3D reconstruction of grains in polycrystalline materials using a tessellation model with curved grain boundaries

Ondřej Šedivý[a*], Tim Brereton[b], Daniel Westhoff[b], Leoš Polívka[a], Viktor Beneš[c], Volker Schmidt[b] and Aleš Jäger[a]

[a] *Czech Academy of Sciences, Institute of Physics, Department of Advanced Structural Materials, Na Slovance 1999/2, 18221 Praha 8;*

[b] *Ulm University, Faculty of Mathematics and Economics, Institute of Stochastics, D-89069 Ulm;*

[c] *Charles University in Prague, Faculty of Mathematics and Physics, Department of Probability and Mathematical Statistics, Sokolovska 83, 18675 Praha 8*

A compact and tractable representation of the grain structure of a material is an extremely valuable tool when carrying out an empirical analysis of the material's microstructure. Tessellations have proven to be very good choices for such representations. Most widely used tessellation models have convex cells with planar boundaries. Recently, however, a new tessellation model — called the generalized balanced power diagram (GBPD) — has been developed that is very flexible and can incorporate features such as curved boundaries and non-convexity of cells. In order to use a GBPD to describe the grain structure observed in empirical image data, the parameters of the model must be chosen appropriately. This typically involves solving a difficult optimization problem. In this paper, we describe a method for fitting GBPDs to tomographic image data. This method uses simulated annealing to solve a suitably chosen optimization problem. We then apply this method to both artificial data and experimental 3D electron backscatter diffraction (3D EBSD) data obtained in order to study the properties of fine grained materials with superplastic behavior. The 3D EBSD data required new alignment and segmentation procedures, which we also briefly describe. Our numerical experiments demonstrate the effectiveness of the simulated annealing approach (compared to heuristic fitting methods) and show that GBPDs are able to describe the structures of polycrystalline materials very well.

Keywords: microstructural characterization; grain boundaries; polycrystalline metals; tessellation modeling; generalized balanced power diagram; simulated annealing

1. Introduction

Scientific advances that allow for the production of high quality tomographic images have opened up many avenues of research in materials science. For example, the availability of detailed 3D images has made possible the empirical investigation of grain boundaries in polycrystalline materials. Grain boundaries are of great scientific interest as they play a significant role in determining many material properties; see, e.g., [1, 2].

When working with empirical data on grain boundaries, it is convenient to have

*Corresponding author. Email: sedivy@fzu.cz

a compact and tractable representation of the grains. This representation should provide an idealized description of the observed material, retaining information about key quantities such as the topology of the grain structure and the curvature of the grain boundaries, while removing extraneous information such as noise arising in the imaging process. Ideally, this representation will aid in the description and estimation of quantities such as grain volumes and curvatures. An additional aim of any such representation is to provide a form of data compression, describing large voxel-based data sets using a much smaller number of parameters. Such compression is of increasing importance with the advent of four dimensional data, which records the evolution of grain structures over time. Note that such a representation often has no relation to the physical processes driving grain formation and growth.

Tessellations present a class of mathematical models that divide space into non-overlapping cells. These have proven to be very good representations of grain structures. As a result, they have been used extensively to model the microstructures of polycrystalline materials; see [3] for an early reference and [4] for a general overview. Most classical tessellation models (including the widely used Voronoi and Laguerre tessellations) have cells with planar boundaries and can only model convex grains. This limits their ability to describe many important features of grain structures. Recently, however, a number of models have been developed for tessellations with non-convex grains; see, for example, [5, 6] and [7]. The *generalized balance power diagram* (GBPD), described in [8], is a very general tessellation model that is able to describe typical grain structures very well. In particular, it is able to model grain structures with a wide range of curved boundaries.

In order to use a model such as the GBPD to describe empirical image data, the parameters of the model must be chosen to fit the data. A number of automatic fitting procedures have been successfully developed for Laguerre tessellations and range from heuristic approaches (see [9]) to more sophisticated approaches that use computational tools to solve an optimization problem; see [10]. Stochastic methods have proven to be particularly effective at solving such optimization problems, which tend to be high dimensional and multi-modal; see, for example, [11] and [12]. Fitting GBPDs is a more difficult problem, because the number of parameters to be fitted is considerably larger. Much of current work on GBPDs uses a heuristic method to fit the tessellation to data; see, e.g. [5] and [7]. In [8], linear programming was successfully used to fit a GBPD by solving an optimization problem, obtaining extremely good fits. However, the methods proposed are limited to relatively small image sizes because the memory and time requirements grow very quickly in the number of grains and voxels.

In this paper, we propose a fast stochastic optimization based method for fitting GBPDs to tomographic image data. This method works for large image sizes and results in significantly better fits than the heuristic method. In addition, it is simple and easy to implement, making it more straightforward for practitioners to use GBPDs as modeling tools. We focus on an application of our method to image data obtained using 3D electron backscatter diffraction (3D EBSD), which is an often used imaging technique; see, e.g., [13–15].

The structure of the paper is as follows. We begin by introducing a number of classical tessellation models, explaining how they relate to GBPDs. We then formulate the problem of fitting GBPDs to image data as an optimization problem, describing efficient methods for calculating the cost function, which are key to the effectiveness of our method. Next, we introduce simulated annealing, which is the basis of our fitting methodology, and describe the specifics of our implementation.

The experimental data, which is a series of 3D EBSD images of an aluminum alloy, is then introduced. We describe a number of novel pre-processing steps used to segment and align this data in preparation for model fitting. Finally, we demonstrate the effectiveness of our method by applying it to both artificial image data and the experimental image data, comparing our results to those obtained using heuristic fits.

2. Generalized balanced power diagrams

2.1. *Tessellations*

A tessellation, $\mathcal{T}$, in $\mathbb{R}^3$ is a division of the space into non-overlapping sets, called *cells* or *grains*. More formally, it is a countable collection of closed sets, $\mathcal{T} = \{C_i \subset \mathbb{R}^3\}$, such that

(1) $\mathring{C}_i \cap \mathring{C}_j = \emptyset$ for all $i \neq j$, where $\mathring{C}_i$ is the interior of the set C_i,
(2) $\bigcup_i C_i = \mathbb{R}^3$,
(3) $\mathcal{T}$ is locally finite (i.e., $\#\{C_i \in \mathcal{T} : C_i \cap B \neq \emptyset\} < \infty$ for all bounded $B \subset \mathbb{R}^3$).

For more details, see, e.g., [6], [16] and [17]. Note that some definitions of a tessellation require that the cells should be convex. However, we consider a more general class of tessellations that includes those with non-convex cells.

Many tessellations can be generated by a locally finite (possibly marked) point pattern, $\mathcal{P}$. The points of $\mathcal{P}$ are called the *seeds* or *generators* of the tessellation. The best studied such tessellation is the *Voronoi tessellation*, for which $\mathcal{P}$ is a collection of unmarked points (i.e., $\mathcal{P} = \{\mathbf{x}_i\} \subset \mathbb{R}^3$). In a Voronoi tessellation, each seed in $\mathcal{P}$ generates a non-empty cell. The cell corresponding to a given seed, $\mathbf{x}_i$, is defined to be all points in $\mathbb{R}^3$ that are closer to $\mathbf{x}_i$ than to any other seed in $\mathcal{P}$ with respect to the distance measure given by

$$\mathrm{d}_{\mathsf{V}}(\mathbf{y}, \mathbf{x}) = \|\mathbf{y} - \mathbf{x}\|,$$

where $\|\cdot\|$ is the Euclidean norm in $\mathbb{R}^3$. That is,

$$C_i = \{\mathbf{y} \in \mathbb{R}^3 : \mathrm{d}_{\mathsf{V}}(\mathbf{y}, \mathbf{x}_i) \leq \mathrm{d}_{\mathsf{V}}(\mathbf{y}, \mathbf{x}') \text{ for all } \mathbf{x}' \in \mathcal{P}\}.$$

The cells of a Voronoi tessellation are convex polyhedra; see, e.g., [17].

The Voronoi tessellation can be extended by modifying the distance measure. One of the most widely used modifications is the *Laguerre tessellation* or *power diagram*. In this case, the generating seeds form a marked point pattern, $\mathcal{P} = \{(\mathbf{x}_i, w_i)\} \subset \mathbb{R}^3 \times \mathbb{R}$. The Laguerre distance measure, called the *power distance*, is defined to be

$$\mathrm{d}_{\mathsf{L}}(\mathbf{y}, (\mathbf{x}, w)) = \|\mathbf{x} - \mathbf{y}\|^2 - w.$$

Like the Voronoi tessellation, the Laguerre tessellation has convex cells. However, Laguerre tessellations can produce tessellations with much more variation in terms of both cell size and aspect ratio than Voronoi tessellations; see [6] and [7]. Indeed, any normal tessellation in three dimensions can be described by a Laguerre tessellation; see [18] and [19].

A limitation of both Voronoi and Laguerre tessellations is that they have planar cell boundaries. Curved boundaries have often been regarded as a negative feature of tessellations in the literature, due to the computational burdens they impose. However, from a modeling perspective, the ability to describe curved grain boundaries is of great value. In particular, much of the physics of polycrystalline materials can be described by the geometry of the grain boundaries; see, for example, [1] and [20]. Tessellations with curved boundaries (and, thus, non-convex cells) can be obtained in a number of different ways. Johnson and Mehl proposed a non-convex tessellation that models crystal growth by considering grains that grow isotropically from nuclei; see [3]. A closely related model is the *additively weighted Voronoi tessellation*. In this case, as for the Laguerre tessellation, the generating point pattern is a marked point pattern, $\mathcal{P} = \{(\mathbf{x}_i, w_i)\} \subset \mathbb{R}^3 \times \mathbb{R}$. The additively weighted Voronoi distance measure is given by

$$\mathrm{d}_{\mathsf{W}}(\mathbf{y}, (\mathbf{x}, w)) = \|\mathbf{x} - \mathbf{y}\| - w.$$

Although the resulting cells can have curved boundaries, the range of curvatures that can be obtained is limited. In particular, the boundary between two cells with differing weights is convex for the cell with the smaller weight and concave for the cell with the larger weight; see [5].

The Voronoi, Laguerre and additively weighted Voronoi tessellations are all examples of isotropic tessellations. In [16], a modification of the Voronoi tessellation, called the *Voronoi-G* tessellation, was proposed. This replaces the Euclidean metric, d_{V}, with the anisotropic distance measure

$$\mathrm{d}_{\mathsf{G}}(\mathbf{y}, \mathbf{x}) = \sqrt{(\mathbf{y} - \mathbf{x})^\top G (\mathbf{y} - \mathbf{x})},$$

where G is a positive definite matrix. The resulting tessellations have planar cell boundaries. Note that if G is the identity matrix, this metric reduces to the Voronoi metric. A similar idea is considered within the framework of Riemannian geometry in [21]. In this case, tessellations with curved cell boundaries can be obtained.

The idea of using positive definite matrices to define metrics is extended in [6], where each cell of the so-called *local Voronoi* tessellation is defined using a local metric. This is achieved as follows. Let $\mathbb{M}$ be the space of 3×3 real-valued positive definite matrices. The tessellation is then generated by $\mathcal{P} = \{(\mathbf{x}_i, M_i)\} \subset \mathbb{R}^3 \times \mathbb{M}$. The metric associated with the ith generator, $(\mathbf{x}_i, M_i)$, is given by

$$\mathrm{d}_{\mathsf{M}}(\mathbf{y}, (\mathbf{x}_i, M_i)) = \sqrt{(\mathbf{y} - \mathbf{x}_i)^\top M_i (\mathbf{y} - \mathbf{x}_i)},$$

with the cell corresponding to $(\mathbf{x}_i, M_i)$ given by

$$C_i = \{\mathbf{y} \in \mathbb{R}^3 : \mathrm{d}_{\mathsf{M}}(\mathbf{y}, (\mathbf{x}_i, M_i)) \leq \mathrm{d}_{\mathsf{M}}(\mathbf{y}, (\mathbf{x}', M')) \text{ for all } (\mathbf{x}', M') \in \mathcal{P}\}.$$

In [5], different choices of local metrics are considered. In particular, a squared local distance is considered. This distance measure takes the form

$$\mathrm{d}_{\mathsf{P}}(\mathbf{y}, (\mathbf{x}_i, M_i)) = (\mathbf{y} - \mathbf{x}_i)^\top M_i (\mathbf{y} - \mathbf{x}_i),$$

where the $\{M_i\}$ are again real-valued positive definite matrices. The local metric model is further extended in [8] to obtain the *generalized balanced power diagram* (GBPD), so called because it generalizes the Laguerre (power) diagram. This is a tessellation generated by a set $\mathcal{P} = \{(\mathbf{x}_i, M_i, w_i)\} \subset \mathbb{R}^3 \times \mathbb{M} \times \mathbb{R}$. The local distance measure is of the form

$$\mathrm{d}_{\mathsf{GB}}(\mathbf{y}, (\mathbf{x}_i, M_i, w_i)) = (\mathbf{y} - \mathbf{x}_i)^\top M_i (\mathbf{y} - \mathbf{x}_i) - w_i.$$

Note that this distance measure includes the Laguerre distance measure as a special case (when each M_i is taken to be the identity matrix).

As stated above, Laguerre tessellations provide good approximations of the structure of many polycrystalline materials. However, they are unable to model non-convex grains and curved boundaries. Johnson-Mehl tessellations also provide good approximations of many grain structures, but they are not always able to describe the curvature of cell boundaries properly; see [5]. The GBPD combines the advantages of both these models and also allows for a much wider range of curvatures than can be obtained using the Johnson-Mehl model. Thus, the GBPD is an ideal choice for describing non-convex grains with curved boundaries, especially when studying phenomena such as grain growth in 3D, where the geometry of the grains is very important. For a comparison of fits obtained by a number of tessellation models, which demonstrates the potential of local metric models such as the GBPD, see [5].

2.2. *Representation and properties of GBPDs*

It can be difficult to work directly with the matrices defining the GBPD. Fortunately, however, these matrices can be represented in terms of ellipsoids centered at the locations of the generators. By working with the ellipsoid representation, it is possible to make changes to the matrices that ensure they remain positive definite. In addition, the ellipsoid representation provides clear intuition about how to make small perturbations to these matrices (which will be useful when fitting the model to data).

Recall that a positive definite matrix, M, can be thought of as an ellipsoid centered at the origin; see, e.g., [22]. The eigenvectors of M correspond to the principal axes of the ellipsoid and the reciprocals of the eigenvalues give the squared lengths of the semi-axes. Such an ellipsoid can be described by six parameters. Three values, $a_1, a_2, a_3 \in \mathbb{R}_+$, describe the lengths of the semi-axes of the ellipsoid. Three values, $\phi \in [0, 2\pi]$, $\theta \in [0, \pi]$ and $\psi \in [0, 2\pi]$, are Euler angels, describing the orientation of the ellipsoid. These describe, respectively, a rotation about the z-axis by ϕ, a rotation around the x-axis by θ, and another rotation about the z-axis by ψ. The eigenvectors of the matrix M are given by the column vectors of the resulting rotation matrix

$$R = \begin{bmatrix} \cos\phi\cos\psi - \cos\theta\sin\phi\sin\psi & -\cos\psi\sin\phi - \cos\phi\cos\theta\sin\psi & \cos\phi\sin\theta \\ \cos\psi\sin\phi + \cos\phi\cos\theta\sin\psi & \cos\phi\cos\theta\cos\psi - \sin\phi\sin\psi & -\cos\phi\sin\theta \\ \sin\theta\sin\psi & \cos\psi\sin\theta & \cos\theta \end{bmatrix}.$$

Note that we order the axes by their lengths, so that a_1 gives the length of the longest axis, a_2 gives the length of the second longest axis and so on. Using this description of M, each generator of the GBPD can be described by 10 parameters: the location of the generator, (x, y, z); the semi-axes lengths of the ellipsoid, (a_1, a_2, a_3); the

Euler angles, (ϕ, θ, ψ); and the weight, w. That is, the generators can be thought of as a marked point pattern of the form

$$\mathcal{P} = \{(x_i, y_i, z_i, a_{1,i}, a_{2,i}, a_{3,i}, \phi_i, \theta_i, \psi_i, w_i)\} \subset \mathbb{R}^3 \times \mathbb{R}^3_+ \times [0, 2\pi] \times [0, \pi] \times [0, 2\pi] \times \mathbb{R}.$$

A special case of the GBPD is obtained when the ellipsoids describing the generators are non-overlapping and the weights are set to zero. In this case, the GBPD can be interpreted as being generated by growing ellipsoids, with the centroids of the ellipsoids acting as nucleation sites. All grains begin growing at the same time. The velocity with which a grain grows in a given direction can be parametrized by the Euler angles and semi-axis lengths of the corresponding ellipsoid. When two grains come into contact with one another, they stop growing at that point. Note that the general GBPD model cannot be described in this fashion, as the weights are typically non-zero and the ellipsoids may be overlapping. However, the ellipsoidal growth model provides some intuition about how the parameters of the GBPD relate to the resulting tessellation. For example, grain shapes and sizes are closely related to the shapes and sizes of the corresponding ellipsoids. A heuristic fit for the parameters of the ellipsoidal growth model can be obtained by approximating each grain by an ellipsoid (using an approach that is described in detail in Section 3.3.1). We use such a fit to choose the initial parameters for our GBPD approximation (i.e., we begin with non-overlapping ellipsoids and all weights set to zero). We then carry out our fitting procedure to obtain non-zero weights and to change the sizes, shapes and orientations of the ellipsoids. For more information on the ellipsoidal growth model, see [7].

3. Fitting GBPDs to the image data

In order to describe a microstructure observed in tomographic image data by a tessellation, a set of generators needs to be found such that the resulting tessellation is a close approximation of the 3D image data. Typically, this 3D image data is preprocessed before a tessellation model is fitted to it. This preprocessing results in a segmented image consisting of N grains, which we enumerate as $1, \ldots, N$. We treat this image as an array of labeled voxels, $I = \{I(x, y, z)\}$, indexed by x, y and z coordinates. These coordinates are all assumed to lie within a bounded window, W, and form a grid $W' \subset W$. The label of each voxel corresponds to the grain to which it belongs. So, for example, $I(x, y, z) = i$ indicates that the voxel at coordinates (x, y, z) belongs to grain i.

Given the image data, the problem of fitting a tessellation model is essentially an optimization problem. The generators, $\mathcal{P}$, should be chosen to minimize some measure of discrepancy between the generated tessellation and the tomographic image data, I. That is, $\mathcal{P}^*$ must be found, where

$$\mathcal{P}^* \in \underset{\mathcal{P}}{\operatorname{argmin}}\ \mathcal{D}_I(\mathcal{P}), \tag{1}$$

with $\mathcal{D}_I$ an appropriately chosen discrepancy measure. This is, in general, a very difficult problem as it is extremely high-dimensional and has many local minima. This means that local search optimization algorithms, such as gradient descent, may not be able to find good approximate solutions to the problem; see, e.g., the

discussion in [12]. Linear programming has been successfully applied to fit GBPDs to image data; see [8]. However, because the dimension of the problem grows very quickly in the image size and number of grains, it is restricted to small data sets. For these reasons, we use a stochastic optimization method, which is able to escape local minima. Stochastic optimization methods have been successfully used to fit Laguerre tessellations to 2D slices (see [11]) and tomographic image data; see [12]. Fitting GBPDs is more challenging, because the problem is higher dimensional and the discrepancy measure is computationally expensive to evaluate.

3.1. *Measuring the discrepancy*

We adopt a volume-based discrepancy measure, the number of incorrectly assigned voxels, in order to evaluate the goodness of fit. More precisely, we consider

$$\mathcal{D}_I(\mathcal{P}) = \#\left\{(x,y,z) \in W' : I(x,y,z) \neq I_{\mathcal{P}}(x,y,z)\right\},$$

where I is the 3D image data and $I_{\mathcal{P}}$ is the voxelized version of the tessellation generated by $\mathcal{P}$. Although, in many cases, such a discrepancy measure is inadvisable (see [12]), it makes sense to use this measure in our case for two reasons. The first is that the methods used in [12] to calculate an interface-based discrepancy do not translate easily to the GBPD case, where the surfaces separating cells are not planar. The second reason is that, in each step of our fitting procedure, we only make local changes to the generator configuration, $\mathcal{P}$. Because each change is local, we usually only need to evaluate a small number of voxels in order to evaluate the corresponding change in the discrepancy.

Simulated annealing, the stochastic optimization method used to solve (1), requires a very large number of evaluations of the discrepancy. As such, it is critical that $\mathcal{D}_I$ can be calculated extremely quickly. In order to achieve this speed of evaluation, we make a number of assumptions that simplify calculations significantly. In practice, these assumptions do not always hold. This means that the fits achieved may not correspond to the best possible fits. However, the resulting fits still significantly outperform fits achieved using the heuristic alternative; see Section 5. If necessary, the fits can be improved by removing some of the assumptions, which will be marked as such below. However, the resulting increases in computational cost may be prohibitive.

In practice, we do not need to calculate the discrepancy of a given generator set, $\mathcal{P}$, but rather $\Delta\mathcal{D}_{\mathcal{P},\mathcal{P}'} = \mathcal{D}_I(\mathcal{P}') - \mathcal{D}_I(\mathcal{P})$, the difference in discrepancy between $\mathcal{P}$ and an alternative generator set, $\mathcal{P}'$, where just one generator has been changed. In order to calculate $\Delta\mathcal{D}_{\mathcal{P},\mathcal{P}'}$, we only consider voxels in the immediate vicinity of the grain whose generator has been changed. We do this by using a cube that roughly approximates the boundaries of this grain. When calculating the discrepancy, we begin by considering voxels on the surface of this cube. We count the number of voxels that have changed to be correctly assigned and the number of voxels that have changed to be incorrectly assigned. We then move outside this cube, adding layers of voxels and counting voxels that have been changed, until we reach a layer where no voxels have changed. Likewise, we move inside the cube, subtracting layers of voxels, until we reach a layer where no voxels have changed. The change in discrepancy, $\Delta\mathcal{D}_{\mathcal{P},\mathcal{P}'}$, is then given by the total number of voxels that have changed to be correctly assigned minus the total number of voxels that have changed to be incorrectly assigned. The cubes corresponding to the grains are stored in memory

and are recomputed every 5×10^5 steps of the fitting procedure.

The cube approach significantly reduces computation time. In doing so, it makes the simplifying assumption that the cell corresponding to each generator is connected. This is not always the case. However, our results suggest that, if disconnected cells occur, there are usually only a few voxels separate from the main part of the cell. If disconnected cells appear to be a significant issue, the shell approach can either be modified to check voxels in a larger area around the generator or it can be abandoned altogether.

In computing the change in discrepancy, we make a further simplifying assumption. Initially, for each voxel, a list of the closest G_{max} generators is identified using the GBPD distance measure described in Section 2. This list is stored in memory during the whole fitting procedure. The closest G_{check} generators from these G_{max} then form the list of generators to be checked in each step. Every N_{update} steps, the distance of the voxel to each of the G_{max} generators is recomputed, and the G_{check} closest generators form the new list of generators to be checked. Note that this increases the memory requirements, but decreases computation time significantly. We have encountered no memory issues with our implementation, which uses $G_{\mathsf{max}} = 40$, $G_{\mathsf{check}} = 15$, and $N_{\mathsf{update}} = 5000$, on a 8GB RAM system.

3.2. *Simulated annealing*

Simulated annealing is a widely used stochastic optimization technique, introduced in [23] and [24], that is able to solve high-dimensional multimodal optimization problems. It draws its inspiration from the physical process of annealing, in which a material is heated up to the temperature where its molecules are easily able to move about, then slowly cooled so that the molecules arrange themselves into a low energy configuration; see, e.g., [25].

Formally, simulated annealing is a Markov chain Monte Carlo (MCMC) technique. In our setting, it works as follows. Suppose that we wish to minimize a function, $\mathcal{D} : \mathcal{X} \to \mathbb{R}$, defined on some general state space $\mathcal{X}$. Further, suppose that this function is reasonably well-behaved and has at least one global minimum. Then, the idea of simulated annealing is to sequentially sample from a family of densities, $\{\pi_T\}$, on $\mathcal{X}$, indexed by *temperature*, T, where

$$\pi_T(\mathbf{x}) = \frac{\exp\left(-\mathcal{D}(\mathbf{x})/T\right)}{\int_{\mathcal{X}} \exp\left(-\mathcal{D}(\mathbf{y})/T\right) \, \mathsf{d}\mathbf{y}} \quad \text{for all } \mathbf{x} \in \mathcal{X},$$

A considerable body of mathematical research has shown that samples from π_T will, with high probability, be very close to global minimizers of $\mathcal{D}$ when T is small; see, e.g., [26] and [27]. In practice, however, it is difficult to sample directly from π_T for low values of T. Instead, an MCMC sampler is used to sample from a distribution whose temperature is gradually lowered. This is done according to a *cooling schedule*, $\{T_n\}$, with $T_n \geq T_{n+1}$ for all $n \geq 1$. In the following, we use a Metropolis-Hastings sampler with *proposal kernel* $Q(\mathbf{x}, \mathbf{x}')$. Here, the proposal kernel can be thought of as a conditional probability density given $\mathbf{x}$, from which the proposed next state $\mathbf{x}'$ of the chain is sampled. Note that, in the classical version of simulated annealing, the proposal kernel is symmetric, i.e., $Q(\mathbf{x}, \mathbf{x}') = Q(\mathbf{x}', \mathbf{x})$, which means the likelihood of proposing a move from $\mathbf{x}$ to $\mathbf{x}'$ is the same as the likelihood of proposing a move from $\mathbf{x}'$ to $\mathbf{x}$. However, in our version of the algorithm, the proposal kernel is

not symmetric. Because of this, the acceptance probability calculated below differs slightly from the one found in standard simulated annealing algorithms. Given *initial conditions*, $\mathbf{x}_{\mathsf{init}}$, the procedure is as follows:

(1) Set $\mathbf{x}_0 = \mathbf{x}_{\mathsf{init}}$. Set $n = 1$.
(2) Sample $\mathbf{x}'$ from the density given by $Q(\mathbf{x}_{n-1}, \cdot)$.
(3) Simulate U from a uniform distribution on $(0, 1)$. Calculate the acceptance probability

$$\alpha\left(\mathbf{x}_{n-1}, \mathbf{x}'\right) = \min\left\{\exp\left(-\frac{\mathcal{D}(\mathbf{x}') - \mathcal{D}(\mathbf{x}_{n-1})}{T_n}\right) \cdot \frac{Q(\mathbf{x}', \mathbf{x}_{n-1})}{Q(\mathbf{x}_{n-1}, \mathbf{x}')}, 1\right\}.$$

If $U < \alpha\left(\mathbf{x}_{n-1}, \mathbf{x}'\right)$, set $\mathbf{x}_n = \mathbf{x}'$. Otherwise, set $\mathbf{x}_n = \mathbf{x}_{n-1}$.
(4) Stop if a prespecified stopping condition is met. Otherwise, set $n = n + 1$ and repeat from 2.

The factors controlling the performance of the simulated annealing algorithm are the initial conditions, the cooling schedule, the choice of proposal kernel and the stopping conditions. If the cooling schedule is chosen appropriately, it can be shown that $\mathbf{x}_n$ converges in distribution to the uniform distribution on the set of global minima as $n \to \infty$; see, e.g., [28] and [29]. In practice, such a slow cooling schedule is not feasible. However, simulated annealing performs very well with much faster cooling schedules; see, e.g., [30].

3.3. *Fitting GBPDs with simulated annealing*

In order to fit a GBPD to the image data, simulated annealing is used to minimize $\mathcal{D}_I(\mathcal{P})$. The proposal is generated by modifying one generator. The probability distribution, $\mathbf{p}(\mathcal{P})$, according to which the generator is chosen depends on the current configuration of generators, $\mathcal{P}$. At the nth step, we first choose a generator randomly according to the distribution $\mathbf{p}(\mathcal{P}_{n-1})$, then update the parameters in a symmetric fashion (i.e., such that the likelihood of changing the parameters from $\mathbf{x}$ to $\mathbf{x}'$ is the same as the likelihood of changing the parameters from $\mathbf{x}'$ to $\mathbf{x}$). Because only one generator is changed in each step, we can use the procedure for calculating $\Delta\mathcal{D}_{\mathcal{P},\mathcal{P}'}$ described in Section 3.1. Given initial conditions, $\mathcal{P}_{\mathsf{init}}$, the fitting algorithm is as follows:

(1) Set $\mathcal{P}_0 = \mathcal{P}_{\mathsf{init}}$. Set $n = 1$.
(2) Choose the generator, i, to modify according to the distribution given by

$$\mathbf{p}(\mathcal{P}_{n-1}) = (p_1(\mathcal{P}_{n-1}), \ldots, p_N(\mathcal{P}_{n-1})).$$

Update the values of the chosen generator in a symmetric fashion. This results in a new configuration, $\mathcal{P}'$.
(3) Simulate U from a uniform distribution on $(0, 1)$. Calculate the acceptance probability

$$\alpha\left(\mathcal{P}_{n-1}, \mathcal{P}'\right) = \min\left\{\exp\left(-\frac{\Delta\mathcal{D}_{\mathcal{P},\mathcal{P}'}}{T_n}\right) \cdot \frac{p_i(\mathcal{P}')}{p_i(\mathcal{P}_{n-1})}, 1\right\}.$$

If $U < \alpha\left(\mathcal{P}_{n-1}, \mathcal{P}'\right)$, set $\mathcal{P} = \mathcal{P}'$. Otherwise, set $\mathcal{P}_n = \mathcal{P}_{n-1}$.

(4) If a prespecified stopping condition is met, stop. Otherwise, set $n = n+1$ and repeat from 2.

The key features in this approach are: the choice of initial conditions; the choice of generator to be updated; the method used to update the generator; the choice of cooling schedule; and, the choice of stopping conditions.

3.3.1. Initial conditions

A good choice of initial conditions increases the speed of convergence of the simulated annealing algorithm and helps ensure that it finds a configuration of generators as close as possible to a global optimum. In our procedure, we first set the weights of the generators to zero (i.e., we set $w_i = 0$ for $i = 1, \ldots, N$). Then, we follow the heuristic approach used in [8] and [5], fitting an ellipsoid to each grain in the image data. In this approach, the component voxels of each grain in the image data are treated as a sample of points in $\mathbb{R}^3$. The component-wise average of these points provides an estimate for the center of mass of the corresponding ellipsoid. The ellipsoid is then obtained using *principal components analysis* (PCA). First, C, the 3×3 covariance matrix of the sample points, is estimated. Then, normalized eigenvectors of C are calculated, such that $UCU^\top = \Lambda$, where U is the matrix with the eigenvectors as columns and $\Lambda = \mathsf{diag}(\lambda_1, \lambda_2, \lambda_3)$ is the diagonal matrix of the corresponding eigenvalues. These normalized eigenvectors give the principal axes of the ellipsoid. The reciprocals of the eigenvalues, λ_1, λ_2 and λ_3, give the corresponding squared semi-axis lengths of the ellipsoid. These are rescaled such that the volume of the ellipsoid matches the volume of the corresponding grain. For more details on the link between PCA and ellipsoids, see, e.g., [31].

3.3.2. Choice of generator to update

The choice of generator to update in each step helps determine the efficiency of the simulated annealing procedure. In order to maximize the speed of convergence, it is desirable to concentrate the computational effort on generators that appear to be a long way from their optimal values (i.e., generators corresponding to grains that do not fit the image data well). At the same time, it must be possible to change the values of any generator, because escaping a local minimum may involve shifting generators away from locally optimal values. We consider three different approaches for randomly sampling the generator to modify.

- S_1: Select the generator uniformly at random. In this case, $p_i(\mathcal{P}) = N^{-1}$ for all $i \in \{1, \ldots, N\}$.
- S_2: Select the generator from a distribution with probabilities depending on the number of incorrectly assigned voxels in the corresponding grains. In this case, $p_i(\mathcal{P})$ is proportional to the number of incorrectly assigned voxels of the ith grain in the tessellation generated by $\mathcal{P}$.
- S_3: Select the generator from a distribution with probabilities depending on the fraction of incorrectly assigned voxels in the corresponding grains. In this case, $p_i(\mathcal{P})$ is proportional to the fraction of incorrectly assigned voxels in the ith grain of the tessellation generated by $\mathcal{P}$.

Note that with S_2 the algorithm is comparatively more likely to choose the generator of a large grain, as large grains contain many more voxels (and, thus, are more

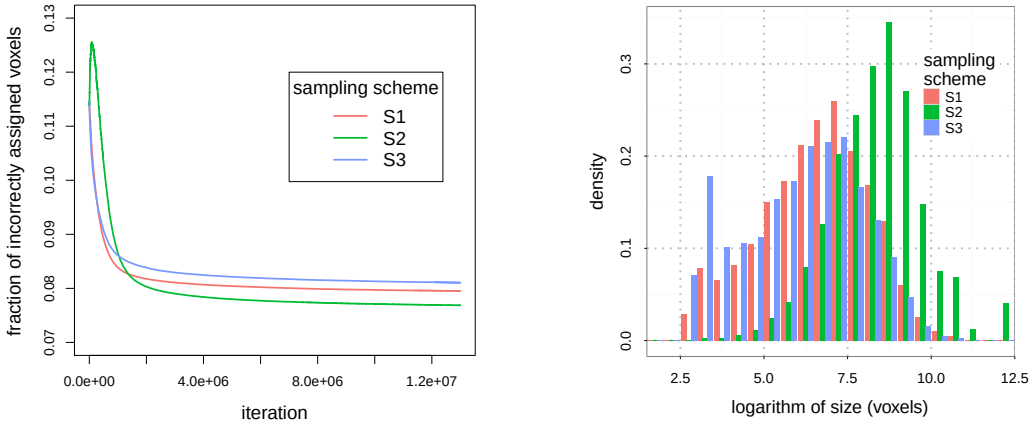

Figure 1. Performance of the sampling schemes. Left: proportion of incorrectly assigned voxels obtained using the different sampling schemes (as a function of the number of iterations). Right: histogram showing how the different sampling schemes favor different cell sizes.

likely to contain more incorrectly assigned voxels). In contrast, with S_3 it is more likely to choose the generator of a small grain, as just a few misaligned voxels may correspond to a large percentage of the voxels in such a grain. This behavior can be observed in the histogram in Figure 1.

Sampling schemes S_2 and S_3 are not symmetric because the probabilities of choosing the generators change in each step of the simulated annealing algorithm. This is corrected for in the algorithm using the $\frac{p_i(\mathcal{P}')}{p_i(\mathcal{P}_{n-1})}$ term in the acceptance probability.

In order to identify the best scheme for sampling the generator, we carried out numerical experiments. These experiments fitted GBPDs to sample B from the empirical image data described in Section 4 using the different sampling schemes. The results illustrated in Figure 1 show that sampling scheme S_2 outperforms the other two schemes. Observe that S_2 increases the probability with which large grains are chosen to be modified. Intuitively, it makes sense that this is a more effective sampling scheme. The larger grains share boundaries with many other grains and these boundaries have large areas. Thus, changes to the generators of these grains should have a large effect on the overall goodness of fit. It should be noted, however, that concentrating on the larger grains can increase the computational cost since it takes longer to evaluate $\Delta\mathcal{D}_{\mathcal{P},\mathcal{P}'}$ for large grains.

3.3.3. Updating the generator

When modifying a generator, the modification should not be too large, as proposing slight perturbations of the existing configuration helps the algorithm find local minima. However, if the perturbations are too small, the algorithm is either unable to escape local minima or does not move quickly enough towards a global minimum. We generate perturbations as follows. When a generator is selected to be modified, we modify either its position (given by x, y, z), its weight (w), the orientation of its ellipsoid (ϕ, θ, ψ), or the semi-axis lengths of its ellipsoid (a_1, a_2, a_3). The choice of which of the four to modify is made uniformly at random. Depending on which parameters are chosen to be modified, the perturbation is made according to the

rules given below.

- *Position:* If we choose to modify the position, this is done by shifting it by an amount uniformly drawn from a ball with center $(0, 0, 0)$ and radius $\bar{w} = c_p(a_1^{(0)} a_2^{(0)} a_3^{(0)})^{1/3}$, where $a_1^{(0)}, a_2^{(0)}$ and $a_3^{(0)}$ are the semi-axis lengths of the generator's ellipsoid in the initial configuration. Shifting the points in this fashion ensures that changes are symmetric, as a shift from (x, y, z) to (x', y', z') has the same likelihood as a shift from (x', y', z') to (x, y, z). Note that $2\bar{w}/c_p$ is a rough estimate of the mean width of the grain. We take a value of 0.1 for the constant c_p. Thus, we shift the position by up to 5% of the mean width each time we make a change. This shift is large enough that the simulated annealing algorithm can escape local minima but small enough that it explores points close to its current position.
- *Weight:* If the weight is chosen to be modified, the new weight is produced by randomly shifting the existing weight by a value uniformly sampled in the interval $[-c_w, c_w]$, where we take $c_w = 0.1$.
- *Orientation:* If the orientation is chosen to be modified, each Euler angle is shifted by a random amount. The shift of ϕ is uniformly selected from the interval $[-\pi/4, \pi/4]$, the shift of θ is uniformly selected from the interval $[-\pi/8, \pi/8]$, and the shift of ψ is uniformly selected from the interval $[-\pi/4, \pi/4]$. Note that these random shifts are sampled independently from each other. Furthermore, they are performed modulo the range of the corresponding angle. The ranges for the possible shifts are again chosen so that reasonably large shifts are possible but many small shifts are also made.
- *Semi-axis lengths:* If we choose to modify the semi-axis lengths, we do so by, for $i \in \{1, 2, 3\}$, randomly shifting a_i by an amount uniformly drawn from the interval $[-c_a\bar{w}, c_a\bar{w}]$, where $\bar{w}$ is as defined above and $c_a = 0.1$. If necessary, we truncate these shifts, in order to ensure that $a_i > 0$ for all $i \in \{1, 2, 3\}$ and that $a_1 \geq a_2 \geq a_3$.

Note that, with the exception of some issues that may arise with truncation, the kernels corresponding to all of these changes are symmetric. That is, the likelihood of reversing a perturbation is the same as the likelihood of making the perturbation. This means that the acceptance probability in the simulated annealing algorithm does not need to be adjusted to take the likelihoods of making various moves into account.

3.3.4. Cooling schedule

The cooling schedule of the simulated annealing algorithm is critical to its performance. If the temperature is cooled too quickly, then the algorithm can become trapped in a local minimum. However, if the temperature is not cooled sufficiently fast, the algorithm will take too long to find a sufficiently good local minimum. We use a modified geometric cooling schedule of the form

$$T_n = T_0 \kappa^{\lfloor n/k \rfloor},$$

where k and κ are parameters controlling the cooling speed and T_0 is the initial temperature. This cooling schedule only reduces the temperature every k steps. This allows the algorithm to approach the stationary limit distribution for a given value of T before the temperature is reduced again. We take k to be 12 times

the number of grains (i.e., $k = 12N$). As, at each step of the simulated annealing algorithm, one of 4 possible changes is made to a generator (either the position is changed, the orientation is changed, the semi-axes are changed or the weight is changed), this allows every parameter of each generator to be modified roughly 3 times before the temperature is reduced again. This appears to be enough steps for the MCMC sampler to get close to stationarity before the temperature is reduced again. For the cooling factor, we take $\kappa = 0.9$. The initial temperature is chosen such that approximately 50% of the modifications of the initial configuration are accepted. This is estimated by generating k perturbations of the initial configuration with one generator modified (exactly as in the simulated annealing algorithm), and then choosing T_0 so that 50% of these modifications would be accepted.

3.3.5. Stopping conditions

The criteria for choosing a stopping condition in our case are somewhat different to those typically required in the simulated annealing literature (see, for example [32]), as we do not require our algorithm to converge to a global minimum. This is because the computing time required to reach a global minimum is far too large to be practical but local minima that provide sufficiently good fits can be found within much shorter time frames. Given the cooling schedule, which ensures that the algorithm can escape local minima that correspond to bad approximations, the stopping condition should ensure that the algorithm does not run for an unnecessarily long time and, at the same time, ensure that it does not terminate while significant reductions in the cost function are still possible. Taking both these goals into account, we halt our algorithm when no reduction in the cost function occurs within k moves, where k is the number of moves carried out at each temperature level.

4. Experimental image data

In order to validate our methodology, we consider tomographic images of an aluminum alloy subjected to severe plastic deformation. These images were produced in order to investigate the properties of fine grained materials with superplastic properties. The tomographic data was acquired using 3D electron backscatter diffraction (3D EBSD). The raw image data presents a number of challenges, particularly in terms of alignment and segmentation. Our methods for overcoming these are briefly described below.

4.1. *Sample treatment and imaging*

The experimental data we consider was obtained from an aluminum alloy with nominal composition Al-3wt.%Mg-0.2wt.%Sc. The alloy was prepared by induction melting of Al 5N, a master alloy with composition Al-2wt.%Sc, and Mg 3N in a graphite crucible in an argon atmosphere (vacuum furnace Balzers VSG02). The alloy was then cast into a $14 \times 14 \times 120$ mm^3 ingot. It was homogenized for 24 h at 753 K in air, then solution treated for 1 h at 863 K and, finally, water quenched. After thermal treatment, the ingot was machined into a specimen with a cross section of 10×10 mm^2 and an approximate length of 55 mm. The specimen was then processed by 8 passes via equal-channel angular pressing (ECAP); see, e.g.,

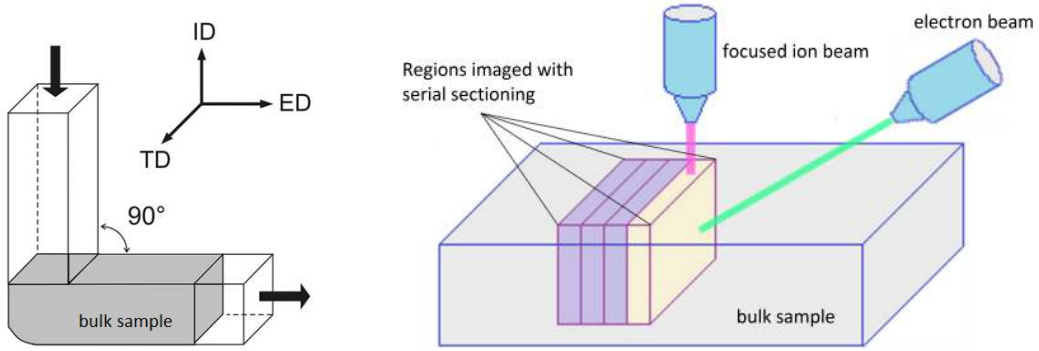

Figure 2. Left: illustration of the ECAP process. ID denotes the insert direction; ED denotes the extrusion direction; and TD denotes the transverse direction. In each pass of the ECAP procedure, the sample is pressed through the L-shaped channel. Right: schematic illustration of the set-up of the dual beam FIB-SEM system for tomographic data acquisition.

[33] for a full discussion of this procedure, which is sketched in Figure 2. After each pass of ECAP, the specimen was rotated by 90° around the axis given by the insert direction (ID). This procedure is known as route B_C. The ECAP processing was carried out at room temperature. After all the passes were finished, the specimen was annealed at 400° C for 1 hour.

The microstructure of a material changes significantly when it is annealed after ECAP. These changes are caused by a number of processes, such as static recrystallization, grain growth and decreasing dislocation density; see [34]. These complex and interacting processes resulted in a microstructure with well-defined fine grains, which can be considered as being in a state just before the intended superplastic deformation.

The imaging was done using a FEI Quanta 3D FEG field-emission scanning electron microscope (SEM) equipped with a high-speed EDAX/TSL EBSD camera and focused ion beam (FIB). The 3D observations were obtained by combining EBSD mapping with micro-milling of the top layer using FIB. The set-up is illustrated in Figure 2. The resulting images consist of a stack of 2D images of equidistantly spaced planar sections.

The three data sets of 3D observations — which we label sample A, sample B and sample C — were taken from the same small region of the material. In order to better investigate our proposed methodology, each data set was acquired using one of three perpendicular mapping planes. Sample A was obtained using a mapping plane normal to the extrusion direction (ED); sample B was obtained using a mapping plane normal to the transverse direction (TD); and sample C was obtained using a mapping plane normal to the ID. The sizes of the sub-volumes analyzed by EBSD are: $22.1 \times 24.0 \times 12.0$ μm for sample A; $27.1 \times 20.0 \times 15.0$ μm for sample B; and $23.1 \times 21.0 \times 19.0$ μm for sample C. The step sizes of the EBSD mapping and the FIB slicing are all 0.1 μm, resulting in a voxel size 10^{-3} μm^3.

4.2. *Alignment of the section planes*

The EBSD data consists of 2D images of slices of the material. These slices must be aligned to form a 3D data set. Because the material is subject to slight shifts during the imaging process, which are caused by movement of the sample between the

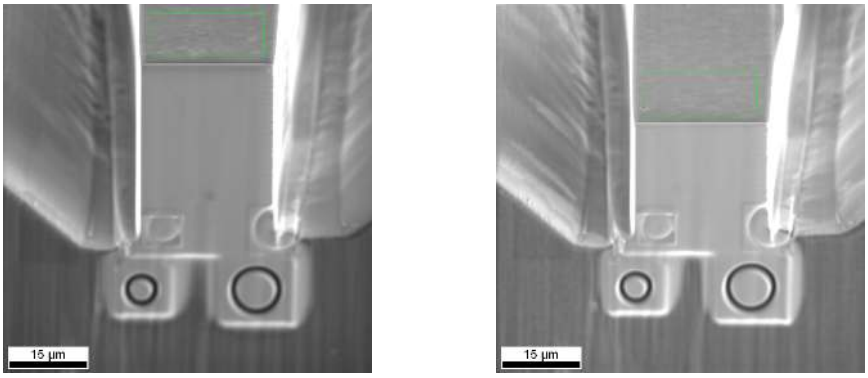

Figure 3. SEM images of sample A. The mapped area is contained in the green rectangle. Two circles are milled onto the specimen using FIB so that it can be properly aligned by image recognition software before each milling or mapping. Note that the dimensions of the mapped area appear distorted because the material is tilted by an angle of 70° to the electron beam. Left: first slice. Right: last slice.

milling and mapping positions and small drifts of the stage during time-consuming EBSD mapping, the correct alignment is not known. Thus, information from the 2D images must be used to decide how the slices should be aligned.

We first use a similar approach to the one implemented in the "align sections (misorientation)" procedure in the DREAM.3D software; see [35]. In this approach, the slices are aligned sequentially, beginning with the bottom slice. At each step, a slice is placed on top of the current slice and shifted until the discrepancy with the slice below is minimized, where the discrepancy is calculated by the number of misorientation angles between voxels in the two layers that are above a specified threshold.

Although this approach often works very well, the alignments obtained for our material are not sufficiently good. This can be seen by comparing the shifts required to obtain the alignments with estimates of the shifts obtained from SEM images of the scanned area taken for each cross section; see Figure 3. For example, using the alignment procedure, the estimated shift between the first and the last cross sections in the ED-direction for sample B is 2.30 μm. In contrast, the estimate using SEM images taken after each EBSD mapping indicates that no shift occurs.

Anisotropy in the grains of the material can cause the above procedure to fail. This is because the alignment procedure attempts to shift the layers so that, on average, the slices of each individual grain are more or less directly above the slices in the layer below. When a material is isotropic, this will lead to a good alignment. However, when anisotropy is present in the material, grains will tend to slant in a particular direction. In this case, the above procedure will artificially straighten the grains (as it does not recognize they should slant), leading to a bad alignment. Using methods from stochastic and convex geometry (see, for example, [36, 37]), we identified significant morphological anisotropy in the material. This anisotropy can also be seen in inverse pole figure (IPF) maps of the top layers of the material, which are shown in Figure 4

Because the SEM images are of significantly lower resolution than the EBSD maps, it is not possible to identify small shifts in the slices with magnitudes of only a few voxels. For this reason, we use a combination of information from the SEM images and information from the tomographic image data in the form of EBSD maps to perform the alignment. The information from the SEM images prevents misalignment due to anisotropy, while the information based on the misorientation

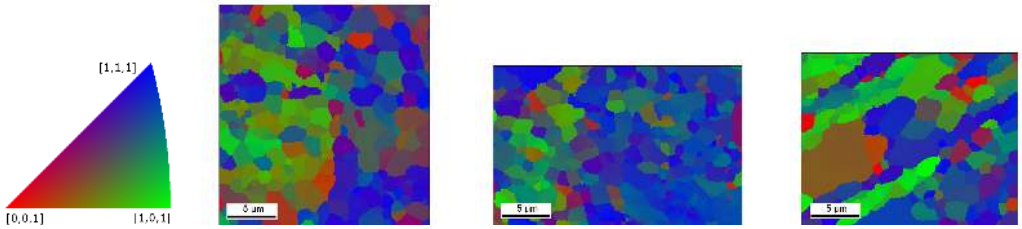

Figure 4. Inverse pole figure maps of the top layers of the three 3D samples. The colors represent the orientation of the crystal c-axis in the inverse pole figure scheme on the left. Note that anisotropy is present in all of the images and is particularly pronounced in sample C.

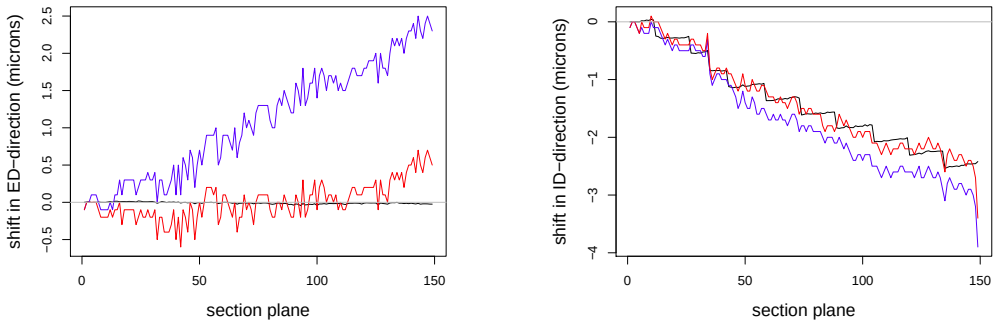

Figure 5. Relative shifts for sample B (with respect to the position of the first slice). Left: ED-direction; right: ID-direction. Black: the shifts estimated from the SEM images; blue: the shifts obtained using the alignment algorithm implemented in [35]; red: the shifts obtained using the modified alignment algorithm.

angles between slices in the tomographic data provides information about the alignment beyond the resolution of the SEM images. The basic idea for combining this information is to try to minimize the sum-of-squares distance between our alignment and the alignment implied by the SEM images. The actual alignment algorithm is quite sophisticated, and will be the subject of a future paper.

The shifts obtained using the final alignment algorithm are shown in Figure 5, along with shifts obtained by the initial algorithm and shifts estimated from the SEM images. Note that the shifts obtained using the new alignment method are very close to those estimated from the SEM images. However, the corrected shifts are more accurate, as they use information about the crystallographic orientations of the individual voxels mapped by 3D EBSD in addition to the lower-resolution SEM images, whose main purpose is the extraction of a systematic trend in the shifts.

4.3. *Segmentation of the grains*

The segmentation of a 3D image of a polycrystalline material into grains can be challenging. One reason is that the requirement that grains are completely enclosed by well-defined grain boundaries is often in conflict with the requirement that a grain should be an area with more or less identical crystal lattice orientation. This is particularly the case in deformed materials, which often exhibit gradual changes in crystal lattice orientation. When dealing with such materials, common segmentation approaches, which focus on identifying grain boundaries, can produce grains

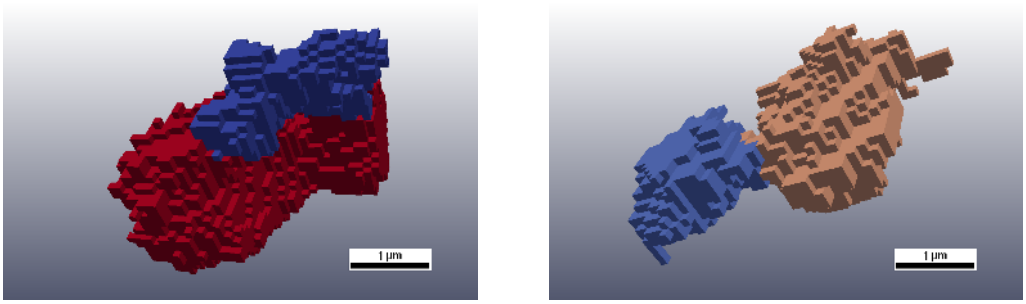

Figure 6. Grains obtained using segmentation and clustering. The different colors distinguish the individual segments used as inputs of the clustering algorithm. Left: a typical grain obtained by clustering two segments. Right: an unrealistic grain, which is not allowed in our procedure due to the minimum shared surface area condition.

with very high orientation spreads; see, e.g., [38]. Our approach aims to avoid this issue. It divides the material into grains by identifying regions where within-grain deviations in crystal orientation are negligible while between-grain differences exceed a threshold used for identifying low angle grain boundaries (LAGBs).

The segmentation proceeds in two steps. Firstly, a voxel-by-voxel segmentation based on misorientation angles is carried out. This builds grains one voxel at a time, with two neighboring voxels placed in the same grain if their misorientation angle is below a certain threshold. The segmentation is performed using an exceptionally low threshold of 0.5°, which is comparable to the measurement error. This results in over-segmentation. Secondly, a simple clustering algorithm is applied to combine the segments into grains. In each step of the clustering algorithm, the two neighboring segments with the smallest difference in average orientation are identified. If the size of the shared surface area of these grains is more than 3% of the smaller of the two grains' surface areas, then the grains are combined. Otherwise, the segments with the next smallest difference in average orientation are selected and the procedure is repeated. The minimum shared surface area condition helps to avoid the production of unrealistic grain shapes, such as the one shown at the right-hand side of Figure 6. The clustering process continues until the smallest difference in average orientation between any two neighboring grains is bigger than 2°, which is a threshold at which LAGBs can be reliably identified. The reader is referred to [39] for discussion on angular resolution and other issues related to the reconstruction of grains from EBSD data.

After segmentation, sample A contains 3052 grains, sample B contains 2828 grains, and sample C contains 3598 grains. Thus the number of parameters that need to be fitted ranges from roughly 28000 to 36000. The sizes of the samples are sufficient for assessing the qualities of the fits obtained by our procedure, as each sample contains a large number of grains and several million voxels. We do not address the question of whether these samples are large enough to be representative volume elements for, e.g., the analysis of superplastic behavior. However, we note that discussion in [40] and the references therein suggest the number of grains required for a representative texture ranges from 1000 to 10000.

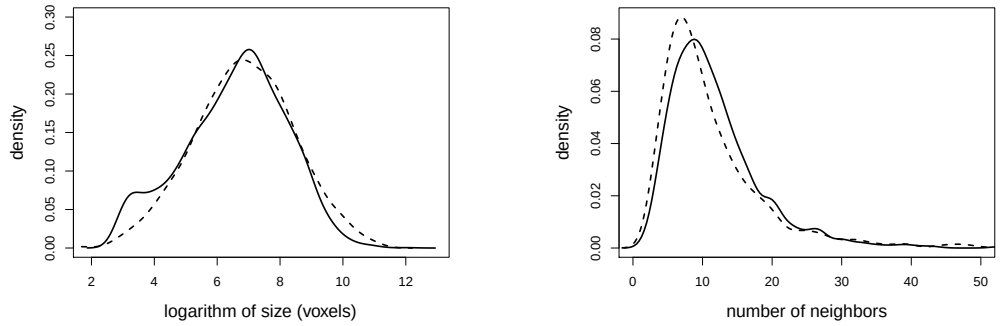

Figure 7. Comparison of the empirical distributions of basic characteristics in sample B (full line) and the simulated data (dashed line). Left: grain sizes; right: number of grain neighbors.

5. Experimental Results

A number of numerical experiments were carried out in order to evaluate both the descriptive power of GBPDs and the effectiveness of our fitting procedure. These were performed using both artificially generated image data and the experimental image data described in Section 4. The computations were performed on a low-end workstation with an Intel Xeon E3-1240 processor with four 3.4 GHz cores and 24 GB RAM. For our samples, simulated annealing required roughly 19 hours per 10 million iterations. As a result, the total computation time for each sample was less than two days. Note that very good fits can be achieved in significantly shorter time frames (for example, by relaxing the stopping criterion).

5.1. *Artificial data*

In order to measure the effectiveness of our fitting procedure, we need to be able to distinguish between errors caused by failings of the fitting procedure and errors caused by the fact that the GBPD may be an inappropriate model of the material. For this reason, we first consider an artificial data set that is known to be described perfectly by a GBPD. This also allows us to measure the effectiveness of our fitting approach in a setting with no experimental error and no error caused by the GBPD being an approximative description of the data.

The artificial data is produced in two steps. First we create the generators of a GBPD. Then, we make a voxelized version of this GBPD. In the first step, we obtain a packing of ellipsoids using the stochastic microstructure model implemented in the DREAM.3D software; see [35]. The ellipsoids serve as a basis for our GBPD. However, in order to obtain a general GBPD, we also need weights. We draw these weights (which may be negative) from a normal distribution. The parameters of the normal distribution are estimated from the tessellations fitted to the experimental data in Section 5.2. We use a normal distribution because the weights in the experimental data appear to be approximately normally distributed. In the second step, the voxelized tessellation is produced by assigning each voxel to the cell of the generator to which it is closest using the GBPD distance measure.

We carried out a number of checks to ensure that the artificial data had similar characteristics to the experimental data. Two such characteristics — grain size

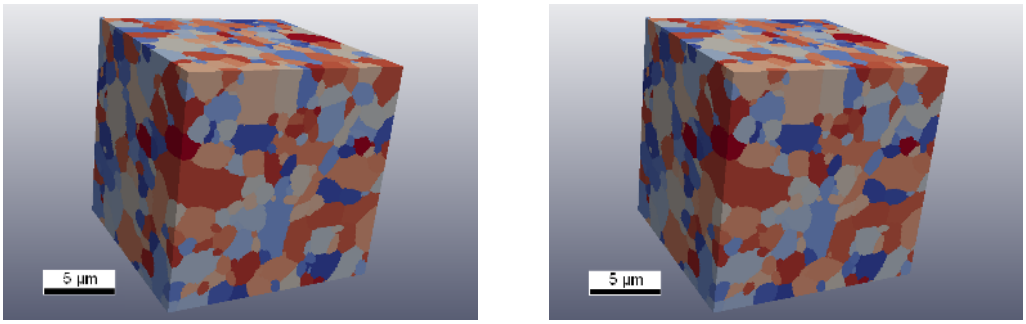

Figure 8. Left: a realization of the stochastic model fitted to sample B. The dimensions of the image are $180 \times 180 \times 180$ voxels, corresponding to $18 \times 18 \times 18 \mu m^3$. Right: the fitted GBPD obtained using the simulated annealing methodology.

Table 1. Statistics on quality of fit for the GBPD approximations of the artificial data.

	Heuristic	Simulated annealing
% of correct voxels	94.48	97.51
% of cells with all neighbors correct	47.94	60.35
% of cells with ≤ 1 incorrect neighbors	79.93	88.28
% of cells with ≤ 2 incorrect neighbors	92.13	96.15
mean number of erroneous neighbours per grain	0.87	0.57

distribution and number of neighbors distribution — are shown in Figure 7, where characteristics estimated from artificial data are compared to those from sample B. Thus, the artificial data can be thought of as image data, free of experimental error, of a microstructure that has similar characteristics to those observed in empirical data and that is perfectly described by a GBPD. Any error in fitting a GBPD to this image data can be solely attributed to our fitting procedure.

Our fitting procedure, described in Section 3, was used to fit a GBPD to artificial image data consisting of 180^3 voxels and containing 1894 grains. The parameters used for the fitting algorithm are as described in Section 3.3. The initial temperature used was $T_0 \simeq 84.3$. The initial configuration was obtained using the heuristic procedure described in Section 3.3.1. The results of the fitting procedure are visualized in Figure 8. The left-hand side of Figure 8 shows artificial image data generated by the stochastic GBPD model. The right-hand side of Figure 8 shows the corresponding reconstruction using our simulated annealing method. Visually, the original data and the GBPD reconstruction appear almost identical.

The results for both the heuristic initial fit and the fit achieved using simulated annealing are summarized in Table 1. Note that the heuristic fit is already quite good with 94.48% of voxels correctly assigned. Using simulated annealing, the number of correctly fitted voxels improves to 97.51%. The neighborhood structure is reasonably well modeled by the heuristic fit, with 47.94% of cells having all neighbors correct and 92.13% of cells having at most two neighbors incorrectly assigned. Simulated annealing improves on this considerably, with 60.35% of cells having all neighbors correctly assigned and 96.15% of cells having at most two incorrect neighbors. Note that, although the simulated annealing algorithm does not result in a perfect reconstruction, the quality of the fit is impressive given that the optimization problem is extremely high-dimensional (and, thus, it is almost impossible to

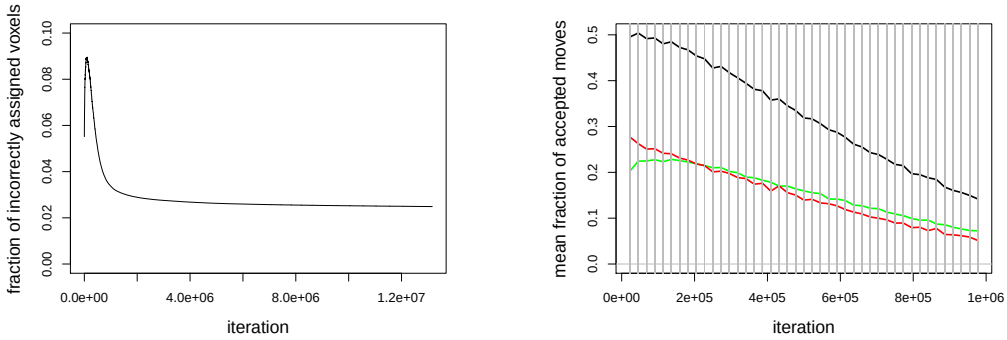

Figure 9. Convergence behavior of the simulated annealing algorithm for the artifical data set. Left: the proportion of incorrectly fitted voxels over time. Right: the proportion of moves accepted as a function of time (black: total moves accepted; green: proportion of moves accepted that decrease the cost function; red: proportion of moves that increase the cost function). The vertical lines show the points at which the temperature changes. Note that only the first 1 million steps are plotted.

find a global minimum corresponding to a perfect fit).

The convergence behavior of the fitting algorithm is shown in Figure 9. The left-hand side of Figure 9 shows the proportion of voxels that are incorrectly assigned as a function of the number of steps of the simulated annealing algorithm. Note that, initially, there is a decrease in the quality of the fit. This is followed by a steady improvement. The algorithm terminates after approximately 13 million steps at a point where the cost function appears to have flattened off and where no significant further improvement seems possible.

5.2. *Experimental data*

The main focus of this paper is on fitting GBPD approximations to tomographic images of polycrystalline materials. In this section, we examine the effectiveness of this method by fitting GBPDs to three 3D EBSD images of Al-3wt.%Mg-0.2wt.%Sc alloy. The generation and preprocessing of these 3D data sets is described in Section 4. After alignment, the 3D images are in irregularly shaped windows. To simplify computation and visualization, we have taken cubic cutouts of these images. The cubes have been chosen to maximize the volume of the cutouts. Each cube contains more than 75% of the volume of the original 3D image.

GBPDs were fitted to each of the tomographic 3D images using the simulated annealing methodology. The results are summarized in Figure 10 and Table 2. Figure 10 shows a 3D image of sample C, together with the GBPD reconstruction obtained using simulated annealing. Notice that, visually, the fit seems to be almost perfect, with the GBPD reconstruction a slightly smoother version of the original image data. Indeed, the statistics given in Table 2 show that the quality of the fits is good. As in the artificial data case, the use of simulated annealing increases the number of correctly assigned voxels by $3-4\%$ in comparison with the heuristic fit. Note that this improvement is concentrated at the boundaries of the grains. This is apparent if we consider the results in the immediate vicinity of the grain boundaries, which we define to be all voxels in the image data that share a face, edge or vertex with a voxel from another grain. These voxels comprise roughly 45% of the

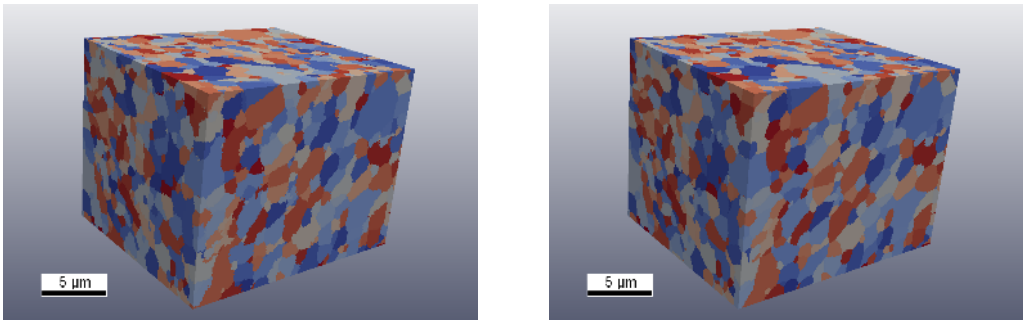

Figure 10. Left: 3D image of sample C. Right: 3D image of the GBPD approximation achieved through the simulated annealing fitting procedure.

Table 2. Statistics on quality of fit for the GBPD approximations of the 3D data. Here, (H) denotes the heuristic fit and (S) denotes the simulated annealing fit. A, B and C denote the data sets as introduced in Section 4.1.

	A(H)	A(S)	B(H)	B(S)	C(H)	C(S)
% of correct voxels	88.18	92.03	88.54	92.32	88.55	92.12
% of correct voxels on grain boundaries	75.43	82.63	75.45	82.62	75.99	82.76
% of cells with all neighbors correct	33.81	35.06	36.59	38.38	32.42	34.35
% of cells with ≤ 1 incorrect neighbors	66.45	67.42	68.37	72.60	63.50	65.86
% of cells with ≤ 2 incorrect neighbors	85.29	86.03	87.58	88.92	82.14	84.12
mean number of erroneous neighbors per grain	1.31	1.26	1.21	1.09	1.42	1.32

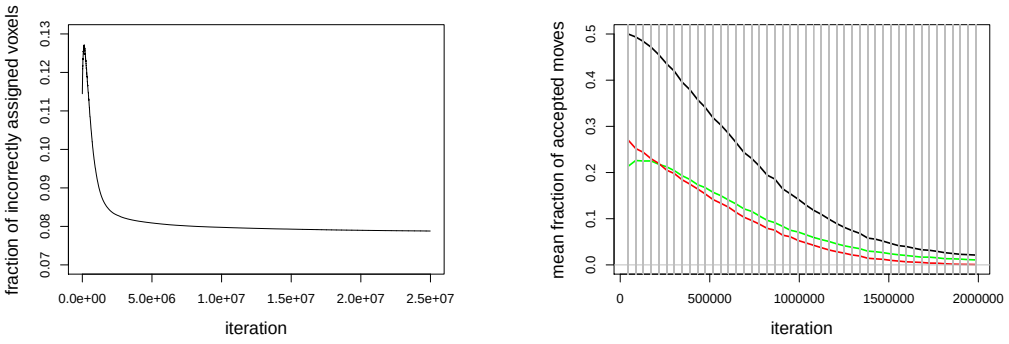

Figure 11. Convergence behavior of the simulated annealing algorithm for sample C. Left: proportion of incorrectly assigned voxels as a function of the number of steps of the algorithm. Right: the proportion of moves accepted as a function of time (black: total moves accepted; green: proportion of moves accepted that decrease the cost function; red: proportion of moves that increase the cost function). The vertical lines show the points at which the temperature changes. Note that only the first 2 million steps are plotted.

analyzed volume in each sample but make up approximately 90% of the reassigned voxels. At the grain boundaries, the increase in the percentage of correctly assigned voxels is roughly 7%.

The convergence behavior of the fitting algorithm is illustrated in Figure 11. As in the artificial data case, the algorithm terminates at a point where no further significant improvement seems possible.

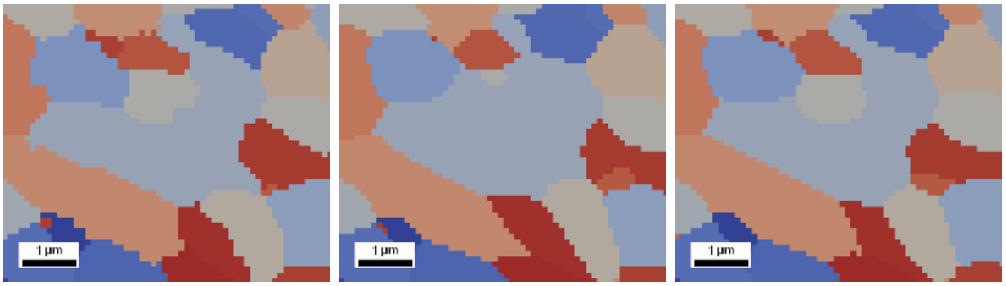

Figure 12. Left: a subsection of a 2D slice of the experimental data. Center: the GBPD reconstruction obtained using the heuristic fit. Right: the GBPD reconstruction obtained using our procedure. Our fit is much better able to describe the shape of the light blue central grain than is the heuristic fit.

5.3. *Discussion of the experimental results*

In all the examples considered above, the numerical results demonstrate an improvement in fit (measured by the percentage of correctly assigned voxels) of approximately 3% - 4% over the heuristic method of fitting ellipsoids using PCA. Although this improvement may seem to be small, it is worth observing that this improvement is almost entirely concentrated at the boundaries of the grains (as the majority of voxels within grains are already correctly assigned). At the boundaries of the grains, the percentage of correctly assigned voxels increases by approximately 7% in each sample. This improvement is quite significant, as the geometries of grain boundaries play important roles in many physical phenomena. Additionally, the heuristic fits are not able to fit as wide a range of grain shapes as the improved fitting procedure considered in the present paper. This seems to be especially the case when fitting very non-convex grains, as illustrated in Figure 12. Note that, due to the large number of grains in the samples and the high resolution of the images, it is not possible to use the linear programming method as in [8] to fit tessellations to the data we consider.

6. Conclusions and Outlook

GBPDs are a class of tessellations that are able to describe features of polycrystalline materials, such as non-convex grains and curved grain boundaries, that are not well described by traditional tessellation models such as Voronoi and Laguerre tessellations. In this paper, we introduced a new method for fitting GBPDs to tomographic image data. This method uses simulated annealing to solve a suitably chosen optimization problem. We showed how this method can be effectively implemented so as to achieve very good fits within reasonable time periods. We applied the method to artificial data generated from a stochastic GBPD model. The high quality of the resulting fits demonstrated the effectiveness of our method in reconstructing GBPDs. We then applied our method to experimentally obtained 3D EBSD data getting extremely high quality fits of much better quality than those obtained using conventional tessellation models. In addition, we described techniques used to align and segment our tomographic 3D EBSD data in preparation for fitting the GBPD.

Because grain boundaries play a key role in determining the physics of polycrystalline materials, the ability to effectively describe these boundaries using a GBPD is a promising advance in the empirical study of such boundaries. For example, es-

timators of important properties such as curvature can be developed that are based on a GBPD fitted to image data. In order to develop and understand the properties of such estimators, the theoretical properties of GBPDs must be further studied. In addition, models could be developed that are able to describe grain information such as crystallographic orientations and misorientations at grain boundaries. These models could then be used to develop stochastic models, based on random marked tessellations, that could be used for virtual materials design.

7. Acknowledgement

This research was supported by the Czech Ministry of Education, Youth and Sports (MSMT) and the German Academic Exchange Service (DAAD), project no. 7AMB14DE006, the Czech Science Foundation (GACR), project no. GBP108/12/G043, and MSMT, project no. LM2015087.

References

[1] R.R. MacPherson and D.J. Srolovitz, Nature 446 (2007) p. 1053.
[2] L. Priester, *Grain Boundaries: From Theory to Engineering*, Springer, 2013.
[3] W.A. Johnson and R.A. Mehl, Transactions of the American Institute of Mining, Metallurgical, and Petroleum Engineers 135 (1939) p. 416.
[4] A. Okabe, B. Boots, K. Sugihara and S.N. Chiu, *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*, 2nd ed., J. Wiley & Sons, 2000.
[5] H. Altendorf, F. Latourte, D. Jeulin, M. Faessel and L. Saintoyant, Image Analysis & Stereology 33 (2014) p. 121.
[6] D. Jeulin, *Random tessellations and Boolean random functions*, in *Mathematical Morphology and Its Applications to Signal and Image Processing*, Springer, 2013, p. 25.
[7] K. Teferra and L. Graham-Brady, Computational Materials Science 102 (2015) p. 57.
[8] A. Alpers, A. Brieden, P. Gritzmann, A. Lyckegaard and H.F. Poulsen, Philosophical Magazine 95 (2015) p. 1016.
[9] A. Lyckegaard, E.M. Lauridsen, W. Ludwig, R.W. Fonda and H.F. Poulsen, Advanced Engineering Materials 13 (2011) p. 165.
[10] A. Liebscher, Philosophical Magazine 95 (2015) p. 2777.
[11] A. Liebscher, D. Jeulin and C. Lantuéjoul, Journal of Microscopy 258 (2015) p. 190.
[12] A. Spettl, T. Brereton, Q. Duan, T. Werz, C. Krill, D.P. Kroese and V. Schmidt, Philosophical Magazine 96 (2016) p. 166.
[13] O. Engler and V. Randle, *Introduction to Texture Analysis: Macrotexture, Microtexture, and Orientation Mapping*, 2nd ed., CRC Press, 2010.
[14] A.J. Schwartz, M. Kumar, B.L. Adams and D.P. Field, *Electron Backscatter Diffraction in Materials Science*, Springer, 2009.
[15] A.J. Wilkinson and T.B. Britton, Materials Today 15 (2012) p. 366.
[16] T.H. Scheike, Advances in Applied Probability 26 (1994) p. 43.
[17] S.N. Chiu, D. Stoyan, W.S. Kendall and J. Mecke, *Stochastic Geometry and its Applications*, 3rd ed., J. Wiley & Sons, 2013.
[18] F. Aurenhammer, Discrete & Computational Geometry 2 (1987) p. 49.
[19] C. Lautensack and S. Zuyev, Advances in Applied Probability 40 (2008) p. 630.
[20] D.J. Rowenhorst, A.C. Lewis and G. Spanos, Acta Materialia 58 (2010) p. 5511.
[21] Q. Du and D. Wang, SIAM Journal on Scientific Computing 26 (2005) p. 737.
[22] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004.
[23] V. Černý, Journal of Optimization Theory and Applications 45 (1985) p. 41.

[24] S. Kirkpatrick, C.D. Gelatt and M.P. Vecchi, Science 220 (1983) p. 671.
[25] P.J. van Laarhoven and E.H. Aarts, *Simulated Annealing: Theory and Applications*, Springer, 1987.
[26] R. Holley and D. Stroock, Communications in Mathematical Physics 115 (1988) p. 553.
[27] C.J.P. Bélisle, Journal of Applied Probability 29 (1992) p. 885.
[28] P. Brémaud, *Markov Chains: Gibbs Fields, Monte Carlo Simulation, and Queues*, Springer, 1999.
[29] S. Geman and D. Geman, IEEE Transactions on Pattern Analysis and Machine Intelligence 6 (1984) p. 721.
[30] E. Aarts and J. Korst, *Simulated Annealing and Boltzmann Machines: A Stochastic Approach to Combinatorial Optimization and Neural Computing*, J. Wiley & Sons, 1989.
[31] T.W. Anderson, Annals of Mathematical Statistics 34 (1963) p. 122.
[32] P. Siarry, G. Berthiau, F. Durdin and J. Haussy, ACM Transactions on Mathematical Software 23 (1997) p. 209.
[33] R.Z. Valiev and T.G. Langdon, Progress in Materials Science 51 (2006) p. 881.
[34] K. Dám, P. Lejček and A. Michalcová, Materials Characterization 76 (2013) p. 69.
[35] M. Groeber and M. Jackson, Integrating Materials and Manufacturing Innovation 3 (2014) p. 5.
[36] V. Beneš and A.M. Gokhale, Kybernetika 36 (2000) p. 149.
[37] M. Kiderlen and A. Pfrang, Journal of Microscopy 219 (2005) p. 50.
[38] S.W. Cheong and H. Weiland, Materials Science Forum 558-559 (2007) p. 153.
[39] F.J. Humphreys, Journal of Materials Science 36 (2001) p. 3833.
[40] A. Brahme, Y. Staraselski, K. Inal and R.K. Mishra, Metallurgical and materials transactions A 43 (2012) p. 5298.