Available online at www.sciencedirect.com
ScienceDirect
Comput. Methods Appl. Mech. Engrg. 369 (2020) 113175
Computer methods in applied mechanics and engineering
www.elsevier.com/locate/cma

Fast methods for computing centroidal Laguerre tessellations for prescribed volume fractions with applications to microstructure generation of polycrystalline materials

Jannick Kuhn[a], Matti Schneider[b,*], Petra Sonnweber-Ribic[a], Thomas Böhlke[b]

[a] Robert-Bosch GmbH, Corporate Sector Research and Advance Engineering, Germany
[b] Karlsruhe Institute of Technology (KIT), Institute of Engineering Mechanics, Germany

Received 13 March 2020; received in revised form 27 April 2020; accepted 22 May 2020
Available online 25 June 2020

Abstract

Ideas from the mathematical theory of optimal transport have recently been transferred to the micromechanics of polycrystalline materials, leading to fast methods for generating polycrystalline microstructures with grains of prescribed volume fraction in terms of centroidal Laguerre tessellations. In this work, we improve the state of the art solvers.

For a given set of seeds and corresponding volume fractions summing to unity, there is a set of Laguerre weights such that the corresponding Laguerre tessellation realizes the prescribed volume fractions exactly. Furthermore, the Laguerre weights are unique up to a constant and can be determined by solving a convex optimization problem. However, whenever the optimization algorithm encounters a weight vector leading to an empty cell, the optimization problem is no longer locally strictly convex. To account for the latter, backtracking strategies are typically employed.

We show that modern gradient-based optimization algorithms devoid of backtracking, like the Malitsky–Mishchenko method and the Barzilai–Borwein scheme easily overcome the described difficulty, leading to a significant speed-up compared to more traditional solvers. Furthermore, for computing centroidal Laguerre tessellations of prescribed volume fraction, we propose an Anderson-accelerated version of Lloyd's algorithm, and show, by numerical experiments, that it consistently reduces the run-time.

We demonstrate the capabilities of our proposed methods for generating microstructures of polycrystalline materials with prescribed grain size distribution.

Keywords: Laguerre tessellations; Polycrystalline materials; Gradient solvers; Grain size distributions

1. Introduction

1.1. State of the art

Modern imaging techniques, like serial sectioning techniques [1,2], optical microscopy [3,4], FIB-SEM [5–7], electron backscatter diffraction [8–10] and X-ray diffraction microscopy [11,12], give insights into the precise

* Corresponding author.
E-mail address: matti.schneider@kit.edu (M. Schneider).

https://doi.org/10.1016/j.cma.2020.113175

arrangement of the microstructure of polycrystalline materials. Combined with dedicated material models, these microstructures enable understanding the complex effective mechanical behavior of polycrystalline materials subjected to various loading conditions by crystal plasticity finite element simulations [13–15] or methods based on spectral methods [16–18], and, eventually, help designing industrial components in terms of their strength, durability and energy footprint .

Unfortunately, solely relying upon imaging techniques for obtaining the necessary microstructural data appears insufficient, essentially for two reasons. On the one hand, obtaining high-quality microstructural images can be laborious and costly [19], in particular in terms of laboratory equipment. On the other hand, it is difficult to cover all possible "microstructural states of interest", either because the taken images do not exhibit the degree of statistical homogeneity necessary or because the parameter space is simply too large to be sampled in reasonable time.

As a viable alternative, synthetic microstructure models for polycrystalline materials have gained popularity recently. These models are related to the imaging techniques mentioned above essentially in the same way that mechanical material models are related to mechanical experiments. Indeed, these models should depend on a limited number of physically relevant parameters, and permit reducing stochastic errors in the underlying measurements.

Microstructure models may be roughly classified [19–21] into physics-based and geometrical models. Physics-based microstructure models are typically based on a physical description of the solidification process governing the microstructure formation of the polycrystalline material under consideration. Apart from dynamic vertex methods [22–24], cellular automata [25–27] and Monte-Carlo approaches [28,29], phase-field models [30–32] may be used for creating polycrystalline microstructures.

Alternatively, geometrical models typically model the individual grains of the polycrystalline microstructures as non-overlapping polyhedra, leading to so-called tessellations. In contrast to the physics-based microstructure models, the geometrical approaches are less computationally demanding, but at the expense of less physical justification. Tessellation models may be categorized into models with convex grains and those with non-convex grains. The latter approaches are capable of generating much more realistic microstructures than when restricting to convex grains, but their fitting is either based on having the images available [33–36] or optimization schemes to match target size and shape distribution [37–42].

Among the geometrical approaches with convex grains, Voronoi tessellations [43] are the most widely-used [44–46]. Voronoi tessellations are described in terms of a set of so-called seed points. To each such seed point, the corresponding Voronoi cell encompasses all points whose closest seed point is the seed point referred to above.

Choosing the seed points of Voronoi tessellations randomly, also known as a Poisson–Voronoi tessellation, leads to microstructure models whose properties, i.e., the grain size distribution and the average number of neighboring cells, are unlike those found in real materials [47–50]. To overcome this issue, several techniques have been developed. For instance, instead of randomly-prescribed seeds, the seeds may be determined from a pre-computed sphere packing, increasing the degree of realism significantly [44,45]. As a complementary approach, one may demand that the Voronoi tessellation is centroidal, i.e., that the seeds of each cell coincide with their centroids [51]. Dedicated algorithms for generating such centroidal Voronoi have been developed, for instance in terms of the classical Lloyd algorithm [52–54], the Lloyd–Newton method [55] or Quasi-Newton methods [56].

Voronoi tessellations are a simple model for a homogeneous and isotropic grain-growth process of polycrystalline cells. However, it has been found empirically that Voronoi tessellations are unable to generate microstructures with real-world grain-size distributions, in general [47,48]. In contrast, Ball–Carstensen [57] prove that any polycrystal consisting of convex grains, the interior grains are polyhedral and form a so-called Laguerre tessellation [58–60].

In addition to a number of seeds, for Laguerre tessellations, a scalar weight corresponding to each seed has to be prescribed. This weight might be roughly thought of as a price which governs the region of influence of each seed. Indeed, by properly increasing or decreasing the "price" of a single seed (while fixing all others), the cell volume may be in- or decreased appropriately.

This intuition may be formalized mathematically, see Lautensack–Zuyev [61], who showed that any *normal* tessellation with convex cells of Euclidean d-space with $d \geq 3$ may be obtained as a Laguerre tessellation. A complementary result was established by Aurenhammer–Hoffmann–Aronov [62], who showed that, for a prescribed set of seeds within a finite domain Ω, and corresponding (non-zero) volumes summing to the volume of Ω, there is a vector of weights such that the corresponding Laguerre tessellation realizes the desired volumes. Furthermore, up to the addition of a constant, the weight vector is unique.

In addition, the article of Aurenhammer–Hoffmann–Aronov [62] provided a computational framework for determining the weight vector in question by showing that the underlying problem may be obtained from a

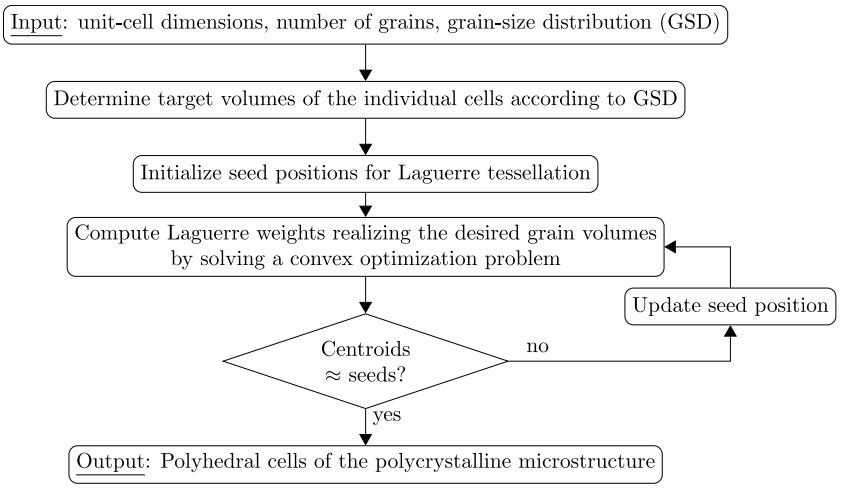


Fig. 1. General flowchart for computing Laguerre-type models of polycrystalline microstructures, following Bourne et al. [66].

variational principle with *convex* objective function. This point of view was connected to the theory of semi-discrete optimal transport by Mérigot [63], and further refined by Lévy [64] and Kitagawa–Mérigot–Thibert [65].

In a recent article, Bourne et al. [66] used this convex variational principle for generating Laguerre tessellations modeling polycrystalline microstructures with prescribed volume fractions per grain, see Fig. 1 for an overview. Their insights led to the development of *fast* techniques for generating polycrystalline microstructures which are orders of magnitude faster than other approaches in the literature, for instance based on derivative-free optimization, see Quey–Renversade [67], statistical methods [68] or cross-entropy approaches [69], and more accurate than heuristic approaches [70].

1.2. Contributions

In this work, we improve upon the fast methods of Bourne et al. [66], see Fig. 1 for an overview, as they essentially rely upon black-box optimization methods.

Bourne et al. [66] use the limited-memory Broyden–Fletcher–Goldfarb–Shanno (BFGS) method [71] with backtracking for solving the convex optimization problem. The backtracking is used to avoid generating empty cells during the iterative process, and is also used for the corresponding Newton-type methods [65].

In Section 2, we set up our notation for computing periodic Laguerre tessellations and the resulting convex optimization problem. We discuss pertinent solvers in Section 3. Of particular interest here are modern non-monotone gradient-type solvers, i.e., the methods of Malitsky–Mishchenko [72] and Barzilai–Borwein [73], whose application to optimal-transport problems appears to be novel. Compared to more traditional solvers of (Quasi-)Newton type, they combine low cost per iteration with non-monotone convergence behavior, which turns out to be beneficial for the *overall* run-time. Indeed, non-monotone methods have gained popularity in recent years [74], as the user is mostly interested in minimizing the time to solution, and ensuring a monotone convergence of the algorithm in question is typically of secondary importance.

For computing centroidal Laguerre tessellations, we review the classical Lloyd's algorithm in Section 2.2. Subsequently, we apply Anderson acceleration to Lloyd's algorithm. Anderson acceleration [75] is a dedicated technique for improving the convergence behavior of general fixed-point iterations. Its mathematical analysis is more recent, establishing it as a particular Quasi-Newton method of multi-secant type [76] and establishing the improved convergence behavior also rigorously [77,78].

In Section 4, we study the numerical effectiveness of the proposed algorithms for selected problems of increasing complexity. We study the dependence of the algorithms on the number of grains and the prescribed grain-size distribution, with and without centering. Furthermore, we exhibit their capabilities for polycrystalline materials of industrial relevance.

2. Periodic Laguerre tessellations and centering

2.1. Periodic Laguerre tessellations

Let Y be a rectangular domain in $\mathbb{R}^d$, which we fix as $Y = [0, L_1) \times [0, L_2) \times \cdots \times [0, L_d)$ for positive lengths L_i. Y shall serve as our unit cell of interest, and we furnish it with periodic boundary conditions. We denote the corresponding periodic distance function by $d : Y \times Y \to \mathbb{R}_{\geq 0}$. For a sequence $(x_1, x_2, \ldots, x_K) \in Y^K$ of distinct points lying inside Y and a given set of real weights $w \in \mathbb{R}^K$, we denote by $D_1, D_2, \ldots, D_K$ the periodic Laguerre tessellation [43] of Y with seeds $(x_1, x_2, \ldots, x_K)$ and weights $(w_1, w_2, \ldots, w_K)$. Each cell D_i is a subset of Y defined via

$$D_i = \left\{ x \in Y \mid d(x, x_i)^2 - w_i \leq d(x, x_j)^2 - w_j \quad \text{for} \quad j \in \{1, 2, \ldots, K\} \right\}. \tag{2.1}$$

The interiors of the Laguerre cells are mutually disjoint, and the Laguerre cells cover the original volume Y. Clearly, adding a constant to each single weight does not change the associated Laguerre cells. Notice that the seed x_i does not need to lie in the corresponding Laguerre cell D_i. Also, the Laguerre cell may be empty. This becomes clear when imagining the weight w_i as a "price". Loosely speaking, if the price is excessive, the turnover, which is related to the volume of the cell, will be zero.

For the special case of identical weights, $w_1 = w_2 = \cdots = w_K$, a Laguerre tessellation is called a Voronoi tessellation, see Aurenhammer [43]. For this special case, the seeds are always contained in the corresponding cell. However, the kinds of tessellations which may be generated as Voronoi tessellations are rather limited, for instance in terms of the achievable volume-fraction distributions, see section 2.1.1 in Quey–Renversade [67]. In contrast, Laguerre tessellations may generate any normal tessellation with convex cells in $\mathbb{R}^d$ for $d \geq 3$, see Lautensack–Zuyev [61]. In this context, normal means that if the seeds are in general position, each k-face of the tessellation arises as the intersection of exactly $d - k + 1$ cells for $k = 0, 1, \ldots, d - 1$.

For this article, Laguerre tessellations serve as simple geometrical models for polycrystalline microstructures, e.g., of metallic materials, alloys, ceramics or polycrystalline ice. We refer to section 2.1 in Bargmann et al. [19] for a recent overview.

For a given sequence $(x_1, x_2, \ldots, x_K) \in Y^K$ of distinct points lying inside Y and a given set of volume fractions $(\phi_1, \phi_2, \ldots, \phi_K) \in \mathbb{R}^K$, each of which is non-negative, and which sum to unity, there is a corresponding set of weights $(w_1, w_2, \ldots, w_K) \in \mathbb{R}^K$, s.t. the corresponding Laguerre cells realize the prescribed volume fractions, i.e.,

$$\frac{|D_i|}{L_1 L_2 \cdots L_d} = \phi_i, \tag{2.2}$$

see Aurenhammer–Hoffmann–Aronov [62], where $|D_i|$ denotes the volume of cell D_i. The proof of this result is interesting in its own right, as it immediately gives rise to a computational technique for computing the associated weight vector w. As noted earlier, the component-wise addition of a constant to the weight vector does not change the associated Laguerre cells. Thus, we define

$$W_K = \left\{ w \in \mathbb{R}^K \;\middle|\; \sum_{i=1}^{K} w_i = 0 \right\}, \tag{2.3}$$

the $(K-1)$-dimensional subspace of $\mathbb{R}^K$ with vanishing mean. For a fixed sequence $(x_1, x_2, \ldots, x_K) \in Y^K$ of distinct points lying inside Y and a given set of volume fractions $(\phi_1, \phi_2, \ldots, \phi_K) \in \mathbb{R}^K$, define the function

$$g : W_K \to \mathbb{R}, \quad w \mapsto \sum_{i=1}^{K} \left(\phi_i \cdot L_1 L_2 \cdots L_d - |D_i|\right) w_i + \sum_{i=1}^{K} \int_{D_i} d(x, x_i)^2 \, dx, \tag{2.4}$$

where D_i denotes the ith associated Laguerre cell (2.1). Aurenhammer–Hoffmann–Aronov [62] have shown that the function g (2.4) is a concave function which admits maximizers. Furthermore, each critical point (which is a maximum by concavity of g) satisfies the first order optimality condition

$$0 = Dg(w)[v]$$

in terms of the directional derivative of g at $w \in W_K$ in direction $v \in W_K$. More explicitly,

$$Dg(w)[v] \equiv \frac{d}{ds} g(w + sv)\big|_{s=0} = \sum_{i=1}^{K} (\phi_i \cdot L_1 L_2 \cdots L_d - |D_i|)\, v_i, \tag{2.5}$$

i.e., any critical point of g satisfies Eq. (2.2). Furthermore, it can be shown that g is even strictly concave on the set

$$\{w \in W_K \mid |D_i| > 0 \quad \text{for all} \quad i = 1, 2, \ldots, K\}, \tag{2.6}$$

see Kitagawa–Mérigot–Thibert [65]. It is clear that g is not strictly concave outside of the set (2.6). Indeed, suppose that we have $w \in W_K$, s.t. the i-the Laguerre cell is empty. Then, the Laguerre cells remain unchanged if we increase the ith weight w_i. In particular, the function g is constant along the ray

$$w + s\left[e_i - \frac{1}{K}(1, 1, \ldots, 1)\right], \quad s \geq 0,$$

where e_i is the ith unit vector in $\mathbb{R}^K$. For completeness, notice that the ith component of the gradient of g on W_K computes as

$$\nabla_i g(w) = \phi_i \cdot L_1 L_2 \cdots L_d - |D_i|, \tag{2.7}$$

cf (2.5). Notice that $\nabla g(w) \in W_K$ automatically, i.e., its mean vanishes.

The concavity of g (or, equivalently, the convexity of $-g$) permits using established convex optimization techniques as in Nocedal–Wright [79] to be applied, see Section 3 for details. To finish this paragraph, we report on a result in Bourne–Roper [54]. The function g (2.4) is even twice differentiable on W_k with second derivative

$$D^2 g(w)[v, v] = v^T H(w) v, \quad w, v \in W_K,$$

in terms of the Hessian matrix $H(w) \in \mathbb{R}^{K \times K}$ with entries

$$H_{ij}(w) = \frac{A_{ij}}{2\, d(x_i, x_j)} \tag{2.8}$$

for $i \neq j$, where A_{ij} denotes the area of the face between the ith and jth cell (which is formally set to zero if the cells do not share a face), and $H_{ii} = -\sum_{j \neq i} H_{ij}$ for $i = 1, 2, \ldots, K$.

As g is concave and twice differentiable, the Hessian matrix $H(w)$ is negative semidefinite. However, $H(w)$ does not have full rank, as all rows and columns have zero mean, leading to a zero eigenvalue. In particular, it is not invertible. As a countermeasure, we introduce the matrix

$$\tilde{H}(w) = H + \frac{\mathrm{tr}(H(w))}{K^2}\, \underline{1}\, \underline{1}^T \tag{2.9}$$

with the vector $\underline{1} = (1, 1, \ldots, 1) \in \mathbb{R}^K$ of all ones. The latter modification makes sure that

1. If the system

   $$\tilde{H}(w) w = v$$

   is solved for $\Delta w \in \mathbb{R}^K$ with $v \in W_K$, then $\Delta w \in W_K$ automatically.
2. If the eigenvalues of $H(w)$, considered as an operator on W_K, are contained in $[\lambda_-, \lambda_+]$, the eigenvalues of $\tilde{H}(w)$ are contained in $[\lambda_-, \lambda_+]$ as well. Furthermore, if $|D_i| > 0$ for all $i = 1, 2, \ldots, K$, the function g is even strictly convex, and, thus, the eigenvalues of $H(w)$ on W_K are strictly negative, i.e., $\lambda_+ < 0$. As a consequence, also the eigenvalues of $\tilde{H}(w)$ are strictly negative, ensuring invertibility of $\tilde{H}(w)$.

These two properties are useful when implementing Newton's method, see Section 3.4. Item 1 means that we may essentially forget about the space W_K, and work in $\mathbb{R}^K$ without second thoughts. The second item makes sure that our modification does not lead to an ill-conditioning of the linear system determining (3.8) the Newton increment. Indeed, for any $\lambda > 0$, the matrix

$$H - \lambda \underline{1}\, \underline{1}^T$$

is negative semi-definite. If λ is not chosen carefully, it may deteriorate the conditioning of the matrix in question. We know that

$$\frac{\mathrm{tr}(H(w))}{K} \in [\lambda_-, \lambda_+],$$

as it corresponds to the mean of the eigenvalues. Furthermore, the unit vector $\frac{1}{\sqrt{K}}\underline{1}$ spans the kernel of $H(w)$ (at least on the set where g is strictly concave), and $\frac{1}{K}\underline{1}\,\underline{1}^T$ is the orthogonal projector onto this kernel. Combining these observations leads to item 2.

2.2. Generating centroidal Laguerre tessellations

For a tessellation $\{D_i\}$ of Y denote by $c_i \in Y$ a centroid of D_i, i.e., a minimizer of

$$\int_{D_i} d(c,x)^2\,dx \longrightarrow \min_{c\in Y}.$$

The periodic boundary conditions may give rise to special situations if the cells D_i "wrap around Y". As a simple example, for the single cell Y, *any* $c \in Y$ will be a centroid.

However, if the Laguerre cell D_i is sufficiently small, i.e., it may be displaced in Y to a configuration not touching the boundary $\partial Y \subseteq \mathbb{R}^d$, the classical representation as the center of mass

$$c_i = \frac{1}{|D_i|}\int_{D_i} x\,dx \tag{2.10}$$

(relative to a suitable local coordinate system) may be used. We shall rely upon the latter expression for practical purposes.

A Laguerre tessellation is called centroidal [80] if each seed is a centroid of its corresponding cell. Although, when looking at solidification of polycrystalline materials, there does not seem to be any requirement that the nucleation site corresponds to the grain centroid, it has been observed, see Bourne et al. [66], that centroidal Laguerre tessellations exhibit a better shape regularity than general Laguerre tessellations.

Suppose a set of (mutually different) seeds $(x_1, x_2, \ldots, x_K) \in Y^K$ and volume fractions $\phi \in \mathbb{R}^K$ with $\phi_i > 0$ and $\sum_{i=1}^K \phi_i = 1$ are given. By the reasoning of the previous section, we find a set of weights $w \in \mathbb{R}^K$ whose associated Laguerre tessellation with seeds $(x_1, x_2, \ldots, x_K)$ has cells with volume fractions ϕ. Denote by $(c_1, c_2, \ldots, c_K) \in Y^K$ the centroids of the respective Laguerre cells. A simple method for iteratively approaching a centroidal Laguerre tessellation with volume fractions ϕ as above is to use the centroids as new seeds. A pseudo-code of this classical Lloyd-type algorithm [54] is shown in Algorithm 1.

Algorithm 1 Lloyd-type centering (ϕ, maxit, tol)

```
Determine initial guess (x_1^0, x_2^0, ..., x_K^0) of seeds
k ← 0
residual ← +∞
while k < maxit and residual > tol do
    compute Laguerre tessellation with seeds (x_1^k, x_2^k, ..., x_K^k) for volume fractions φ
    compute centroids (c_1^k, c_2^k, ..., c_K^k) by (2.10)
    (x_1^{k+1}, x_2^{k+1}, ..., x_K^{k+1}) ← (c_1^k, c_2^k, ..., c_K^k)
    k ← k + 1
    update residual, cf Eq. (2.11)
end while
return ε, residual
```

Several remarks are in order:

1. The initial guess of seeds may be determined by pseudo- or quasi-random number generators.
2. For computing the weights of the Laguerre tessellation for given seeds and target volume-fractions, any of the algorithms described in Section 3 may be used.

3. For assessing convergence at the kth iteration, we use the residual

$$\text{residual} = \sqrt{\frac{\sum_{i=1}^{K} \phi_i \, d(x_i^k, c_i^k)^2}{(\max L_i)^2}}, \tag{2.11}$$

   i.e., the L^2-norm of the difference between seeds and centroids, weighted by volume-fraction and normalized by the maximum edge length of the unit cell. A typical value for tol is 10^{-1}.
4. The Lloyd-type algorithm presented can be shown to converge by using a monotonicity argument [54].

The problem of obtaining a centroidal Laguerre tessellation may also be written as an energy-minimization problem, see Liu et al. [56]. However, the resulting optimization problem admits a multitude of minimizers, i.e., we face a non-convex optimization problem, in general, and it is difficult to design powerful computational strategies. For instance, there are examples where the BFGS method does not converge, even with exact line search, cf. Dai [81].

As an alternative, we investigate Anderson acceleration [75], which is a technique for accelerating general fixed point iterations

$$x^{k+1} = F(x^k),$$

where, in our context, $x^k \in Y^K$ and the mapping

$$F : Y^K \to Y^K \tag{2.12}$$

is given by computing the K centroids of the Laguerre tessellation, see Algorithm 1.

More precisely, the Anderson-accelerated version of the fixed point iteration (2.12) with depth $m > 0$ stores the last m iterates

$$x^k, x^{k-1}, \ldots, x^{k-m+1} \quad \text{and their images w.r.t. } F, \quad F(x^k), F(x^{k-1}), \ldots, F(x^{k-m+1}),$$

computes $\alpha^k \in \mathbb{R}^m$ as the minimizer of the problem

$$\left\| \sum_{i=1}^{m} \alpha_i (F(x^{k+1-i}) - x^{k+1-i}) \right\|^2 \longrightarrow \min_{\sum_{i=1}^{m} \alpha_i = 1} \tag{2.13}$$

and proposes the next iterate as

$$x^{k+1} = \sum_{i=1}^{m} \alpha_i F(x^{k+1-i}). \tag{2.14}$$

Several remarks are in order:

1. Problem (2.13) is a quadratic problem with linear constraint. In particular, its KKT conditions are described by an $(m+1) \times (m+1)$ linear system. As m is typically small (not exceeding 10), this problem can be solved directly with negligible computational overhead.
2. Recall that we are using periodic boundary conditions for the cell $Y = [0, L_1) \times [0, L_2) \times \cdots \times [0, L_d)$. Thus, for any $y \in Y$ and $z \in Z$, there are countably many vectors $\xi \in \mathbb{R}^d$ with

$$y_i = z_i + \xi_i \mod L_i \quad (i = 1, 2, \ldots, d). \tag{2.15}$$

   Indeed, for any such ξ, and any d-tuple $(k_1, k_2, \ldots, k_d)$ of integers, we may form the vector

$$\tilde{\xi} = \xi + \sum_{i=1}^{d} k_i L_i e_i, \tag{2.16}$$

   which also satisfies (2.15) with ξ_i replaced by $\tilde{\xi}_i$. In turn, any two vectors ξ and $\tilde{\xi}$ satisfying Eq. (2.15) are related by (2.16) with a suitable tuple $(k_1, k_2, \ldots, k_d)$ of integers. Thus, we define $y - z$ as the vector ξ satisfying (2.15) with minimal length (which is well-defined if y and z are sufficiently close in the periodic distance). Using this definition K-times clarifies how the difference vectors $F(x^{k+1-i}) - x^{k+1-i}$ in (2.13) have to be understood.
3. The system matrix for the KKT-system of problem (2.13) may be updated at each iteration by computing m inner products.

4. For the first steps, i.e., where $k < m$, $\tilde{m} = k - 1$ is used for the current depth. We implement this by forcing the last few components of α^k to zero.
5. For linear equations, Anderson acceleration of depth m is "essentially equivalent" to GMRES(m) [82], see Toth–Kelley [77].
6. For general nonlinear equations, Anderson acceleration may be interpreted as a Quasi-Newton method, see Fang–Saad [76]. More precisely, it builds up an approximate Hessian from the last m secants. Under a suitable non-degeneracy condition it can be shown rigorously that Anderson acceleration improves the convergence rate of linearly converging fixed point iterations [78]. For degenerate problems, safeguarding strategies are necessary for improving convergence [83].
7. The Anderson-accelerated Lloyd-type centering generalizes the relaxed (or damped) Lloyd method for $m = 1$ with variable relaxation.

The resulting algorithm is summarized in Algorithm 2. For $m = 0$, we (formally) recover the original algorithm (Algorithm 1). For our numerical experiments, we set $m = 4$.

Algorithm 2 Anderson-accelerated Lloyd-type centering (ϕ, m, maxit, tol)

Determine initial guess $(x_1^0, x_2^0, \ldots, x_K^0)$ of seeds
$k \leftarrow 0$
residual $\leftarrow +\infty$
while $k <$ maxit **and** residual $>$ tol **do**
    compute Laguerre tessellation with seeds $(x_1^k, x_2^k, \ldots, x_K^k)$ for volume fractions ϕ
    compute centroids $(c_1^k, c_2^k, \ldots, c_K^k)$ by (2.10)
    update the linear system for solving (2.13)
    determine the vector α^k by solving (2.13)
    update $x^{k+1} \leftarrow \sum_{i=1}^{m} \alpha_i F(x^{k+1-i})$
    $k \leftarrow k + 1$
    update residual, cf Eq. (2.11)
end while
return ε, residual

3. Optimization methods

In this section, we describe pertinent optimization algorithms used for solving the concave optimization problem (2.4)

$$g(w) \longrightarrow \max_{w \in W_K}$$

for the unknown weight vector, and given seeds and volume fractions (which we assume to be fixed throughout this entire section). To bring it into a more convenient form, we shall equivalently rewrite the latter problem as a minimization problem via

$$f(w) \longrightarrow \min_{w \in W_K} \tag{3.1}$$

with $f(w) = -g(w)$, i.e., in terms of the convex function f. In particular, we shall use descent methods instead of ascent algorithms, for instance.

3.1. The gradient-descent method

Recall that the gradient of the function f (3.1) is given, cf. Eq. (2.7), component-wise by

$$\nabla_i f(w) = |D_i| - \phi_i \cdot L_1 L_2 \cdots L_d \tag{3.2}$$

Thus, a simple gradient-descent method

$$w^{n+1} = w^n - \alpha_n \nabla f(w^n), \tag{3.3}$$

involving a positive sequence of step sizes α_n, may be used for solving (3.1), as pioneered by Aurenhammer–Hoffmann–Aronov [62]. If the (local) Lipschitz constant of the gradient of f is known to be M, the iterative scheme (3.3) can be shown to converge for the choice $\alpha_n = \frac{1}{M}$, see Nesterov [74]. However, for convex f, the convergence rate is only logarithmic, in general. If, in addition, f is m-strongly convex in the vicinity of the minimizer, the convergence rate improves to be linear, and the choice $\alpha_n = \frac{2}{M+m}$ optimizes the convergence rate (estimate). Notice that the required iteration count is proportional to M/m.

Unfortunately, for the problem (3.1) both the Lipschitz constant and the strong convexity constant are not easily available. Of course, both may be obtained as the minimum and maximum eigenvalue of the Hessian (2.8). However, even computing the entries of the Hessian matrix (not even accounting for extracting the extremal eigenvalues) leads to significant computational overhead.

We are only interested in the qualitative performance of the gradient scheme (3.3) in order to comparing it asymptotically to more elaborate minimization schemes. Thus, taking into account the necessary dimension of the step-size (the weights have dimensions length^2, the gradient has dimension d), we simply use

$$\alpha_n = 0.1\,(L_1 L_2 \cdots L_d)^{\frac{2}{d}-1}. \tag{3.4}$$

Aurenhammer et al. [62] propose using a step-size of Polyak-type [84]. However, the latter requires a good estimate of the optimal objective-value (which is not available, in general) to be competitive. Thus, we stick to the simple choice (3.4).

3.2. The Malitsky–Mishchenko solver

Malitsky–Mishchenko [72] designed an adaptive strategy for choosing the step size in gradient methods which is globally convergent for general convex continuously differentiable functions (not necessarily with globally Lipschitz-continuous gradient) that automatically exploits strong convexity. The method is non-monotone, but significantly outperforms gradient-descent methods with fixed step-size, in general.

The Malitsky–Mishchenko method involves, in addition to the step-size α_n, another sequence $\{\theta_n\}$ which is (formally) initialized by $\theta_0 = +\infty$. Then, the scheme proceeds by updating a running estimate of the inverse Lipschitz constant of the gradient as the step-size

$$\alpha_n = \min\left(\sqrt{1+\theta_{n-1}}\alpha_{n-1}, \frac{\|w^n - w^{n-1}\|}{2\,\|\nabla f(w^n) - \nabla f(w^{n-1})\|}\right) \tag{3.5}$$

and gradient-descent steps (3.3), see Algorithm 1 in Malitsky–Mishchenko [72]. The sequence θ_n is updated in each step using the step-size $\theta_n = \frac{\alpha_n}{\alpha_{n-1}}$. The method requires an initial step size α_0, which we take as for the gradient-descent scheme (3.4).

3.3. The Barzilai–Borwein method

Originally devised as a special type of Quasi-Newton method, the Barzilai–Borwein [73] scheme may also be interpreted as an adaptive gradient-descent method using the step size

$$\alpha_n = \frac{\|w^n - w^{n-1}\|^2}{(\nabla f(w^n) - \nabla f(w^{n-1}))^T (w^n - w^{n-1})}. \tag{3.6}$$

For the Barzilai–Borwein method the initial step-size α_0 is also taken as for the gradient-descent scheme (3.4).

The Barzilai–Borwein step (3.6) leads to r-linear convergence when applied to strongly convex quadratic objective functions [85]. By a perturbation argument [86], the method can also be shown to be locally q-linearly convergent when applied to general strongly convex and twice differentiable objective functions. However, there are counter-examples to global convergence for the latter class, leading to the development of non-monotone backtracking techniques [87] and step-size limiting approaches [88], for instance. When applying the Barzilai–Borwein method to computational micromechanics [89], these globalization techniques were not necessary for the computational examples considered. On the contrary, in most cases they decreased the performance profile. For computing Laguerre tessellations, we made a similar observation, i.e., we did not encounter a single case where the Barzilai–Borwein method diverged, see Section 4. Thus, we shall rely upon the Barzilai–Borwein prescription (3.6) *without* globalization, although we are not aware of a mathematical justification for this choice.

3.4. Newton's method

For finding a critical point of a twice continuously differentiable function f, Newton's method iteratively updates

$$w^{n+1} = w^n + \alpha_n \xi^n, \tag{3.7}$$

where the Newton increment ξ^n solves the equation

$$\nabla^2 f(w^n)\, \xi^n = -\nabla f(w^n). \tag{3.8}$$

$\alpha_n \in (0, 1]$ is a damping factor (similar to the step-size of the gradient method). If w^* is a critical point of f with non-degenerate Hessian $\nabla^2 f(w^*)$ and $\nabla^2 f$ satisfies a Lipschitz condition close to w^*, Newton's method with $\alpha_n \equiv 1$ can be shown to be quadratically convergent in the vicinity of w^* [90]. However, global convergence is not achievable that way. Among the various globalization techniques for Newton methods [79], the damped Newton method (3.7) is particular popular. Far away from a critical point, $\alpha_n < 1$ may be necessary to ensure global convergence. However, in the vicinity of a critical point, no damping is necessary ($\alpha_n \equiv 1$ for $n \geq n_0$). In this way, the local convergence speed of Newton's method is preserved.

If f is (strictly) convex, ξ^n determined by (3.8) is always a descent direction for f at w^n. Thus, a proper backtracking strategy allows us to determine a suitable step size α_n. More precisely, given a backtracking factor $\rho \in (0, 1)$, starting from $\alpha_n^0 = 1$, the step size is iteratively shrunk, i.e., $\alpha_n^{k+1} = \rho \alpha_n^k$, until actual descent is observed, either in function value

$$f(w^n + \alpha_n^k \xi^n) < f(w^n)$$

or in its gradient

$$\|\nabla f(w^n + \alpha_n^k \xi^n)\| < \|\nabla f(w^n)\|.$$

For the problem at hand (3.1), we shall rely upon the latter criterion, as ∇f is cheaper to compute than f. Under suitable conditions, this backtracking strategy can be shown to be globally convergent under technical conditions, see Boyd–Vandenberghe [91]. We take $\rho = \frac{1}{2}$ as our backtracking factor.

In the context of Laguerre tessellations with prescribed volumes, a further issue needs to be addressed. For computing the Newton increment ξ^n, Eq. (3.8) needs to be solvable. In Section 2.1, we pointed out that constant vectors are in the kernel of $\nabla^2 f(w)$ for any $w \in \mathbb{R}^K$. However, this is no problem if Eq. (3.8) is interpreted as an equation on W_K, i.e., $\nabla f(w^n) \in W_K$ is given and $\xi^n \in W_K$ is sought. A simple remedy consists in adding the (suitably scaled) projector onto the kernel onto $\nabla^2 f(w^n)$, see Eq. (2.9).

Even restricted to W_K, the function f is not globally strictly convex. At such a point w, also the Hessian will be degenerate (even when restricted to W_K). More precisely, taking a look at the Hessian of g, we see that an empty Laguerre cell D_i implies that both the ith row and the ith column only contain zeros.

A workaround has been proposed by Lévy [64] by adding the constraint of non-empty cells to the backtracking strategy of Newton's scheme. This approach has been analyzed in Kitagawa et al. [65] and shown to be globally convergent when started at a feasible point (i.e., with non-empty Laguerre cells).

However, for grain-size distributions with large variation, we found this cautious backtracking strategy to be computationally inefficient. Although the method is quadratically convergent locally, the first few steps may require extensive backtracking. As a simple alternative, we add – in case of vanishing ith cell – a suitably scaled 1 on the ith diagonal entry of the Hessian. In this way, we ensure obtaining a descent direction and avoid the extensive backtracking.

3.5. BFGS

For large number K of grains, the computationally most demanding step of Newton's method (3.7) is solving the linear system (3.8) which determines the Newton increment. If a direct solver is used, on the order of K^3 floating point operations are required, which may be excessive for large K.

To reduce the effort, Quasi-Newton methods approximate the inverse of the Hessian $\nabla f(x)^{-1}$ directly, see Nocedal–Wright [79]. The most popular Quasi-Newton method is the Broyden–Fletcher–Goldfarb–Shanno

method [92–95], or BFGS, in short. In a nutshell, for every n, the next iterate is computed as for Newton's method (3.7)

$$w^{n+1} = w^n + \alpha_n \xi^n,$$

but the increment is determined by

$$\xi^n = -H_n \nabla f(w^n),$$

where $H_n \in \mathbb{R}^{K\times K}$ is a sequence of approximations to the inverse Hessian, which is, for BFGS, updated via

$$H_{n+1} = H_n + \frac{s_n^T y_n + y_n^T H_n y_n}{(s_n^T y_n)^2} s_n s_n^T - \frac{1}{s_n^T y_n}\left(H_n y_n s_n^T + s_n y_n^T H_n\right) \tag{3.9}$$

where $y_n = \nabla f(w^{n+1}) - \nabla f(w^n)$ and $s_n = w^{n+1} - w^n$. For strictly convex objective function f, the BFGS update (3.9) preserves symmetry and positive definiteness of the Hessian approximation, see Nocedal–Wright [79]. Thus, if the initial approximation H_0 is symmetric positive definite, all further H_n will remain symmetric and positive definite.

The effectiveness of BFGS crucially depends on the initial choice H_0. A common choice [79] is to use the Barzilai–Borwein step size (3.6), i.e.,

$$H_0 = \frac{\|w^0 - w^{-1}\|^2}{(\nabla f(w^0) - \nabla f(w^{-1}))^T (w^0 - w^{-1})}\,\mathrm{Id},$$

provided some initial gradient step has been taken, as for the Barzilai–Borwein method. When enriched by a backtracking strategy for determining the step size α_n in (3.7), BFGS can be shown to converge for strongly convex functions (and further technical conditions) with a superlinear rate [79].

Notice that we do not need to pay specific attention to the kernel of $\nabla^2 f$ consisting of component-wise equal weights. This is a result of H_n approximating the inverse of $\nabla^2 f$ on W_K. Indeed, H_0 is a multiple of the identity (with a proper scaling) and the BFGS update (3.9) only modifies W_K.

The BFGS method reduces the complexity required for the Newton step from $O(K^3)$ to $O(K^2)$ (due to the update of H_n and the matrix–vector product required for computing ξ^n). To further reduce the complexity to $O(K)$, limited-memory BFGS ($L-BFGS$), see Nocedal [71], may be used. In general, however, the superlinear convergence is lost. Also, $L-BFGS$ is often inferior to the Barzilai–Borwein method, as the latter scheme's non-monotonicity can be advantageous [89,96].

4. Numerical demonstrations

4.1. Setup and used hardware

The algorithms described in Sections 2.2 and 3 were implemented in Python 3.7 with Cython extensions. For computing Laguerre tessellations, we rely upon the Voro++ [97] code which we exposed to Python via Cython bindings using an OpenMP parallelization.

To reduce the stochastic influence, we use the Sobol low-discrepancy sequence [98] for drawing log-normal grain-size distributions [99]. The numerical experiments were conducted on desktop computer with 6 3.7 GHz CPUs and 32 GB RAM.

For solving the linear system for the Newton increment (3.8), we rely upon NumPy's linalg.solve routine, which wraps LAPACK [100]. The "higher-level" vector and matrix operations are realized by NumPy, for instance the BFGS update (3.9).

For all convex optimization methods of Section 3, we initialize the weights to zero. Using the estimate of Lyckegaard et al. [70], as suggested in Bourne et al. [66], did not decrease the run-time, on average.

To ensure comparability between different solvers, all experiments are initialized with identical seeds. We realize this by *using* the 3D Sobol sequence on the (scaled) unit cube. The first few elements of the 3D Sobol sequence are a little pathological, as they correspond to the corners of the cell and some other very regularly positioned spots, leading to *extremely regular* Laguerre tessellations. Therefore, we skip a few elements of the Sobol sequence. To ensure comparability, we fix the number of elements to be skipped as 1234 (for no specific reason).

Table 1
Iteration counts for different solvers for mono-sized GSD.

Number of grains K	250	500	1000	2000	4000	8000	16000
Gradient method	352	628	1017	1817	3275	5411	9758
Malitsky–Mishchenko	16	28	37	40	62	87	124
Barzilai–Borwein	18	20	31	32	39	59	66
BFGS	11	15	18	19	22	32	38
Newton	2	3	3	3	3	3	3

At this point, we wish to draw attention to the convergence criteria and tolerances used. For the convex optimization solvers, we test convergence by

$$\frac{\|\nabla f(w)\|}{\min_{i=1}^{K} \phi_i} \leq \text{tol} \tag{4.1}$$

with $\text{tol} = 10^{-2}$. This choice ensures that the volume fraction of the smallest grain is accurate to two significant digits (which we assume sufficient for materials-science purposes). For centering, we use Eq. (2.11), i.e.,

$$\sqrt{\frac{\sum_{i=1}^{K} \phi_i \, d(x_i^k, c_i^k)^2}{(\max L_i)^2}} \leq \text{tol}$$

with tol is 10^{-1}. As Bourne et al. [66], we noticed no visible improvement of the tessellation quality for lower values of tol. In contrast, the increase in computation times became quite visible.

We abbreviate grain-size distribution as GSD throughout Section 4. For all the experiments in this section, we use cubical cells, i.e., $L_1 = L_2 = L_3$. The prescribed algorithms work for anisotropic unit cells ("bricks") as well.

4.2. Computing the weights of a Laguerre tessellation with prescribed volume fractions

The overall goal of this section is to evaluate the solvers introduced in Section 3. For that purpose, we evaluate their iteration counts and run-times for problems of different complexity and grain count.

As our first example, we seek Laguerre tessellations with a mono-sized GSD, i.e., for K grains, each grain should have a volume fraction $\frac{1}{K}$.

As our smallest number of grains, we choose $K = 250$. For smaller grain count, the run-times become even more negligible and a sensible evaluation is difficult. Starting from $K = 250$, we subsequently double the grain count up to 16000 cells. Larger numbers are possible (and computationally feasible), but probably limited in terms of applicability to computational homogenization [16,17].

The iteration counts of the investigated solvers are listed in Table 1. The gradient method (3.3) serves as our basic reference. Even for $K = 250$, more than 300 iterations are required. For higher K, the iteration counts are proportional to K, i.e., they roughly double when doubling the number of grains.

For the Malitsky–Mishchenko method (3.5), the iteration counts are significantly lower than for the gradient method, roughly by one or two orders of magnitude. Still, we see a dependence of the iteration count on the grain number. More precisely, the iteration count depends approximately linearly on K, but with a non-trivial offset at $K = 0$. The Barzilai–Borwein method leads to a consistently lower iteration count than the Malitsky–Mishchenko method, except for $K = 250$, but no more than a factor of 2. BFGS (3.9) outperforms the Barzilai–Borwein scheme consistently in terms of iteration counts, also by no more than a factor of two. Newton's method, see section 3.4, operates on a completely different level in terms of iteration counts, requiring $2 - 3$ iterations for all grain counts considered. In particular, we see quadratic convergence in action.

When evaluating computational methods, apart from the iteration counts, also the run-times are of particular importance, see Table 2. Even at first glance we see that the gradient method is not competitive in terms of run-time, which is certainly not surprising. The other four solvers perform quite well in terms of run-time for the majority of problems, requiring only a few seconds.

The Malitsky–Mishchenko and the Barzilai–Borwein method are both gradient methods of similar type, i.e., the computational expense of their individual steps are comparative (see also Table 3). For $K \leq 2000$, both of their

Table 2
Run-time in s for different solvers for mono-sized GSD.

Number of grains K	250	500	1000	2000	4000	8000	16000
Gradient method	1.59	3.98	9.63	30.54	100.73	325.34	1155.00
Malitsky–Mishchenko	0.18	0.28	0.47	0.77	2.02	5.35	14.69
Barzilai–Borwein	0.19	0.27	0.54	0.65	1.30	3.69	8.21
BFGS	0.16	0.27	0.50	1.29	5.31	26.93	103.64
Newton	0.14	0.16	0.20	0.45	1.52	5.41	32.67

Table 3
Run-time per iteration in $10^{-2}s$ for different solvers and a mono-sized GSD.

Number of grains K	250	500	1000	2000	4000	8000	16000
Gradient method	0.45	0.63	0.95	1.68	3.08	6.01	11.84
Malitsky–Mishchenko	1.12	0.99	1.27	1.93	3.27	6.15	11.84
Barzilai–Borwein	1.06	1.35	1.75	2.02	3.34	6.26	12.45
BFGS	1.48	1.79	2.78	6.79	24.15	84.14	272.73
Newton	6.92	5.17	6.80	14.86	50.58	180.18	1088.62

run-times are similar (and similarly negligible). Only starting at $K = 4000$, differences are visible. For $K = 16000$, the Barzilai–Borwein method is about twice faster than Malitsky–Mishchenko.

Jumping ahead to Newton's scheme for the time being, we see that for small K (up to $K = 2000$), Newton's method is the fastest method investigated. For larger number of grains, Newton's scheme falls back behind the Barzilai–Borwein method in terms of run-time.

BFGS appears to be in between Barzilai–Borwein and Newton. For small grain count, it is slower than Newton's method, and for large K, it is slower than the Barzilai–Borwein scheme.

More insight into these observations may be gained by investigating the run-time per iteration of the different solvers under consideration, see Table 3. Remarkably, the three gradient-type methods require roughly the same computational effort per iteration. The expense is slightly higher for the Barzilai–Borwein method than for the other two gradient-type methods because the Python overhead plays a more distinct role, essentially due to the lower number of iterations required for Barzilai–Borwein method. Recall that we exposed the C++ - library Voro++ [97] to Python via Cython, and the major computational workload of the gradient-type methods amounts to computing Laguerre tessellations via Voro++. This library computes the Laguerre cells individually (and sequentially), using cell-linked lists [101]. As our initial seeds are more or less equally distributed, and – for mono-sized GSD – we look for cells of equal volume, the effort for computing a single Laguerre cell is effectively independent of the cell count. In turn, the total computational expense for computing the entire Laguerre tessellation is proportional to K. We see this trend quite clearly in Table 3 for the gradient-type solvers, highlighting the quality of the Voro++ - implementation. Notice the non-perfect scaling of the run-times, which is a result of the Python overhead and becomes less pronounced for large K.

A close look at the run-time per iteration for BFGS reveals a quadratic dependence on K. In contrast, the effort required for a Newton step grows cubically in K. These observations also clarify why the overall run-times of BFGS lag behind those of Newton's method for large K. BFGS has a lower effort per iteration than Newton's method, but the overall iteration count is also significantly larger than for Newton's method, tipping the overall scale towards Newton.

To finish this first example, we closer examined the convergence behavior of the solvers under consideration. For that purpose, for $K = 1000$ grains, we solve Eq. (2.2) up to a precision of $\text{tol} = 10^{-10}$. This level of accuracy is certainly beyond what is necessary for applications, but gives interesting insights into the internal mechanisms of the solvers. The residual (4.1), is plotted vs. the iteration count in Fig. 2. We did not include Newton's method due to its quadratic convergence. It appears that all shown solvers converge linearly. The gradient method converges slowly, but in a steady and monotonic fashion. Both the Malitsky–Mishchenko (MM) and the Barzilai–Borwein (BB) method do not exhibit monotone convergence behavior but lead to an oscillation of the residual by one or two orders of magnitude. Overall, both schemes converge r-linearly. This complies with theoretical estimates, see Section 3. Overall, the convergence speed of BB is higher than for MM.

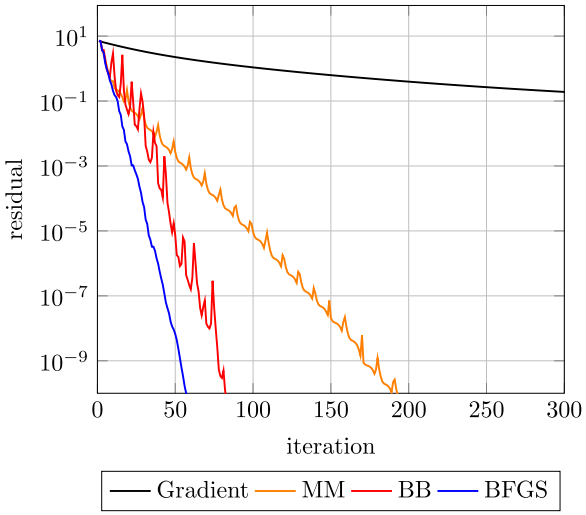


Fig. 2. Convergence behavior of different inner solvers for the mono-sized GSD and $K = 250$ grains.

Table 4
Iteration counts for different solvers and the two-sized GSD.

Number of grains K per phase	125	250	500	1000	2000	4000	8000
Malitsky–Mishchenko	34	41	86	83	110	203	295
Barzilai–Borwein	25	32	40	48	63	71	114
BFGS	19	20	29	28	32	41	59
Newton	3	5	6	4	4	5	7

BFGS also converges linearly, but in a monotonic fashion. Put differently, we have q-linear convergence. BFGS exhibits a higher convergence rate than the other solvers shown. However, no superlinear convergence is observed. This is consistent to theory, as to enable superlinear convergence, the approximation to the inverse Hessian needs to be close, as well, see section 8.4 in Nocedal–Wright [79]. As the Hessian is a 1000×1000-matrix, the less than 100 iterations (and in turn, the corresponding matrix updates (3.9)) are simply insufficient to match the true inverse Hessian closely.

As our next computational setup we consider a grain-size distribution with exactly two (distinct) volume fractions, each of which is equally probable. This may be viewed as a simplified bimodal grain-size distribution [102]. Following Bourne et al. [66], we use a factor of 5 between the two respective volume fractions. Thus, we consider K grains with volume fraction $\frac{1}{3K}$ and K grains with volume fraction $\frac{5}{3K}$, each. To keep consistency to the mono-sized GSD, the total number of grains $2K$ is varied between 250 and 16000, incrementally by factors of 2. Due to its lack of competitiveness for the mono-sized GSD, we did not consider the gradient-descent scheme any further.

The iteration counts for the four investigated solution methods are listed in Table 4. Both for Malitsky–Mishchenko and for Barzilai–Borwein, the iteration count increases monotonically with K. A similar trend may be observed for Newton's method and BFGS. However, there are exceptions between $K = 250$ and $K = 1000$. It is also evident that the iterations counts for the two-sized GSD are consistently higher than for the mono-sized GSD, see Table 1.

Comparing the iteration counts of Malitsky–Mishchenko and Barzilai–Borwein, we see that for increasing K, the discrepancy between the two solvers, in terms of iteration count, is increasing. For $K = 8000$, Malitsky–Mishchenko requires about three times the iteration count of Barzilai–Borwein.

As for the mono-sized GSD, BFGS requires less iterations than Barzilai–Borwein, roughly by a factor of two for the larger K.

For Newton's method, no quadratic convergence behavior can be observed for the two-sized GSD. Indeed, up to 7 Newton iterations are necessary to trigger the convergence criterion. Intuitively, this appears clear. Newton's method exhibits its strength close to the minimum. However, the seeds are evenly distributed, and the weights are

Table 5
Run-time in s for different solvers and the two-sized GSD.

Number of grains K per phase	125	250	500	1000	2000	4000	8000
Malitsky–Mishchenko	0.25	0.37	0.97	1.52	3.51	12.73	36.55
Barzilai–Borwein	0.24	0.32	0.53	0.93	2.08	4.44	14.51
BFGS	0.22	0.29	0.90	1.96	7.86	34.08	171.39
Newton	0.14	0.19	0.27	0.52	1.85	8.71	92.74

equal initially. Thus, initially, the computed Laguerre cells will have roughly equal volume. In particular, they do not appear to serve as a promising starting value for Newton's method. Therefore, a few preliminary descent steps are necessary to bring the iterates into the domain of quadratic convergence.

The required run-times are listed in Table 5. In general, the run-times exceed those of the mono-sized problem, see Table 2. The Malitsky–Mishchenko method consistently lags behind the Barzilai–Borwein solver in terms of run-time. Newton's method is very strong except for the highest grain size considered. More precisely, up to $K = 2000$, Newton's method outperforms all other solvers considered. For $K = 4000$ and $K = 8000$, the Barzilai–Borwein method is fastest. Again for the two-sized scenario, BFGS does not perform admirably.

Notice that up to $K = 2000$, the run-times of Barzilai–Borwein and Newton's method are minimal (up to about $2s$). Also, for the highest grain count considered, 16000 grains, the Barzilai–Borwein solver needs less than $15s$. Compared to the mono-sized problem, see Table 2, this amounts to an overall increase by a factor less than two.

For further analysis, we discard BFGS. To model more realistic grain-size distributions, we use a log-normal distribution for the equivalent radius of the grains. To be more precise, for a grain of volume V, we associate the radius r of a sphere with precisely this volume to it

$$V = \frac{4\pi}{3} r^3, \quad \text{i.e.,} \quad r = \sqrt[3]{\frac{3V}{4\pi}}.$$

It is empirically known [103] that the distribution of this equivalent radius follows a log-normal distribution with probability density function [99]

$$\rho(r) = \frac{1}{r} \cdot \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(\ln(r) - \mu)^2}{2\sigma^2}\right), \tag{4.2}$$

depending on the two real parameters μ and σ. The mean and standard deviation compute as [99]

$$\text{mean} = e^{\mu + \frac{1}{2}\sigma^2} \quad \text{and} \quad \text{stdev} = e^{\mu + \frac{1}{2}\sigma^2}\sqrt{e^{\sigma^2} - 1}.$$

As the mean equivalent radius plays no role in the geometry generation, we set it equal to unity. In turn, we obtain the relation $\mu = -\frac{1}{2}\sigma^2$, reducing the original two parameters to the single parameter σ. In turn, the expression for the standard deviation simplifies to

$$\text{stdev} = \sqrt{e^{\sigma^2} - 1}. \tag{4.3}$$

Thus, for given standard deviation stdev, we may determine the parameter σ characterizing the log-normal distribution (4.2) with mean 1 explicitly

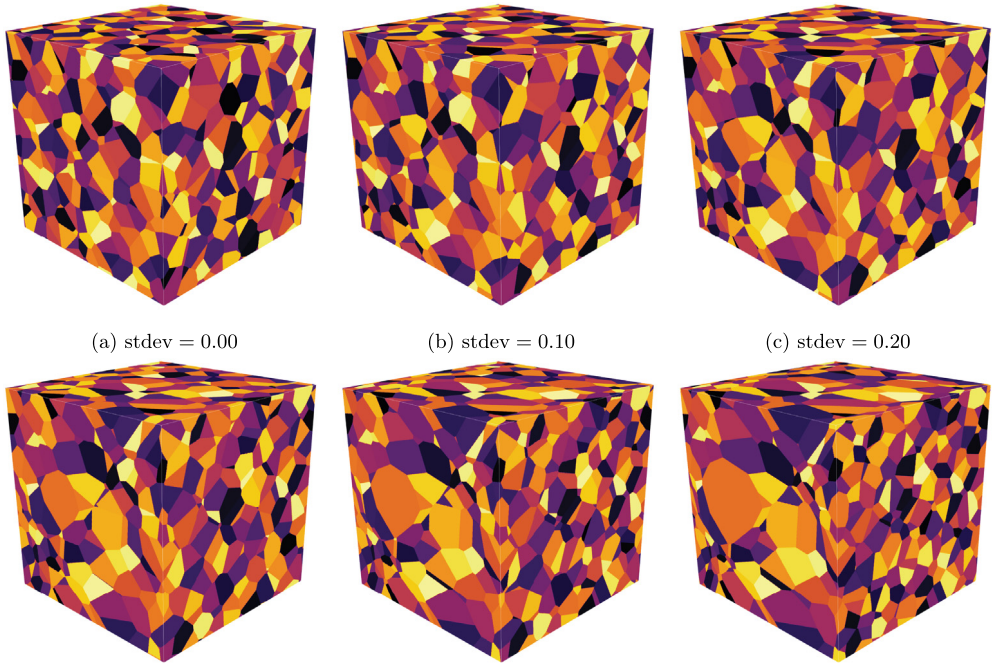
$$\sigma = \sqrt{\log(1 + \text{stdev}^2)}.$$

Thus, we may parameterize grain-size distributions by the standard deviation. For vanishing standard deviation, we recover the mono-sized distribution. Spettl et al. [103] report stdev = 0.35 as typical for a polycrystalline material that underwent grain-boundary migration by capillary effects.

To sum up, we prescribe the unit cell size (L_1, L_2, L_3), the number K of grains and the standard deviation stdev of the log-normal distribution. Then, we sample radii $r_1, r_2, \ldots, r_K$ according to the log-normal distribution by use of the Sobol sequence [98]. Then, we compute the appropriate volume fractions by

$$\phi_i = \frac{r_i^3}{\sum_{j=1}^{K} r_j^3},$$

which serves as input for our optimization routines.



(a) stdev = 0.00 (b) stdev = 0.10 (c) stdev = 0.20

(d) stdev = 0.30 (e) stdev = 0.40 (f) stdev = 0.50

Fig. 3. Influence of increasing standard deviation for polycrystalline microstructures with log-normally distributed equivalent radii and 1000 grains. The seed locations are fixed and the plots show different draws from the log-normal distribution.

Table 6
Iteration counts / run-times in s for different solvers and varying standard deviation for $K = 1000$ cells.

	MM	BB	Newton
0.00	37 / 0.35	31 / 0.30	3 / 0.11
0.05	94 / 0.99	60 / 0.60	3 / 0.12
0.10	134 / 1.41	69 / 0.75	3 / 0.13
0.15	162 / 1.83	99 / 1.17	4 / 0.14
0.20	215 / 2.61	115 / 1.39	5 / 0.32
0.25	272 / 3.48	137 / 1.74	9 / 0.58
0.30	343 / 4.62	176 / 2.37	5 / 0.19
0.35	460 / 7.03	168 / 2.42	5 / 0.19
0.40	628 / 9.74	250 / 3.95	11 / 0.75
0.45	934 / 15.07	359 / 5.96	25 / 2.93
0.50	1742 / 29.77	499 / 8.85	48 / 6.05

For the experiment at hand, we fix the grain count to $K = 1000$ and vary the standard deviation from 0 to 0.5 in 0.05-steps. Fig. 3 contains images for microstructures generated for varying standard deviation. The seeds are identical for all structures. We see that for increasing standard deviation some grains grow, whereas others shrink.

Please note that the colors appearing in the figures of this article showing polycrystalline microstructures are chosen randomly. Their sole purpose is to distinguish the individual cells. In particular, they are devoid of any physical meaning.

The iteration counts and run-times for increasing standard deviation and the Malitsky-Mishchenko and the Barzilai–Borwein scheme as well as Newton's method are listed in Table 6.

Apparently, as a general rule of thumb, increasing standard deviation also leads to an increase in iteration count *and* run-time for all solvers considered. This is not only caused by the increased difficulty, but also by the convergence criterion (4.1) in use. Indeed, the smallest among the volume fractions enters the convergence criterion. As the smallest volume fraction decreases for increasing standard deviation, the convergence criterion becomes

Table 7
Centering iteration counts for different optimization solvers, combined with the Lloyd-type algorithm (Algorithm 1) and its Anderson-accelerated version (Algorithm 2), and the mono-sized GSD.

Number of grains K	250	500	1000	2000	4000	8000	16000
BB + Algorithm 1	23	19	16	12	10	7	6
Newton + Algorithm 1	23	19	16	12	10	7	6
BB + Algorithm 2	13	11	10	8	7	6	5
Newton + Algorithm 2	13	11	10	8	7	6	5

stricter, as well. At this point, we may question the convergence criterion we use. However, this convergence criterion is standard, see, for instance, Lévy [64].

Taking a closer look at the iteration counts in Table 6, we see that the Malitsky–Mishchenko method consistently requires more iterations than the Barzilai–Borwein method. For stdev $= 0.5$, it even requires more than thrice the iteration count of Barzilai–Borwein. The required iterations for the Barzilai–Borwein method increases superlinearly in the standard deviation. It appears that the iteration count roughly doubles every 0.1-step in standard deviation.

For increasing standard deviation, Newton's method is still superior, but with a smaller margin, because it can profit less from its local quadratic convergence behavior. For low standard deviation (below 0.15), three iterations suffice. For intermediate standard deviation (less than 0.4), only a few Newton steps need to be taken before entering the domain of quadratic convergence. For larger standard deviation, the iteration counts increase significantly, rising up to 48 for stdev $= 0.5$.

Recall that we used $K = 1000$ grains. For stdev $= 0$, we recover the mono-sized experiment. The run-times for this case lie well below one second. For Malitsky–Mishchenko, compared to the mono-sized case, for stdev $= 0.5$ the run-time is increased by a factor of 100. For the other two solvers, the increase is less severe, but still noticeable. Overall, for standard deviations up to 0.4, Newton's method is extremely fast, with run-times below one second. Only for higher standard deviation, a significant increase in run-time is observed. The run-times for the Barzilai–Borwein method increase more steadily with increasing standard deviation, and are only slightly above those of Newton's method.

Recall, from the previous experiments, that the optimal solver strongly depends on the number of grains present in the tessellation. For $K = 1000$ grains, Newton's method always proved the fastest. For the last experiment, we saw that the standard deviation of the grain-size distribution also plays a significant role when evaluating the performance of the solvers in question.

4.3. Computing centroidal Laguerre tessellations for prescribed volume fractions

In the previous Section 4.2, we computed periodic Laguerre tessellations for prescribed volume fractions and *fixed* seeds. In this section, we wish to determine centroidal Laguerre tessellations for prescribed volume fractions, and add the centering algorithms of Section 2.2 on top of the optimization algorithms of Section 3. We have seen that either Newton's method or the Barzilai Borwein scheme proved computationally most efficient for the problems in Section 4.2. Thus, we restrict to these two solvers for the subsequent investigations. For the entire section, we use the convergence criterion (2.11) with tolerance tol $= 10^{-1}$. Recall that we check convergence for the convex optimization solvers by condition (4.1) with tol $= 10^{-2}$.

Consistent to Section 4.2, we first investigate a mono-sized GSD with increasing number of grains.

The number of centering iteration counts are listed in Table 7. We see that the number of centering iterations depends only on the "outer" algorithm used, and is independent on the "inner" algorithm. This fact appears clear at first, but becomes more interesting as we chose the tolerance for inner residual (4.1) comparatively large. Indeed, for inexact Newton methods [104], the choice of the inner tolerance has a strong influence on the overall performance of the scheme, see Dembo–Eisenstat–Steihaug [104]. However, this seems not to be the case for the problem at hand.

Comparing the Lloyd-type algorithm (Algorithm 1) and its Anderson-accelerated version Algorithm 2, we see Anderson acceleration decreases the iteration count in any case. However, for high grain count, the improvement is not large.

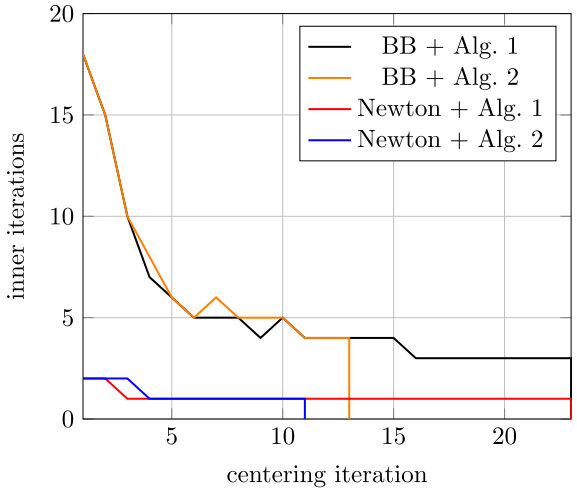


Fig. 4. Inner iterations vs centering iterations for the outer solvers for the mono-sized GSD and $K = 250$ grains..

Table 8
Run-time in s for different optimization solvers, combined with the Lloyd-type algorithm (Algorithm 1) and its Anderson-accelerated version (Algorithm 2), and the mono-sized GSD.

Number of grains K	250	500	1000	2000	4000	8000	16000
BB + Algorithm 1	0.59	0.81	1.33	2.25	4.49	9.61	19.27
Newton + Algorithm 1	0.32	0.42	0.73	1.85	7.51	23.81	140.84
BB + Algorithm 2	0.49	0.72	1.24	1.98	3.90	9.06	18.03
Newton + Algorithm 2	0.28	0.33	0.55	1.39	6.48	23.66	131.66

Table 9
Iteration counts for different optimization solvers, combined with the Lloyd-type algorithm (Algorithm 1) and its Anderson-accelerated version (Algorithm 2), and the two-sized GSD.

Number of grains K per phase	125	250	500	1000	2000	4000	8000
BB + Algorithm 1	31	20	17	13	9	7	6
Newton + Algorithm 1	31	20	17	13	9	7	6
BB + Algorithm 2	18	13	12	10	7	6	5
Newton + Algorithm 2	18	13	12	10	7	6	5

The associated run-times are listed in Table 8. Up to $K = 2000$ grains, Newton's method is faster. For higher grain count, the Barzilai–Borwein method leads to a lower run-time. Comparing the centering algorithms, Anderson acceleration consistently lowers the run-times.

More insight might be gained from Fig. 4, where the number of inner iterations is shown vs. the current centering iteration for $K = 250$. For Lloyd's method, see Algorithm 1, the inner iteration count decreases monotonically for increasing centering iteration. This property is preserved by Anderson acceleration (except for a single centering iteration).

To sum up, for large grain counts, i.e., $K \geq 4000$, combining the Barzilai–Borwein method with Algorithm 2 performs best, requiring less than $20s$ for the largest problem considered.

As our next example, as in Section 4.2, we consider a two-sized grain-size distribution with a ratio of 5 between the volume fractions of the two "phases", each of which has K grains.

The centering iteration counts are listed in Table 9. Compared to the mono-sized GSD, the iteration counts are slightly higher, on average. Still, the observations for the mono-sized GSD also apply for this two-sized case. As already observed in Bourne et al. [66], the time per centering iteration decreases. Therefore, we investigate the corresponding run-times, contained in Table 10. Already for $K = 4000$, Barzilai–Borwein combined with Algorithm 2

Table 10
Run-time in s for different optimization solvers, combined with the Lloyd-type algorithm (Algorithm 1) and its Anderson-accelerated version (Algorithm 2), and the two-sized GSD.

Number of grains K per phase	125	250	500	1000	2000	4000	8000
BB + Algorithm 1	1.40	1.57	2.42	4.04	6.42	12.45	33.41
Newton + Algorithm 1	0.51	0.63	1.34	3.05	16.01	55.08	406.16
BB + Algorithm 2	1.08	1.20	2.05	3.44	5.84	11.84	31.01
Newton + Algorithm 2	0.35	0.49	1.39	3.00	14.74	51.67	389.14

Table 11
Centering iteration counts and run-times in s for different optimization solvers, combined with the Lloyd-type algorithm (Algorithm 1) and its Anderson-accelerated version (Algorithm 2), and varying standard deviation. $*$ means that the method did not converge in a reasonable amount of time.

	BB Algorithm 1	Newton Algorithm 1	BB Algorithm 2	Newton Algorithm 2	BB Algorithm 1	Newton Algorithm 1	BB Algorithm 2	Newton Algorithm 2
0.00	16	16	10	10	1.37	0.59	1.06	0.45
0.05	16	16	10	10	1.87	0.59	1.62	0.55
0.10	16	16	10	10	2.35	0.69	1.99	0.48
0.15	16	16	10	10	3.23	1.15	2.75	1.13
0.20	16	16	10	10	4.09	3.35	3.41	3.47
0.25	17	17	11	11	4.92	8.07	4.37	8.36
0.30	18	18	12	12	6.87	8.30	6.32	8.22
0.35	18	18	12	13	8.68	9.67	7.63	10.12
0.40	20	20	15	15	12.70	20.74	11.38	19.62
0.45	21	21	17	17	18.53	25.66	17.61	27.22
0.50	22	22	16	16	25.15	29.56	23.44	29.91

outperforms all other variants. For the highest grain count considered, the latter combination converges in about half a minute, which is about 33% higher than for the mono-sized GSD.

As our last example in this paragraph, we consider the problem of computing a centroidal periodic Laguerre tessellation with $K = 1000$ grains with log-normally distributed GSD, cf equation (4.2), and varying standard deviation.

The number of centering iterations, cf. Table 11, is almost independent of the inner solver (different only by 1, at most). We see that the number of centering iterations increases monotonically with increasing standard deviation (except for Anderson and the transition from 0.45 to 0.5). However, in contrast to the inner iterations, see Table 6, the increase is modest. For both centering algorithms, the run-times are comparable, and thus mainly determined by the inner solver.

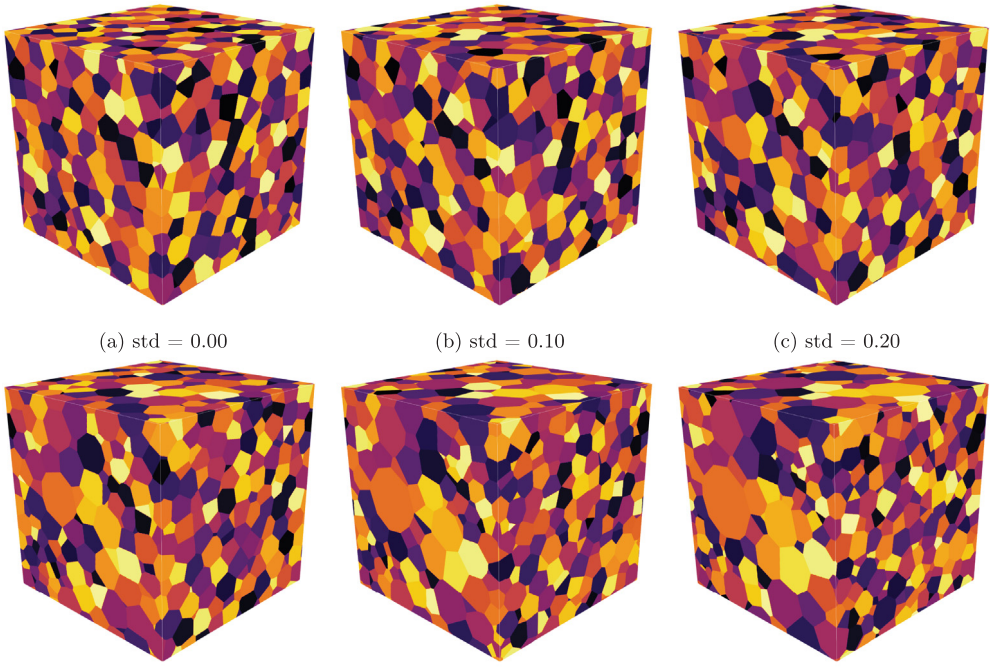
Taking a closer look at the run-times for Lloyd-type centering, we see that Newton's method serves as the faster inner solver up to a standard deviation of 0.2. However, Barzilai–Borwein is only slightly faster. Indeed, for stdev $= 0.5$, both inner solvers lead to an overall run-time less than half a minute.

The generated centroidal polycrystalline microstructures, for different values of the standard deviation, are shown in Fig. 5. For comparison, the non-centroidal versions may be of interest, cf Fig. 3. Recall that we used identical seeds for all cases, i.e., independent of standard deviation and whether centering is applied or not. Also, the coloring schemes are kept consistent. Thus, it is possible to match the various grains for all cases considered.

Apparently, the centroidal tessellations in Fig. 5 exhibit a much higher shape regularity than the non-centered tessellations of Fig. 3. Also, the sphericity of the large grains becomes more pronounced for increasing standard deviation.

4.4. Examples with a higher level of complexity

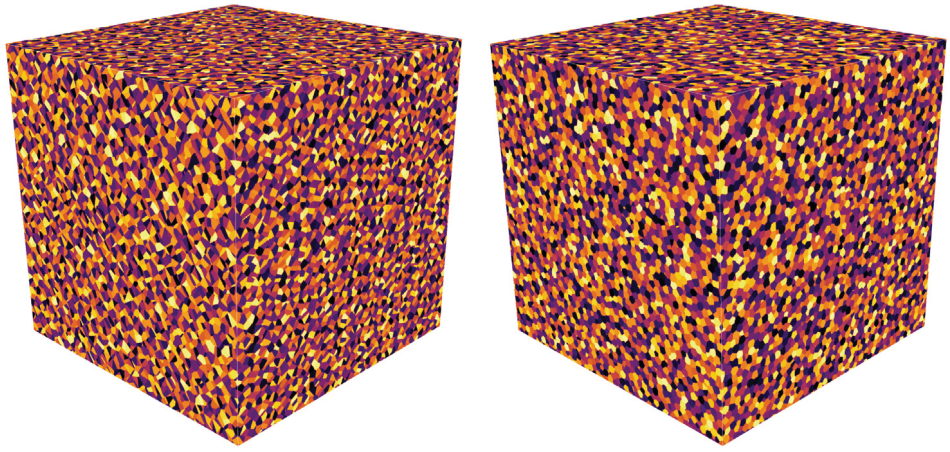
After thoroughly assessing the performance of the convex-problem solvers and the centering, we shall investigate examples with a higher degree of complexity.



(a) std = 0.00 (b) std = 0.10 (c) std = 0.20

(d) std = 0.30 (e) std = 0.40 (f) std = 0.50

Fig. 5. Influence of increasing standard deviation for *centered* polycrystalline microstructures with log-normally distributed equivalent radii and 1000 grains.

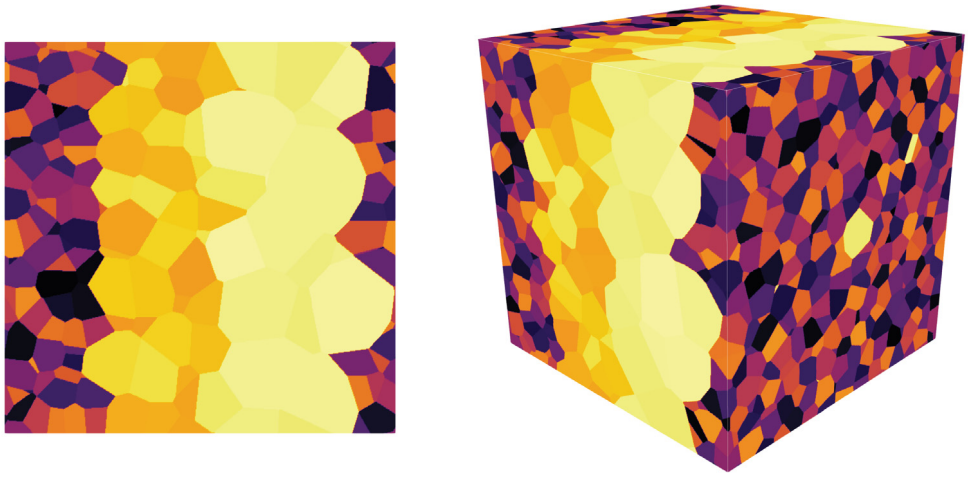


(a) Non-centroidal Laguerre tessellation (b) Centroidal Laguerre tessellation

Fig. 6. Generated microstructure with 100000 grains of identical volume.

We used the Barzilai–Borwein method to generate a microstructure with $100000 = 10^6$ grains of identical volume, up to an accuracy of 10^{-2}, as before. For quasirandom initial positions, the Barzilai–Borwein method needed $282.85s$, i.e., less than 5 minutes. The resulting microstructure is shown in Fig. 6(a).

Coupled to the Anderson-accelerated Lloyd algorithm, solved up to an accuracy of 10^{-4}, took $338.32s$, i.e., less than six minutes. The resulting microstructure is shown in Fig. 6(b) and may be compared to its non-centered cousin, cf Fig. 6(a).



(a) Slice through the center (b) Volumetric view

Fig. 7. Generated polycrystalline microstructure with spatial size-gradient.

Even without the multi-level strategies of Mérigot [63] and Lévy [64], we are able to generate microstructures with high complexity to be used in crystal-plasticity FEM [13–15] of FFT [16–18] frameworks in a matter of seconds to a few minutes.

The framework presented may also be used for generating microstructures with a morphological texture, i.e., a prescribed spatial gradient in the grain sizes. Such microstructures may be the result of high temperature gradients that may occur inside of polycrystalline metals, for example during welding [105].

We follow the ideas of Bourne et al. [66] and restrict the initial positions of the seeds to particular subsets. Of course, the resulting grains need not be entirely contained in the pre-selected areas, in particular if coupled to the centering methods. Still, interesting microstructure emerges, as we shall demonstrate by an explicit example.
We consider a cubical cell $[0, L] \times [0, L] \times [0, L]$. We shall divide the grains into three classes:

1. 1024 grains with initial positions restricted to $\left[0, \frac{L}{4}\right] \times [0, L] \times [0, L]$ accounting for $\frac{1}{3}$ of the volume fraction.
2. 256 grains with initial positions restricted to $\left[\frac{L}{4}, \frac{L}{2}\right] \times [0, L] \times [0, L]$ accounting for $\frac{1}{3}$ of the volume fraction.
3. 64 grains with initial positions restricted to $\left[\frac{L}{2}, L\right] \times [0, L] \times [0, L]$ accounting for $\frac{1}{3}$ of the volume fraction.

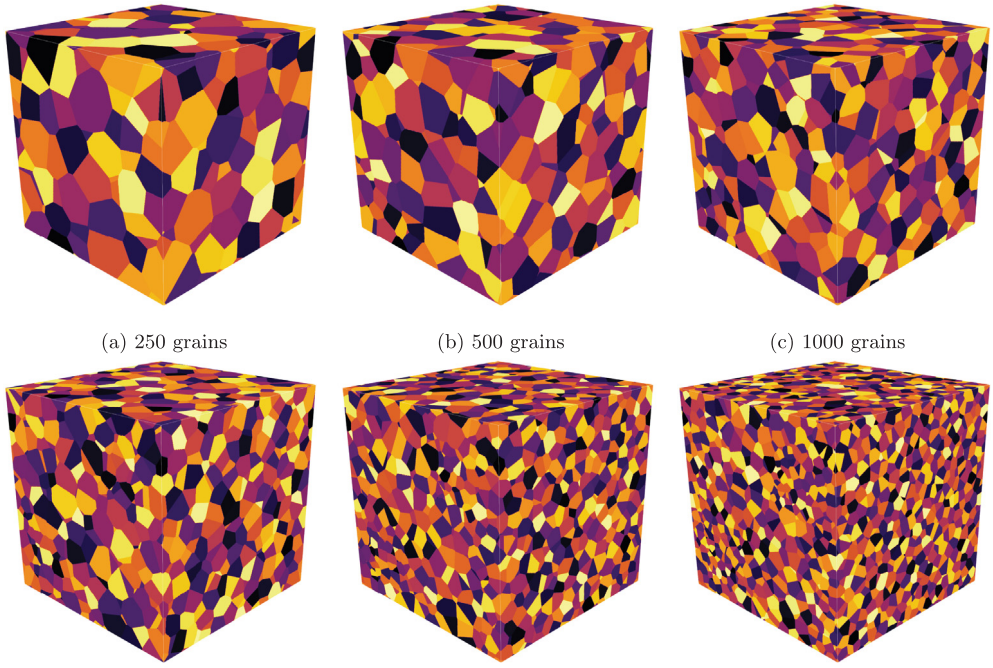
Thus, we have three types of grains. Each type has the same total volume. However, the grain volume of class 1 is four times smaller than for the second class, and sixteen times smaller than the third class. Also, the first class is restricted to the first quarter of the x-face, whereas the second class is restricted to the second quarter, and the third phase makes up the last half of the x-axes.

Generating the microstructure with Barzilai–Borwein and Anderson–Lloyd (with accuracy as above) took $2.0s$. The resulting microstructure is shown in Fig. 7(b). The first class is shown in red-to-black, the second class in yellow, whereas the third class is characterized by a grayish coloring.

As discussed in the introduction of this article, modern image-based characterization techniques enable measuring grain size distributions of polycrystalline materials empirically. Although analytically given grain size distributions have their merits, relying upon empirically given GSDs permits a close comparison to real-world specimens, in particular concerning their inherent imperfections and deviations from theoretical GSDs.

To demonstrate the capabilities of the proposed methodology, we use the empirical GSDs determined by Döbrich et al. [47] as our target GSDs for generating synthetic microstructures. More precisely, we use histogram bins for describing the desired GSD, see Fig. 9.

Different centroidal periodic Laguerre tessellations were generated with an increasing number of grains using Barzilai–Borwein in combination with the Anderson–Lloyd method. The resulting microstructures are shown in



(a) 250 grains (b) 500 grains (c) 1000 grains

(d) 2000 grains (e) 4000 grains (f) 8000 grains

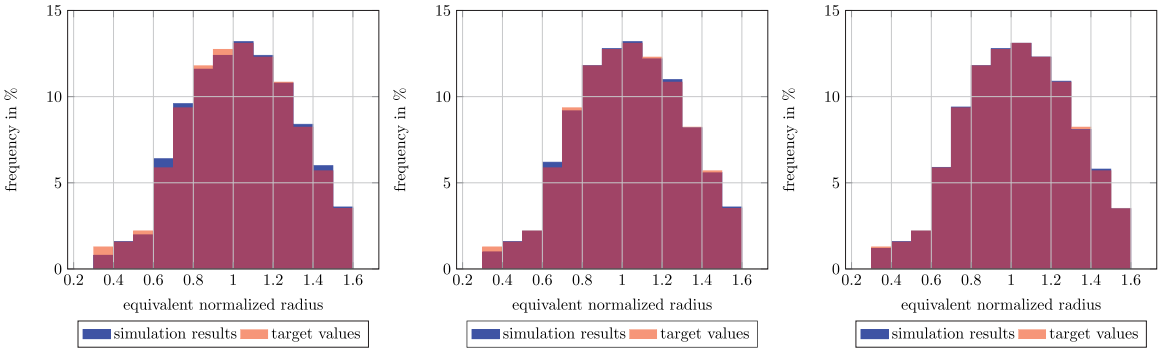
Fig. 8. Generated microstructures with empirical GSD of Döbrich et al. [47].

Table 12
Run-times for different numbers of grains and the empirical GSDs of Döbrich et al [47] using BB and Algorithm 2.

Number of grains	250	500	1000	2000	4000	8000
Run-time in s	0.53	1.62	3.04	10.57	33.15	133.51

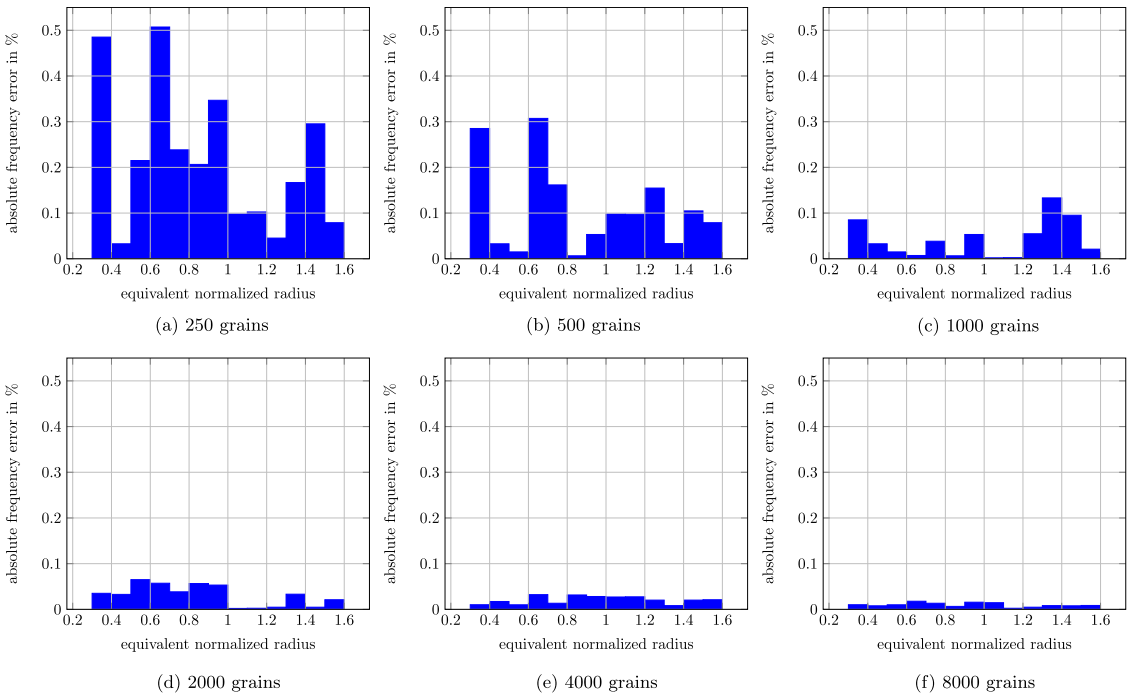
Fig. 8, and the run-times are collected in Table 12. We see that even the largest structure with 8000 grains required only slightly more than two minutes to compute. Of particular importance to us is the ability of the methodology to match any prescribed empirical GSD. For this reason, we take a look at Fig. 9, where both the target and the realized histograms are shown for 250, 500 and 1000 grains. For 250 and 500 grains, differences between the histograms are visible, although on a level probably within engineering accuracy. For 1000 grains, the target and the realized histograms are difficult to distinguish by eye. To gain more insight, we plotted the absolute errors (in %) between the desired and targeted frequencies per bin, cf Fig. 10. We see a maximum error of about 0.5% for 250 grains, 0.3% for 500 grains and below 0.15% for 1000 grains. For a higher grain count, the error decreases further. More precisely, we see that the error decreases as $1/K$, where K denotes the number of grains, i.e., the error reduces roughly by 50% when doubling the number of grains. This is no surprise, but a result of our quasirandom sampling of the grain sizes, which theoretically predicts a $1/K$ decrease of error upon sampling [106]. In contrast, a purely random sampling only reduces the expected error by a factor of $1/\sqrt{K}$. To gain some intuition into this difference, essentially to arrive at the same error for random sampling that we obtained with quasirandom sampling and 1000 grains would require 1000^2, i.e., one million, grains, on average.

Last but not least let us remark that the maximum error decreases linearly, but the bin-wise error might not. Indeed, taking a look at the second-smallest bin, the error only decreases for 2000 grains compared to 250 grains. However, the corresponding error is already small for 250 grains.



(a) 250 grains (b) 500 grains (c) 1000 grains

Fig. 9. Comparison of targeted and reached bin frequencies for the GSD of Döbrich et al. [47].



(a) 250 grains (b) 500 grains (c) 1000 grains

(d) 2000 grains (e) 4000 grains (f) 8000 grains

Fig. 10. Absolute frequency error of the microstructure realizations of Fig. 8, as a result of the quasi-random sampling, compared to the target values of Döbrich et al. [47].

5. Conclusion

This work has been devoted to generating geometry-based microstructure models for polycrystalline materials in terms of Laguerre tessellations. According to Ball–Carstensen [57] Laguerre tessellations describe polycrystalline microstructures with convex grains. Put differently, in order to describe a geometrically more complex polycrystalline microstructure, non-convex grains have to be considered.

In a recent work, Bourne et al. [66] have shown that, for any given set of seed points and prescribed volume fractions, a corresponding set of Laguerre weights realizing a Laguerre tessellation with the prescribed volume fractions may be determined by solving a convex optimization problem. As a consequence, using black-box optimization methods, Bourne et al. [66] were able to generate polycrystalline microstructures of industrial complexity in the order of minutes.

In this article, we retained the basic algorithmic framework of Bourne et al. [66] for computing the weights of a periodic Laguerre tessellation for prescribed seed locations and volume fractions, and exploited the benefits of modern non-monotone methods of gradient type, i.e., the Malitsky–Mishchenko method and the Barzilai–Borwein scheme, in the context of semi-discrete optimal transport.

In the numerical experiments we saw that, for small to moderate grain count and low standard deviation of the grain-size distribution, Newton's method outperforms all other methods considered. For larger number of grains or for larger spread of the grain sizes, the Barzilai–Borwein method proved superior in terms of run-time. As the run-time for small- to medium-sized problems is small anyway, the Barzilai–Borwein method may serve as a general-purpose method for computing the weights of Laguerre tessellations.

To increase the physical realism, we also contributed to computing centroidal Laguerre diagrams of prescribed volume fractions by suggesting to combine the classical Lloyd's algorithm to Anderson acceleration, a general-purpose acceleration scheme for fixed-point methods. For all problems considered, the Anderson-accelerated Lloyd's method proved superior to the classical Lloyd's method. Anderson acceleration might also prove more powerful if higher accuracy for centering is required, in particular when compared to BFGS-type centering [56,107]. However, in the context of polycrystalline microstructures, such a high level of accuracy appears not to be necessary.

All in all, combining the improved convex optimization solver, the centering and an OpenMP-parallel implementation permits generating polycrystalline microstructures of industrial complexity and relevance in a matter of seconds to minutes, at most, reducing the run-time and increasing the number of grains compared to the work of Bourne et al. [66].

In particular, for grain-size distribution with large standard deviation of the grain sizes, it appears imperative to accept iterates with vanishing Laguerre cells. Classically, backtracking methods are used to avoid empty Laguerre cells during the iterative process. The proposed gradient methods avoid such backtracking — in fact, their performance decreases when backtracking is used. Furthermore, we also proposed a simple regularization strategy for Newton's method in case of vanishing Laguerre cell volume which leads to a robust and competitive computational scheme.

Our solution schemes may be further improved by using multi-level strategies, as pioneered by Mérigot [63] and Lévy [64]. However, for applications to micromechanics, the number of grains to consider typically does not exceed a few thousand.

Notice that we describe the morphological texture of a polycrystalline aggregate in terms of the seed locations and the volume fractions, taking into account only one-point statistics [20]. We have shown, by example, that microstructures with a spatial gradient in the grain-size distribution may be generated. Also, experimentally determined grain-size distributions may be reproduced with high accuracy, provided the equivalent radii are drawn in a pseudo-random fashion. Still, it might be interesting to see whether the algorithms could be extended to more general morphological features, e.g., sphericity.

Last but not least it is clear that, in order to conduct computational experiments, the generated polycrystalline microstructures need to be furnished with crystal orientations per grain, leading to possible crystallographic texture of the material under consideration. However, the latter is typically realized by a post-processing procedure, see, e.g., Quey–Villani–Maurice [108], and is thus complementary to the approach presented.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

MS thanks the German Research Foundation (DFG) for support within the International Research Training Group "Integrated engineering of continuous–discontinuous long fiber reinforced polymer structures" (GRK 2078). TB gratefully acknowledges the partial financial support by the German Research Foundation (DFG) under project BO 1466/12-2.

We are grateful to the anonymous reviewers for carefully reading the submitted manuscript and for the numerous improvements which they suggested.

References

[1] J.E. Spowart, H.E. Mullens, B.T. Puchala, Collecting and analyzing microstructures in three dimensions: a fully automated approach, JOM 55 (10) (2003) 35–37.

[2] A.J. Kubis, G.J. Shiflet, R. Hull, D.N. Dunn, Focused ion-beam tomography, Metall. Mater. Trans. A 37 (7) (2004) 1935–1943.

[3] N. Chawla, V.V. Ganesh, B. Wunsch, Three-dimensional (3D) microstructure visualization and finite element modeling of the mechanical behavior of SiC particle reinforced aluminum composites, Scr. Mater. 51 (2) (2004) 161–165.

[4] T. Wiederkehr, B. Klusemann, D. Gies, H. Müller, B. Svendsen, An image morphing method for 3D reconstruction and FE-analysis of pore networks in thermal spray coatings, Comput. Math. Sci. 47 (4) (2010) 881–889.

[5] R.K. Bansal, A. Kubis, R. Hull, J.M. Fitz-Gerald, High-resolution three-dimensional reconstruction: a combined scanning electron microscope and focused ion-beam approach, J. Vac. Sci. Technol. B 24 (2) (2006) 554–561.

[6] M.A. Groeber, B.K. Haley, M.D. Uchic, D.M. Dimiduk, S. Ghosh, 3D reconstruction and characterization of polycrystalline microstructures using a FIB-SEM system, Mater. Charact. 57 (4) (2006) 259–273.

[7] S. Zaefferer, S.I. Wright, D. Raabe, Three-dimensional orientation microscopy in a focused ion beam-scanning electron microscope: a new dimension of microstructure characterization, Metall. Mater. Trans. A 39 (2) (2008) 374–389.

[8] S. Korte, M. Ritter, C. Jiao, P.A. Midgley, W.J. Clegg, Three-dimensional electron backscattered diffraction analysis of deformation in MgO micropillars, Acta Mater. 59 (19) (2011) 7241–7254.

[9] B.L. Adams, T. Olson, The mesostructure - property linkage in polycrystals, Prog. Mater. Sci. 43 (1) (1998) 1–87.

[10] B.C. Larson, W. Yang, G.E. Ice, J.D. Budai, J.Z. Tischler, Three-dimensional X-ray structural microscopy with submicrometre resolution, Nature 415 (6874) (2002) 887–890.

[11] H.F. Poulsen, Three-Dimensional X-Ray Diffraction Microscopy, Springer, Berlin, 2004.

[12] H. Abdolvand, M. Majkut, J. Oddershede, S. Schmidt, U. Lienert, B.J. Diak, P.J. Withers, M.R. Daymond, On the deformation twinning of Mg AZ31B: A three-dimensional synchroton X-ray diffraction experiment and crystal plasticity finite element model, Int. J. Plast. 70 (2015) 77–97.

[13] R. Becker, Analysis of texture evolution in channel die compression - I. Effects of grain interaction, Acta Metall. Mater. 39 (1991) 1211–1230.

[14] F. Barbe, L. Decker, D. Jeulin, G. Cailletaud, Intergranular and intragranular behavior of polycrystalline aggregates. Part 1: FE model, Int. J. Plast. 17 (2001) 513–536.

[15] F. Roters, P. Eisenlohr, L. Hantcherli, D.D. Tjahjanto, T.R. Bieler, D. Raabe, Overview of constitutive laws, kinematics, homogenization and multiscale methods in crystal plasticity finite-element modeling: theory, experiments, applications, Acta Mater. 58 (2010) 1152–1211.

[16] P. Shanthraj, P. Eisenlohr, M. Diehl, F. Roters, Numerically robust spectral methods for crystal plasticity simulations of heterogeneous materials, Int. J. Plast. 66 (2015) 31–45.

[17] D. Wicht, M. Schneider, T. Böhlke, An efficient solution scheme for small-strain crystal-elasto-viscoplasticity in a dual framework, Comput. Methods Appl. Mech. Engrg. 358 (2020) 112611.

[18] R.A. Lebensohn, A.D. Rollett, Spectral methods for full-field micromechanical modelling of polycrystalline materials, Comput. Math. Sci. 173 (2020) 109336.

[19] S. Bargmann, B. Klusemann, J. Markmann, J.E. Schnabel, K. Schneider, C. Soyarslan, J. Wilmers, Generation of 3D representative volume elements for heterogeneous materials: A review, Prog. Mater. Sci. 96 (2018) 322–384.

[20] S. Torquato, Random Heterogeneous Materials, Springer, New York, 2002.

[21] J. Ohser, F. Mücklich, Statistical Analysis of Microstructures in Materials Science, Wiley, Hoboken, 2000.

[22] K. Kawasaki, T. Nagai, K. Nakashima, Vertex models for two-dimensional grain growth, Phil. Mag. B 60 (3) (1989) 399–421.

[23] D. Weygand, Y. Bréchet, J. Lépinoux, W. Gust, Three-dimensional grain growth: a vertex dynamics simulation, Phil. Mag. B 79 (5) (1999) 703–716.

[24] L.A. Barrales-Mora, G. Gottstein, L. Shvindlerman, Three-dimensional grain growth: analytical approaches and computer simulations, Acta Mater. 56 (20) (2008) 5915–5929.

[25] K.G.F. Janssens, An introductory review of cellular automata modeling of moving grain boundaries in polycrystalline materials, Math. Comput. Simulation 80 (2010) 1361–1381.

[26] M. Kühbach, G. Gottstein, L.A. Barrales-Mora, A statistical ensemble cellular automaton microstructure model for primary recrystallization, Acta Mater. 107 (2016) 366–376.

[27] R.S. Saluja, R.G. Narayanan, S. Das, Cellular automata finite element (CAFE) model to predict the forming of friction stir welded blanks, Comput. Math. Sci. 58 (2012) 87–100.

[28] M.P. Anderson, G.S. Grest, D.J. Srolovitz, Computer simulation of normal grain growth in three dimensions, Phil. Mag. B 59 (3) (1989) 293–329.

[29] B. Radhakrishnan, T. Zacharia, Monte Carlo simulation of grain boundary pinning in the weld heat-affected zone, Metall. Mater. Trans. A 26 (8) (1995) 2123–2130.

[30] C. Krill Iii, L.-Q. Chen, Computer simulation of 3-D grain growth using a phase-field model, Acta Mater. 50 (12) (2002) 3059–3075.

[31] B. Nestler, A 3D parallel simulator for crystal growth and solidification in complex alloy systems, J. Cryst. Growth 275 (1) (2005) 273–278.

[32] J. Höetzer, V. Rehn, W. Rheinheimer, M.J. Hoffmann, B. Nestler, Phase-field study of pore-grain boundary interaction, J. Ceram. Soc. Japan 124 (4) (2016) 329–339.

[33] A. Alpers, A. Brieden, P. Gritzmann, A. Lyckegaard, H.F. Poulsen, Generalized balanced power diagrams for 3D representations of polycrystals, Phil. Mag. 95 (9) (2015) 1016–1028.

[34] O. Šedivý, T. Brereton, D. Westhoff, L. Polívka, V. Beneš, V. Schmidt, A. Jäger, 3D reconstruction of grains in polycrystalline materials using a tessellation model with curved grain boundaries, Phil. Mag. 96 (18) (2016) 1926–1949.
[35] K. Teferra, L. Graham-Brady, Tessellation growth models for polycrystalline microstructures, Comput. Mater. Sci. 102 (2015) 57–67.
[36] K. Teferra, D.J. Rowenhorst, Direct parameter estimation for generalised balanced power diagrams, Phil. Mag. Lett. 98 (2) (2018) 79–87.
[37] A. Brahme, M.H. Alvi, D. Saylor, J. Fridy, A.D. Rollett, 3D reconstruction of microstructure in a commercial purity aluminum, Scr. Mater. 55 (1) (2006) 75–80.
[38] M. Groeber, S. Ghosh, M.D. Uchic, D.M. Dimiduk, A framework for automated analysis and simulation of 3D polycrystalline microstructures. Part 2: Synthetic structure generation, Acta Mater. 56 (6) (2008) 1274–1287.
[39] M.A. Groeber, M.A. Jackson, DREAM. 3D: A digital representation environment for the analysis of microstructure in 3D, Integrating Mater. Manuf. Innov. 3 (2014) 56–72.
[40] S. Mandal, J. Lao, S. Donegan, A.D. Rollett, Generation of statistically representative synthetic three-dimensional microstructures, Scr. Mater. 146 (2018) 128–132.
[41] L. St-Pierre, E. Héripré, M. Dexet, J. Crépin, G. Bertolino, N. Bilger, 3D simulations of microstructure and comparison with experimental microstructure coming from O.I.M analysis, Int. J. Plast. 24 (9) (2008) 1516–1532.
[42] J.C. Tucker, L.H. Chan, G.S. Rohrer, M.A. Groeber, A.D. Rollett, Tail departure of log-normal grain size distributions in synthetic three-dimensional microstructures, Metall. Mater. Trans. A 43 (8) (2012) 2810–2822.
[43] F. Aurenhammer, Voronoi diagrams - a survey of a fundamental geometric data structure, ACM Comput. Surv. 23 (1991) 345–405.
[44] F. Fritzen, T. Böhlke, E. Schnack, Periodic three-dimensional mesh generation for crystalline aggregates based on Voronoi tessellations, Comput. Mech. 43 (5) (2009) 701–713.
[45] R. Quey, P.R. Dawson, F. Barbe, Large-scale 3D random polycrystals for the finite element method: generation, meshing and remeshing, Comput. Meth. Appl. Mech. Engnrg. 200 (17) (2011) 1729–1745.
[46] S. Bargmann, B. Svendsen, M. Ekh, An extended crystal plasticity model for latent hardening in polycrystals, Comput. Mech. 48 (6) (2011) 631–645.
[47] K.M. Döbrich, C. Rau, C.E. Krill III, Quantitative characterization of the three-dimensional microstructure of polycrystalline Al-Sn using X-ray microtomography, Metall. Mater. Trans. A 35 (7) (2004) 1953–1961.
[48] T. Luther, C. Könke, Polycrystal models for the analysis of intergranular crack growth in metallic materials, Eng. Fract. Mech. 76 (15) (2009) 2332–2343.
[49] M. Groeber, S. Ghosh, M.D. Uchic, D.M. Dimiduk, A framework for automated analysis and simulation of 3D polycrystalline microstructures.: Part 1: statistical characterization, Acta Mater. 56 (6) (2008) 1257–1273.
[50] T. Xu, M. Li, Topological and statistical properties of a constrained Voronoi tessellation, Phil. Mag. 89 (4) (2009) 349–374.
[51] Q. Du, V. Faber, M. Gunzburger, Centroidal Voronoi tessellations: applications and algorithms, SIAM Rev. 41 (4) (1999) 637–676, 1999.
[52] S.P. Lloyd, Least squares quantization in PCM, IEEE Trans. Inform. Theory 28 (2) (1982) 129–137.
[53] Q. Du, M. Emelianenko, L. Ju, Convergence of the Lloyd algorithm for computing centroidal Voronoi tessellations, SIAM J. Numer. Anal. 44 (2006) 102–119.
[54] D.P. Bourne, S.M. Roper, Centroidal power diagrams, Lloyd's algorithm and applications to optimal location problems, SIAM J. Numer. Anal. 53 (6) (2015) 2545–2569.
[55] Q. Du, M. Emelianenko, Acceleration schemes for computing centroidal Voronoi tessellations, Numer. Linear Algebra Appl. 13 (2–3) (2006) 173–192.
[56] Y. Liu, W. Wang, B. Lévy, F. Sun, D.-M. Yan, L. Lu, C. Yang, On centroidal Voronoi tessellation – Energy smoothness and fast computation, ACM Trans. Graph. 28 (4) (2016) article 244.
[57] J.M. Ball, C. Carstensen, Geometry of polycrystals and microstructure, arXiv e-prints, 2015, pp. 1–7, vol. 1507.05197.
[58] H. Imai, M. Iri, K. Murota, Voronoi diagram in the Laguerre geometry and its applications, SIAM J. Comput. 14 (1) (1985) 93–105.
[59] K. Hitti, P. Laure, T. Coupez, L. Silva, M. Bernacki, Precise generation of complex statistical representative volume elemens (RVEs) in a finite element context, Comput. Math. Sci. 61 (2012) 224–238.
[60] H. Altendorf, F. Latourte, D. Jeulin, M. Faessel, L. Saintoyant, 3D reconstruction of a multiscale microstructure by anisotropic tessellation models, Image Anal. Stereol. 33 (2) (2014) 121–130.
[61] C. Lautensack, S. Zuyev, Random Laguerre tessellations, Adv. Appl. Probab. 40 (2008) 630–650.
[62] F. Aurenhammer, F. Hoffmann, B. Aronov, Minkowski-Type theorems and least-squares clustering, Algorithmica 20 (1998) 61–76.
[63] Q. Mérigot, A multiscale approach to optimal transport, Comput. Graph. Forum 30 (2011) 1583–1592.
[64] B. Lévy, A numerical algorithm for L^2 semi-discrete optimal transport in 3D, ESAIM Math. Model. Numer. Anal. 49 (6) (2015) 1693–1715.
[65] J. Kitagawa, Q. Mérigot, B. Thibert, Convergence of a Newton algorithm for semi-discrete optimal transport, J. Eur. Math. Soc. 21 (9) (2019) 2603–2651.
[66] D.P. Bourne, P.J.J. Kok, S.M. Roper, W.D.T. Spanjer, Laguerre Tessellations and polycrystalline microstructures: A fast algorithm for generating grains of given volumes, arXiv e-prints, 2019, pp. 1–24, vol. 1912.07188.
[67] R. Quey, L. Renversade, Optimal polyhedral description of 3D polycrystals: Method and application to statistical and synchrotron X-ray diffraction data, Comput. Meth. Appl. Mech. Engnrg. 330 (2018) 308–333.
[68] C. Lautensack, Fitting three-dimensional Laguerre tessellations to foam structures, J. Appl. Stat. 35 (9) (2008) 985–995.
[69] L. Petrich, J. Staněk, M. Wang, D. Westhoff, L. Heller, P. Šittner, C.E. Krill III, V. Beneš, V. Schmidt, Reconstruction of grains in polycrystalline materials from incomplete data using Laguerre tessellations, Microsc. Microanal. 25 (2019) 743–752.

[70] A. Lyckegaard, E.M. Lauridsen, W. Ludwig, R.W. Fonda, H.F. Poulsen, On the use of Laguerre tessellations for representations of 3D grain structures, Adv. Eng. Mater. 13 (3) (2011) 165–170.
[71] J. Nocedal, Updating quasi-Newton matrices with limited storage, Math. Comput. 35 (151) (1980) 773–782.
[72] Y. Malitsky, K. Mishchenko, Adaptive gradient descent without descent, arXiv e-prints, 2019, pp. 1–20, vol. 1910.09529.
[73] J. Barzilai, J.M. Borwein, Two-point step size gradient methods, IMA J. Numer. Anal. 8 (1988) 141–148.
[74] Y. Nesterov, Introductory Lectures on Convex Optimization: A Basic Course, in: Mathematics and its applications, Kluwer Academic Publishers, Springer US, Doordrecht, 2004.
[75] D.G. Anderson, Iterative procedures for nonlinear integral equations, J. ACM 12 (4) (1965) 547–560.
[76] H.-R. Fang, Y. Saad, Two classes of multisecant methods for nonlinear acceleration, Numer. Linear Algebra Appl. 16 (2009) 197–221.
[77] A. Toth, C.T. Kelley, Convergence analysis for Anderson acceleration, SIAM J. Numer. Anal. 53 (2) (2015) 805–819.
[78] C. Evans, S. Pollock, L.G. Rebholz, M. Xiao, A proof that anderson acceleration improves the convergence rate in linearly converging fixed point methods (but not in those converging quadratically), arXiv e-prints, 2018, pp. 1–22, vol. 1810.08455.
[79] J. Nocedal, S.J. Wright, Numerical Optimization, Springer, New York, 1999.
[80] A. Brieden, P. Gritzmann, On optimal weighted balanced clusterings: Gravity bodies and power diagrams, SIAM J. Discrete Math. 26 (2) (2012) 415–434.
[81] Y.-H. Dai, A perfect example for the BFGS method, Math. Program. 138 (1–2) (2013) 501–530.
[82] Y. Saad, M.H. Schultz, GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems, SIAM J. Sci. Stat. Comput. 7 (1986) 856–869.
[83] A. Fu, J. Zhang, S. Boyd, Anderson accelerated Douglas–Rachford splitting, arXiv e-prints, 2019, pp. 1–35, vol. 1908.11482.
[84] B.T. Polyak, Introduction to Optimization, Optimization Software, Inc., New York, 1987.
[85] Y.-H. Dai, L.-Z. Liao, R-linear convergence of the Barzilai and Borwein gradient method, IMA J. Numer. Anal. 22 (2002) 1–10.
[86] W.B. Liu, Y.-H. Dai, Minimization algorithms based on supervisor and searcher cooperation: I - faster and robust gradient algorithms for minimization problems with stronger noises, J. Optim. Theory Appl. 111 (2001) 359–379.
[87] R. Fletcher, On the Barzilai–Borwein method, in: L. Qi, K. Teo, X. Yang (Eds.), Optimization and Control with Applications, Springer US, Boston, MA, 2005, pp. 235–256.
[88] O. Burdakov, Y.-H. Dai, N. Huang, Stabilized Barzilai–Borwein method, J. Comput. Math. 37 (6) (2019) 916–936.
[89] M. Schneider, On the Barzilai–Borwein basic scheme in FFT-based computational homogenization, Internat. J. Numer. Methods Engrg. 118 (2019) 482–494.
[90] L.V. Kantorovich, On Newton's method for functional equations, Dokl. Akad. Nauk SSSR 59 (1948) 1237–1240.
[91] S. Boyd, L. Vandenberghe, Convex Optimization, Cambridge University Press, 2004.
[92] C.G. Broyden, The convergence of a class of double-rank minimization algorithms: 2. The new algorithm, J. Math. Anal. Appl. 6 (1970) 222–231.
[93] R. Fletcher, A new approach to variable metric algorithms, Comput. J. 13 (1970) 317–322.
[94] D. Goldfarb, A family of variable-metric methods derived by variational means, Math. Comput. 24 (1970) 23–26.
[95] D.F. Shanno, Conditioning of quasi-Newton methods for function minimization, Math. Comput. 24 (1970) 647–656.
[96] D. Wicht, M. Schneider, T. Böhlke, On quasi-Newton methods in fast Fourier transform-based micromechanics, Internat. J. Numer. Methods Engrg. online (2020) 1–34.
[97] C.H. Rycroft, Voro++: A three-dimensional voronoi cell library in C++, Chaos 19 (2009) 041111.
[98] I.M. Sobol', On the distribution of points in a cube and the approximate evaluation of integrals , USSR Comput. Math. Math. Phys. 7 (4) (1967) 86–112.
[99] N.L. Johnson, S. Kotz, N. Balakrishnan, 14: Lognormal Distributions, Continuous Univariate Distributions, second ed., in: Wiley Series in Probability and Mathematical Statistics: Applied Probability and Statistics, vol. 1, Wiley, New York, 1994.
[100] E. Anderson, Z. Bai, C. Bischof, L.S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, D. Sorensen, LAPACK Users' Guide, third ed., Society for Industrial and Applied Mathematics, Philadelphia, PA, 1999.
[101] M.P. Allen, D.J. Tildesley, Computer Simulation of Liquids, Clarendon Press, Oxford, 1987.
[102] B. Flipon, C. Keller, R. Quey, F. Barbe, A full-field crystal-plasticity analysis of bimodal polycrystals, Int. J. Solids Struct. 184 (2020) 178–192.
[103] A. Spettl, T. Werz, C.E. Krill III, V. Schmidt, Parametric representation of 3D grain ensembles in polycrystalline microstructures, J. Stat. Phys. 154 (2014) 913–928.
[104] R.S. Dembo, S.C. Eisenstat, T. Steihaug, Inexact Newton methods, SIAM J. Numer. Anal. 19 (1982) 400–408.
[105] P. Lehto, J. Romanoff, H. Remes, T. Sarikka, Characterisation of local grain size variation of welded structural steel, Weld. World 60 (4) (2016) 673–688.
[106] S. Asmussen, P.W. Glynn, Stochastic Simulation: Algorithms and Analysis, Springer, New York, 2007.
[107] J.C. Hateley, H. Wei, L. Chen, Fast methods for computing centroidal Voronoi tessellations, J. Sci. Comput. 63 (2015) 185–212.
[108] R. Quey, A. Villani, C. Maurice, Nearly uniform sampling of crystal orientations, J. Appl. Crystallogr. 51 (4) (2018) 1162–1173.