barycentric Interpolation Method

We describe barycentric interpolation applied in the VFZO framework in more detail. This includes:

1 triangulation of a VFZ mesh (Appendix A.1)

2 finding intersections between arbitrary VFZOs and the VFZ mesh (i.e. finding intersecting facets) (Appendix A.2)

3 calculating interpolated values of an arbitrary VFZO property using the intersecting facet (Appendix A.3)

A.1 Triangulating a Voronoi Fundamental Zone Mesh

Creation of a simplicial mesh is necessary to perform barycentric interpolation. Due to the difficulty of visualizing a 7-sphere, we provide visual illustrations of the process as applied to lower-dimensional analogues. After GBOs have been symmetrized into a VFZ (Section 2.1.1), the triangulation process is as follows:

1.1 Apply a SVD transformation to remove the U(1)-symmetry degeneracy inherent in the VFZO coordinates (Appendix A.1.1)

1.2 To reduce computational burden of the triangulation, linearly project VFZOs onto a hyperplane that is tangent to the vector between the origin and the mean of the input VFZOs

1.3 Perform a second SVD transformation (Appendix A.1.3)

1.4 Compute the triangulation according to the quickhull algorithm [11] using built-in methods

In the explanation of each of these steps that follows, e make reference to lower-dimensional visual analogues of the VFZO triangulation procedure, which are given in Figure A.1, Figure A.2, and Figure A.3. We note that 3D Cartesian coordinates in Figure A.1 correspond to 8D Cartesian coordinates, whereas 3D Cartesian coordinates in Figure A.2 and Figure A.3 correspond to 7D Cartesian coordinates. This is intentional for two reasons:

- Figure A.1 illustrates that unsymmetrized 8D Cartesian GBOs are analogous to a point cloud on the 2-sphere (Figure A.1a) and that a 8D Cartesian VFZO set, which has already been symmetrized, is analogous to a geodesic arc on the 2-sphere (Figure A.1b). A VFZO set has a degenerate dimension that can then be removed by a rigid SVD transformation to 7D

Cartesian coordinates (analogous to 2D Cartesian coordinates in Figure A.1c). This sequence would be more difficult to visualize if Figure A.1a was meant to represent a point cloud on the 3-sphere (4D Cartesian coordinates), etc.

- Figure A.2 illustrates a second SVD transformation from normalized 7D Cartesian coordinates (Figure A.2a) to 6D Cartesian coordinates (Figure A.2b). Key issues are retained that would otherwise be lost (Section S3.1) if an arc on a circle (1-sphere) to 1D Cartesian coordinates were used instead[14]. Additionally, the use of actual triangles is a more familiar and compelling illustration of *triangulation*.

While lower dimensional analogues are useful for visualizing and understanding the process of triangulation, a written description is also given in the following sections. As appropriate, we refer back to the teaching figures described in this section.

A.1.1 Singular Value Decomposition Transformation from 8D Cartesian to 7D Cartesian

To reduce the computational complexity of triangulating a high-dimensional mesh [11], some simplifications are made. First, the degenerate octonion dimension obtained from analytically minimizing $U(1)$ symmetry [8] is removed via a rigid (i.e. distance- and angle-preserving) SVD transformation, analogous to a Cartesian rotation and translation (see 3D to 2D SVD transformation from Figure A.1b to Figure A.1c).

A.1.2 Linearly Project onto Hyperplane

Next, the resulting 7D Cartesian representation of each VFZO is projected onto a hyperplane that is tangent to the centroid (i.e. mean) of the VFZO set[15] (Figure A.2a). By performing this linear projection, one of the dimensions becomes degenerate.

[14]Non-intersection issues due to high-aspect ratios and consideration of facets connected up to `nnMax` NNs do not manifest in triangulations on the surface of a 1-sphere because one of the two facets (i.e. line segments) connected to the first NN mesh vertex relative to the prediction point is guaranteed to have an intersection.

A.1.3 Singular Value Decomposition Transformation from 7D Cartesian to 6D Cartesian

This additional degeneracy is removed via a second SVD transformation, this time to 6D Cartesian coordinates (see 3D to 2D projection in Figure A.2a-b). Finally, the resulting points can be triangulated via the quickhull algorithm [11] (see VFZO repository function `sphconvhulln.m` and built-in MATLAB function `delaunayn()`), which relies on Euclidean distances[16]. Because the simplicial mesh is defined by a list of edges between vertices for each simplicial facet, this list applies immediately to the VFZO set in its 7D Cartesian coordinates (i.e. no reverse transformation is necessary to use the mesh on the 6-sphere in 7D).

A.2 Intersections in a Voronoi Fundamental Zone Mesh

Once the triangulation has been determined, we need to find which facet each prediction point intersects (i.e. find the intersecting facet). There are two sub-steps:

2.1 The same rigid transformation needs to be applied to the prediction points as was applied to the input points, otherwise the prediction points won't line up properly with the mesh (Appendix A.2.1)

2.2 Facets nearby a prediction point are then identified and tested for intersection (Appendix A.2.2).

[15]This is *not* a rigid transformation; however, it approximates one with sufficient accuracy to produce a high-quality triangulation in a VFZ.

[16]While the triangulation algorithm used in this work relies on Euclidean distances (the use of which is possible via the VFZO framework), other distance metrics that are non-Euclidean [32] could potentially be incorporated into the barycentric approach such as by doing an edge-length based simplex reconstruction [33, 34] using the VFZ triangulation edge lengths.

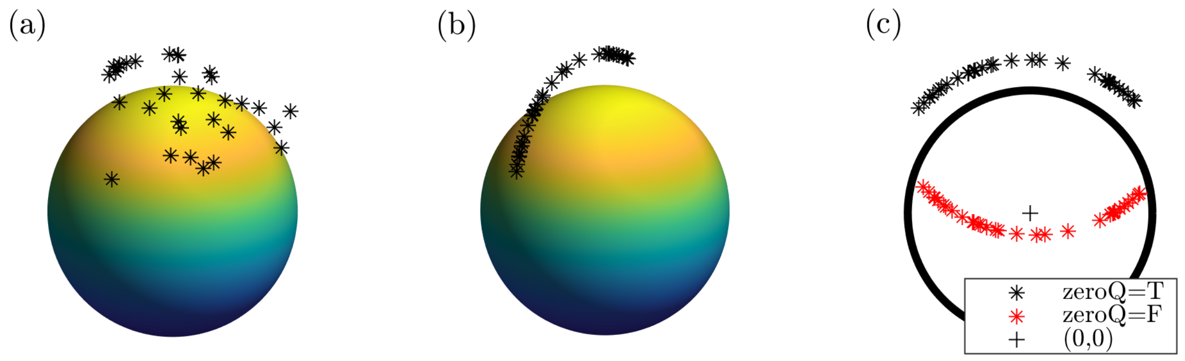

Figure A.1: 3D Cartesian to 2D Cartesian analogue of 8D Cartesian to 7D Cartesian degeneracy removal via rigid SVD transformation as used in barycentric interpolation approach. (a) Starting spherical arc points on surface of 2-sphere, (b) rotational symmetrization applied w.r.t. z-axis (analogous to U(1) symmetrization), and (c) degenerate dimension removed via singular value decomposition transformation to 2D Cartesian with either the origin (black plus) preserved (black asterisks, `zeroQ=T`) for triangulation or ignored (red asterisks, `zeroQ=F`) for mesh intersection. The spheres (a,b) and circle (c) each have a radius of 0.8 and are used as a visualization aid only.

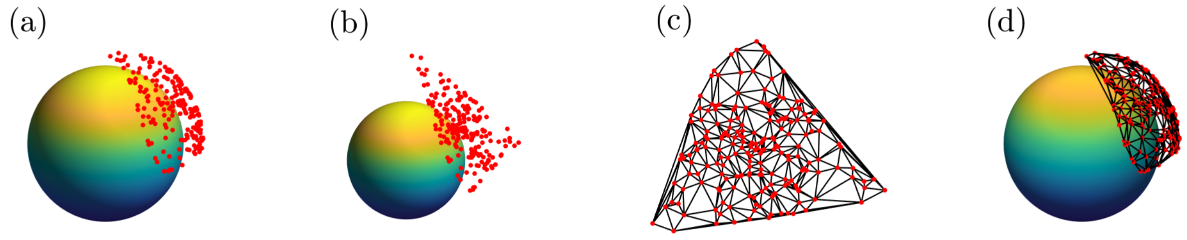

Figure A.2: 3D Cartesian to 2D Cartesian analogue of 7D Cartesian to 6D Cartesian mesh triangulation used in barycentric interpolation approach. (a) 3D Cartesian input points are (b) linearly projected onto hyperplane that is tangent to mean of starting points. (c) The degenerate dimension is removed via a rigid SVD transformation to 2D Cartesian and the Delaunay triangulation (black lines) is calculated, with input vertices (red). Delaunay triangulation superimposed onto normalized input points (d). The spheres in (a), (b), and (d) have a radius of 0.8 and are used for visualization only.

A.2.1 Apply Same Singular Value Decomposition to prediction Points

The positions of the prediction points relative to the mesh need to be constant even after the rigid SVD transformation. This is easily accomplished by:

2.1a concatenating both input and prediction points

2.1b using the `interp5DOF` sub-routine `proj_down` (which depends on MATLAB's built-in SVD implementation `svd()`) to perform the transformation

2.1c subsequently separating the transformed input and prediction points (reverse of concatenation step)

To map new points onto the mesh, the `USV` structure output by `proj_down.m` needs to be stored and supplied in future calls to `proj_down.m`.

A.2.2 Testing Nearby Facets for Intersections

Once the prediction points are lined up properly with the mesh, the intersecting facet needs to be found. The facet containing the prediction point (i.e. intersecting facet) is the one for which the point's barycentric coordinates are positive. Consequently, we determine facet affiliation by:

2.2a linearly projecting the prediction point onto the hyperplane defined by a mesh facet's vertices (Figure A.3)

2.2b computing the point's barycentric coordinates within the facet

2.2c testing that all coordinates are positive [22]

2.2d repeating steps 2.2a-2.2c until an intersection is found or a stop condition is reached (see `nnMax` below).

Due to the large number of facets per point of a high-dimensional triangulation (approximately 2000 facets per vertex for a 50 000 point VFZ triangulation, or 1×10^8 total facets), some simplifications are made in order to determine intersections of prediction points with the mesh. If every edge length of every facet were equal, only facets connected to the first NN would need to be considered to find a proper intersection. However, since the VFZOs are randomly sampled, edge lengths of facets are non-uniform, and non-unity aspect ratio facets exist (Figure A.2, Figure S4). If the facets have high-aspect ratios, the intersecting facets of prediction points can be far from the NNs mesh points relative to the prediction points (see Figure S4 inset), especially near the perimeter of a hyperspherical surface mesh. Rather than loop through every facet to find an intersection (1×10^8 facets in 50 000 VFZO mesh), the prediction point intersections are calculated by considering facets connected to up to some number of NN mesh vertices (`nnMax`) relative to each prediction point (in this work, `nnMax=10`). The NN mesh vertices relative to a prediction point are computed via the MATLAB built-in function `dsearchn` as in the NN approach (Section 2.3.4). The facet IDs of facets connected to these NNs are computed by calling built-in MATLAB function `find()` as in `find(K==nn)`, where `K` is the triangulation from VFZO repository function `sphconvhulln.m` and `nn` is the ID of one of the NN mesh vertices.

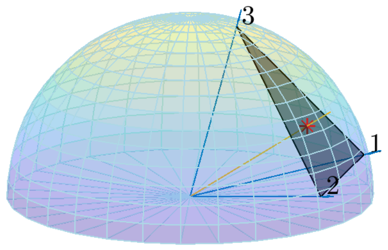

Figure A.3: A ray (red line) is linearly projected from the 2-sphere onto the hyperplane of a mesh facet (transparent black), shown as a red asterisk. The barycentric coordinates are computed as $\lambda_{i\in[1,3]} = \frac{1}{3}$. Because all barycentric coordinates are positive, it is determined that the projected point is an intersection with the mesh. Given vertex values of 8.183, 3.446, and 3.188 for vertices 1, 2, and 3, respectively, the interpolated value is calculated as 4.94 via Eq. (5).

Some prediction points will have no intersecting facet found. From our numerical testing, we determined that this non-intersection phenomenon occurs in two situations:

- high-aspect ratio facets (described above)
- prediction points that are positioned just out-

side the bounds of the mesh but within the bounds of the VFZ, due to the fact that the mesh is a piecewise linear approximation of a surface with a curved perimeter and that randomly sampled points typically do not fall on the true perimeter

In the first case, barycentric interpolation within high-aspect ratio facets may actually lead to worse interpolation error than a NN interpolation strategy due to influence by GBs far from the prediction point. In the second case, there is no true intersection between the prediction point and the mesh. Both issues can be addressed with the same strategy: we apply a NN approach (Section 2.3.4) when an intersecting facet is not found within `nnMax` NNs. In numerical tests, VFZ meshes composed of 388 and 50 000 vertices produced non-intersection rates of $(12.07 \pm 1.02)\,\%$ and $(0.68 \pm 0.11)\,\%$, respectively, over approximately 10 trials and using 10 000 prediction points for each trial.

Testing intersections for nearby facets is handled in the the VFZO repository function `intersect_facet.m` and depends barycentric coordinate computations in `projray2hypersphere.m`.

A.3 Interpolation via Barycentric Coordinates

Once a mesh triangulation has been determined (Appendix A.1) and barycentric coordinates are computed for a prediction point within the input mesh (Appendix A.2), the interpolated value is found by taking the dot product of the prediction point's barycentric coordinates and the properties of the corresponding vertices of the intersecting facet via

$$v_{m,q} = \sum_{i=1}^{N} \lambda_{m,i} v_{m,i} \tag{5}$$

where $\lambda_{m,i}$, $v_{m,q}$, $v_{m,i}$ and N, are the barycentric coordinates of the m-th prediction point, interpolated property at the m-th prediction point, property of the i-th vertex of the intersecting facet for the m-th prediction point, and number of vertices in a given facet ($N = 7$ for facets of the simplicial mesh on the degeneracy-free 6-sphere), respectively. Interpolation of many prediction points simultaneously can be accomplished by a simple, vectorized approach via MATLAB built-in function `dot()` as used in VFZO repository function `interp_bary_fast.m`. This function assumes triangulation and weights have been precomputed. In other words, both input and prediction coordinates remain fixed, and only input property values change. If this is the case, barycentric interpolation of new points is incredibly fast. By contrast, if input coordinates change, the triangulation must be recomputed, and if prediction coordinates change, the intersecting facets must be recomputed. Both triangulation and finding intersecting facets are computationally demanding with respect to memory and runtime (Section 3.2).

Acronyms

5DOF	five degree-of-freedom 5
BRK	Bulatov Reed Kumar 8
GB	grain boundary 1
GBE	grain boundary energy 2
GPR	Gaussian process regression 1
kNN	k-nearest neighbor 8
MAE	mean absolute error 2
NN	nearest neighbor 5
RMSE	root mean square error 2
VFZ	Voronoi Fundamental Zone 2
VFZO	Voronoi Fundamental Zone octonion 1

References

[1] V. V. Bulatov, B. W. Reed, M. Kumar, Grain boundary energy function for fcc metals, Acta Materialia 65 (2014) 161–175. doi:10.1016/j.actamat.2013.10.057.

[2] H.-K. Kim, S. G. Kim, W. Dong, I. Steinbach, B.-J. Lee, Phase-field modeling for 3D grain growth based on a grain boundary energy database, Modelling and Simulation in Materials Science and Engineering 22 (2014) 034004. doi:10.1088/0965-0393/22/3/034004.

[3] J. Li, S. J. Dillon, G. S. Rohrer, Relative grain boundary area and energy distributions in nickel, Acta Materialia 57 (2009) 4304–4311. doi:10.1016/j.actamat.2009.06.004.

[4] S. J. Dillon, G. S. Rohrer, Characterization of the grain-boundary character and energy distributions of yttria using automated serial sectioning and ebsd in the FIB, Journal of the American Ceramic Society 92 (2009) 1580–1585. doi:10.1111/j.1551-2916.2009.03064.x.

[5] V. Randle, G. S. Rohrer, H. M. Miller, M. Coleman, G. T. Owen, Five-parameter grain boundary distribution of commercially grain boundary engineered nickel and copper, Acta Materialia 56 (2008) 2363–2373. doi:10.1016/j.actamat.2008.01.039.

[6] S. E. Restrepo, S. T. Giraldo, B. J. Thijsse, Using artificial neural networks to predict grain boundary energies, Computational Materials Science 86 (2014) 170–173. doi:10.1016/j.commatsci.2014.01.039.

[7] H. K. Kim, W. S. Ko, H. J. Lee, S. G. Kim, B. J. Lee, An identification scheme of grain boundaries and construction of a grain boundary energy database, Scripta Materialia 64 (2011) 1152–1155. doi:10.1016/j.scriptamat.2011.03.020.

[8] T. Francis, I. Chesser, S. Singh, E. A. Holm, M. De Graef, A geodesic octonion metric for grain boundaries, Acta Materialia 166 (2019) 135–147. doi:10.1016/j.actamat.2018.12.034.

[9] I. Chesser, T. Francis, M. De Graef, E. Holm, Learning the grain boundary manifold: Tools for visualizing and fitting grain boundary properties, Acta Materialia 195 (2020) 209–218. doi:10.1016/j.actamat.2020.05.024.

[10] D. L. Olmsted, S. M. Foiles, E. A. Holm, Survey of computed grain boundary properties in face-centered cubic metals: I. Grain boundary energy, Acta Materialia 57 (2009) 3694–3703. doi:10.1016/j.actamat.2009.04.007.

[11] C. B. Barber, D. P. Dobkin, H. Huhdanpaa, The quickhull algorithm for convex hulls, ACM Transactions on Mathematical Software 22 (1996) 469–483. doi:10.1145/235815.235821.

[12] M. De Graef, EMSoft, 2020. doi:10.5281/zenodo.3489720.

[13] S. Baird, O. Johnson, Five Degree-of-Freedom (5DOF) Interpolation, 2020. URL: github.com/sgbaird-5dof/interp.

[14] D. Rowenhorst, A. D. Rollett, G. S. Rohrer, M. Groeber, M. Jackson, P. J. Konijnenberg, M. De Graef, Consistent representations of and conversions between 3D rotations, Modelling and Simulation in Materials Science and Engineering 23 (2015) 083501. doi:10.1088/0965-0393/23/8/083501.

[15] A. Heinz, P. Neumann, Representation of orientation and disorientation data for cubic, hexagonal, tetragonal and orthorhombic crystals, Acta Crystallographica Section A 47 (1991) 780–789. doi:10.1107/S0108767391006864.

[16] H. Grimmer, A unique description of the relative orientation of neighbouring grains, Acta Crystallographica Section A 36 (1980) 382–389. doi:10.1107/S0567739480000861.

[17] B. Luong, Voronoi Sphere, MATLAB Central File Exchange, 2020. URL: https://www.mathworks.com/matlabcentral/fileexchange/40989-voronoi-sphere.

[18] S. Patala, C. A. Schuh, Symmetries in the representation of grain boundary-plane distributions, Philosophical Magazine 93 (2013) 524–573. doi:10.1080/14786435.2012.722700.

[19] E. R. Homer, S. Patala, J. L. Priedeman, Grain Boundary Plane Orientation Fundamental Zones and Structure-Property Relationships, Scientific Reports 5 (2015) 1–13. doi:10.1038/srep15476.

[20] D. L. Olmsted, E. A. Holm, S. M. Foiles, Survey of computed grain boundary properties in face-centered cubic metals-II: Grain boundary mobility, Acta Materialia 57 (2009) 3704–3713. doi:10.1016/j.actamat.2009.04.015.

[21] S. Singh, M. De Graef, Orientation sampling for dictionary-based diffraction pattern indexing methods, Modelling and Simulation in Materials Science and Engineering 24 (2016). doi:10.1088/0965-0393/24/8/085013.

[22] T. Langer, A. Belyaev, H.-P. Seidel, Spherical barycentric coordinates, Proceedings of the fourth Eurographics symposium on Geometry processing (2006) 81–88. URL: http://portal.acm.org/citation.cfm?id=1281957.1281968.

[23] M. Floater, Generalized barycentric coordinates and applications, Acta Numerica 24 (2015) 161–214. doi:10.1017/S09624929. arXiv:1711.05337v1.

[24] M. Meyer, A. Barr, H. Lee, M. Desbrun, Generalized Barycentric Coordinates on Irregular Polygons, Journal of Graphics Tools 7 (2002) 13–22. doi:10.1080/10867651.2002.10487551.

[25] C. E. Rasmussen, C. K. I. Williams, Gaussian Processes for Machine Learning, Adaptive Computation and Machine Learning, MIT Press, Cambridge, Mass, 2006.

[26] A. Tovar, Inverse distance weight function, MATLAB Central File Exchange, 2020. URL: https://www.mathworks.com/matlabcentral/fileexchange/46350-inverse-distance-weight-function.

[27] G. Bean, Hexscatter, MATLAB Central File Exchange, 2020. URL: https://www.mathworks.com/matlabcentral/fileexchange/45639-hexscatter-m.

[28] Y. F. Shen, X. Zhong, H. Liu, R. M. Suter, A. Morawiec, G. S. Rohrer, Determining grain boundary energies from triple junction geometries without discretizing the five-parameter space, Acta Materialia 166 (2019) 126–134. doi:10.1016/j.actamat.2018.12.022.

[29] E. D. Dolan, J. J. More, T. S. Munson, Benchmarking Optimization Software with COPS 3.0, Technical Report, Argonne National Laboratory (ANL), United States, 2004. doi:10.2172/834714.

[30] MATLAB Optimization Toolbox, Constrained Electrostatic Nonlinear Optimization, Problem-Based, 2020. URL: https://www.mathworks.com/help/optim/ug/constrained-electrostatic-problem-based-optimization.html.

[31] J. Han, V. Vitek, D. J. Srolovitz, Grain-boundary metastability and its statistical properties, Acta Materialia 104 (2016) 259–273. doi:10.1016/j.actamat.2015.11.035.

[32] A. Morawiec, On distances between grain interfaces in macroscopic parameter space, Acta Materialia 181 (2019) 399–407. doi:10.1016/j.actamat.2019.09.032.

[33] R. Connor, L. Vadicamo, F. Rabitti, High-dimensional simplexes for supermetric search, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 10609 LNCS (2017) 96–109. doi:10.1007/978-3-319-68474-1_7. arXiv:1707.08370.

[34] J. D. Boissonnat, R. Dyer, A. Ghosh, S. Y. Oudot, Only distances are required to reconstruct submanifolds, Computational Geometry: Theory and Applications 66 (2017) 32–67. doi:10.1016/j.comgeo.2017.08.001. arXiv:1410.7012.