Online supplementary material for the paper Centroidal power diagrams, Lloyd's algorithm and applications to optimal location problems

D. P. Bourne* S. M. Roper†

July 31, 2015

In this online supplementary material we provide further examples of the algorithm (§1), discuss implementation issues (§2, §3) and include the proofs of some auxiliary results (§4) that there was not space for in the main paper.

1 Further illustrations

1.1 An optimal location problem with non-constant ρ

In the block copolymer example in §1.5.1 and §4 of the main paper we had $\rho = 1$. In an optimal location problem ρ need not be uniform and might represent population density. The term $f(m)$ represents the cost of building or running a facility to serve m individuals. The function f is concave, which represents an economy of scale.

A particular case of interest would be to determine where to locate government agencies (stations) to which people must attend at some rate (for example, a trip to the passport office). Somewhat artificially we may propose that the cost per person of a trip of length l is $\tilde{c}\,c(l/L)$ where L is a representative distance, $\tilde{c}$ is a constant with units of cost per person, and c is a non-dimensional cost function. Let $\{P_i\}_{i=1}^N$ be a power diagram with generators $\{\boldsymbol{x}_i, w_i\}$. We assume that if a person belongs to power cell P_i, then they must use the station located at $\boldsymbol{x}_i$, and that they visit the station ω times per year. Then the cost per year C to the people travelling to the locations $\{\boldsymbol{x}_i\}_{i=1}^N$ is

$$C = \tilde{c}\,\omega \sum_{i=1}^{N} \int_{P_i} c\left(\frac{|\boldsymbol{x} - \boldsymbol{x}_i|}{L}\right) \rho\left(\boldsymbol{x}\right)\, d\boldsymbol{x}.$$

Suppose that the cost per year of running a station that serves m individuals is $sf(m/M)$ where M is a characteristic number of people, s has units of cost per year, and f is a non-dimensional cost function. Using $c(x) = x^2$ we obtain

$$\sum_{i=1}^{N} \left\{ sf\left(\frac{m_i}{M}\right) + \frac{\tilde{c}\omega}{L^2} \int_{P_i} |\boldsymbol{x} - \boldsymbol{x}_i|^2 \rho\left(\boldsymbol{x}\right)\, d\boldsymbol{x} \right\}$$

which represents the combined cost per year of running the stations and the travel costs of the users. This cost must be minimised. By rescaling we obtain the energy

$$E(\{\boldsymbol{x}_i, w_i\}) = \sum_{i=1}^{N} \left\{ \lambda f(m) + \int_{P_i} |\boldsymbol{x} - \boldsymbol{x}_i|^2 \rho\left(\boldsymbol{x}\right)\, d\boldsymbol{x} \right\}.$$

*Department of Mathematical Sciences, Durham University, Science Laboratories, South Rd., Durham, DH1 3LE
†School of Mathematics and Statistics, University of Glasgow, University Gardens, Glasgow, G12 8QW

The parameter λ is a measure of the cost of running a station compared to the cost incurred by the individuals using the station; small values of λ represent a station that is low in cost to run and large values of λ represent a station that is high in cost to run (perhaps because of infrequent visits by its users). As a concrete example we take European population data covering metropolitan France and ask where to cite stations for different choices of λ, using the function $f(m) = -m \log m$. Figure 1 shows the results for two different choices of λ, loosely corresponding to departments/regions and their centres of administration and extents.

1.2 An example in three dimensions: crystallization

In this section we implement the generalized Lloyd algorithm in three dimensions for the block copolymer model described in [4],

$$E\left(\{\boldsymbol{x}_i, w_i\}\right) = \sum_{i=1}^{N} \left\{ \lambda m_i^{\frac{2}{3}} + \int_{P_i} |\boldsymbol{x} - \boldsymbol{x}_i|^2 \, d\boldsymbol{x} \right\} \tag{1}$$

where $\boldsymbol{x}_i \in \Omega \subset \mathbb{R}^3$. It was conjectured in [4] that global minimizers of E tend to a body-centred cubic (BCC) lattice as $\lambda \to 0$, meaning that the set $\{\boldsymbol{x}_i\}$ tends to a BCC lattice and $w_i \to 0$ for all i. This conjecture was motivated by block copolymer experiments and by results for the special case $\lambda = 0$, $w_i = 0$, $N \to \infty$: in [3] it was proved that the BCC lattice has asymptotically the lowest energy amongst all lattices and [5] provided numerical evidence that it has asymptotically the lowest energy amongst all possible configurations $\{\boldsymbol{x}_i\}$. Simulations in [4, Sec. 4.1] in two dimensions demonstrate that global minimizers of E for $\lambda > 0$ (centroidal power diagrams) are close to global minimizers of E for $\lambda = 0$ (centroidal Voronoi tessellations). In this section we give further numerical support for the conjecture.

It is computationally expensive to study the limit $\lambda \to 0$ in three dimensions since the optimal number of generators grows like $N \sim \lambda^{-1}$. Instead we restrict our attention to the case where Ω is a periodic cube. Figure 2 shows a representative Voronoi cell generated by the BCC lattice, which is a truncated octahedron (Kelvin proposed a deformed version of the truncated octahedron as a candidate for three-dimensional foams). If N is chosen appropriately, then N of these cells fit exactly into the periodic cube and there is no boundary layer. If $\{\boldsymbol{z}_i\}_{i=1}^N$ are the centres of these cells, then $\{\boldsymbol{z}_i, 0\}_{i=1}^N$ is a critical point of E (because it is a centroidal Voronoi tessellation and all cells have the same mass). We study its stability using the generalized Lloyd algorithm.

We implemented the algorithm in C++ using the Voro++ software library to compute power diagrams [7]. We found that the BCC lattice is stable under small perturbations; if the initial condition $(\boldsymbol{X}^0, \boldsymbol{w}^0)$ is taken to be a small enough perturbation of the BCC lattice, then the generalized Lloyd algorithm converges back to the BCC lattice. See Figure 3, left column (the initial configuration is top-left, the final configuration is bottom-left). This suggests that the BCC lattice is at least a local minimizer of the energy. Under larger perturbations the Lloyd algorithm converges to a different critical point with a higher energy. See Figure 3, right column (the initial configuration is top-right, the final configuration is bottom-right). We also tested the energy of the BCC lattice against the energy of several common lattices and found that it was lower in each case. Due to the non-convexity and flatness of the energy landscape, however, the conjecture requires a more detailed numerical study.

2 Implementation

The generalized Lloyd algorithm relies upon the computation of power diagrams. In this section we briefly review different methods for the calculation of the power diagram given a domain Ω and generators $\{\boldsymbol{x}_i, w_i\}_{i=1}^N$.

2.1 Half-plane intersection

Recall that $F_{ij} = F_{ji} = P_i \cap P_j$ is the boundary between power cells P_i and P_j. Assume that $P_i \cap P_j \neq \emptyset$ and take two distinct points $\boldsymbol{x}$ and $\boldsymbol{y}$ in F_{ij}. By the definition of the cells P_i and P_j we have $|\boldsymbol{x} - \boldsymbol{x}_i|^2 - w_i = |\boldsymbol{x} - \boldsymbol{x}_j|^2 - w_j$ and $|\boldsymbol{y} - \boldsymbol{x}_i|^2 - w_i = |\boldsymbol{y} - \boldsymbol{x}_j|^2 - w_j$. Subtracting leaves

$$(\boldsymbol{x} - \boldsymbol{y}) \cdot (\boldsymbol{x}_i - \boldsymbol{x}_j) = 0.$$

This establishes that boundaries between cells are planes with the normal to F_{ij} parallel to $\boldsymbol{x}_i - \boldsymbol{x}_j$. A point on the plane can be found by writing $\boldsymbol{p} = \boldsymbol{x}_i + s\,(\boldsymbol{x}_j - \boldsymbol{x}_i)$ and noting that $\boldsymbol{p} \in F_{ij}$ implies

$$|\boldsymbol{p} - \boldsymbol{x}_i|^2 - w_i = |\boldsymbol{p} - \boldsymbol{x}_j|^2 - w_j$$

from which we deduce

$$s = \frac{1}{2} + \frac{w_i - w_j}{2\,|\boldsymbol{x}_i - \boldsymbol{x}_j|^2}, \quad \boldsymbol{p} = \frac{1}{2}\,(\boldsymbol{x}_i + \boldsymbol{x}_j) - \frac{(w_j - w_i)}{2\,|\boldsymbol{x}_j - \boldsymbol{x}_i|^2}\,(\boldsymbol{x}_j - \boldsymbol{x}_i)\,.$$

If we define the *half-plane*

$$H_{ij} = H\,(\boldsymbol{x}_i, w_i, \boldsymbol{x}_j, w_j) = \{\boldsymbol{x} \;:\; \|\boldsymbol{x} - \boldsymbol{x}_i\|^2 - w_i \leq \|\boldsymbol{x} - \boldsymbol{x}_j\|^2 - w_j\} \tag{2}$$

then

$$P_i = \bigcap_{\substack{j=1 \\ j \neq i}}^{j=N} H\,(\boldsymbol{x}_i, w_i, \boldsymbol{x}_j, w_j)\,.$$

The observation that power cells can be expressed as the intersection of half-planes, and the explicit expressions for both a point on the plane and the normal to the plane, is the basis for the *half-plane* method for the computation of a power-diagram [6]. The power cell is built iteratively according to Algorithm 1. The naïve half-plane method sets the cell $\tilde{P}_i = \Omega$ initially, and following repeated

Algorithm 1 The half-plane intersection method, [6].

Require: The set Ω is a convex polyhedron with n_Ω faces, and there are N generators $\{\boldsymbol{x}_i, w_i\}_{i=1}^N$.
  for Generator $(\boldsymbol{x}_i, w_i)$ **do**
    $\tilde{P}_i = \Omega$
    for Generators $(\boldsymbol{x}_j, w_j)$, $j \neq i$ **do**
      Calculate H_{ij}, given by (2)
      $\tilde{P}_i \leftarrow \tilde{P}_i \cap H_{ij}$
    end for
    $P_i \leftarrow \tilde{P}_i$
    return Power cell P_i
  end for
  return The power diagram composed of at most N power cells, $\{P_i\}$

intersections with half-planes H_{ij} forms the power cell P_i. As discussed in [6] for Voronoi diagrams, the construction of each cell requires $N-1$ half-plane intersections and the number of operations in each intersection depends upon the number of faces of the cell $\tilde{P}_i$ (we must check whether the boundary of the new half-plane intersects with any of the faces of $\tilde{P}_i$). At worst, each half-plane intersection increases the number of faces by 1. If initially the cell has n_Ω faces, then the total number of checks is at most $n_\Omega + (n_\Omega + 1) + \ldots + (n_\Omega + (N-2)) = (N-2)n_\Omega + (N-1)(N-2)/2 = O(N^2)$. The intersections must be performed to create each cell so the overall time complexity of this method is *at worst* $O(N^3)$ and is usually $O(N^2)$. The complexity of $O(N^3)$ can be improved using more

efficient half-plane intersection algorithms; N half-planes can be intersected using a divide-and-conquer method in $\mathcal{O}(N \log N)$ time [2, §3.1], which means that a power diagram can be constructed using the half-plane intersection method in $\mathcal{O}(N^2 \log N)$ time.

More sophisticated algorithms can construct power diagrams in the worst-case optimal time of $\mathcal{O}(N \log N)$ in $\mathbb{R}^2$ and $\mathcal{O}(N^2)$ in $\mathbb{R}^3$. We briefly describe one such optimal algorithm in the following section.

2.2 Lifting method

An optimal algorithm for the computation of the power diagram is given in [1], in which the generators $\{\boldsymbol{x}_i, w_i\}_{i=1}^N$ are lifted into $\mathbb{R}^{d+1}$. Given a generator $(\boldsymbol{x}, w)$, where $\boldsymbol{x}$ has components x_j, $j = 1, \ldots, d$, the lifted generator is the vector in $\mathbb{R}^{d+1}$ with components $(x_1, x_2, \ldots, x_d, z)$ where $z = |\boldsymbol{x}|^2 - w$. In the power diagram computation the *lower convex hull* of the lifted generators is found, giving rise to a regular triangulation of the generators. The j-faces of the triangulation (for example in two dimensions the 0-faces are the generators, the 1-faces are the edges and the 2-faces are the triangles) are then transformed into $(d-j)$-faces via a *polar map*. The result of this is that the triangulation formed by the lower convex hull of the lifted generators is transformed into the power diagram based on the generators. The expensive step in this calculation is the calculation of the lower convex hull of a set of points in $\mathbb{R}^{d+1}$. When $d = 2$ then convex hull algorithms with complexity $O(N \log N)$ can be used.

2.3 Other implementation issues

Once the power cells are obtained, the centroid and the mass of cell P_i need to be determined to perform a step of the generalized Lloyd algorithm. When using constant ρ the integral results given in the online supplementary material (§3) allow fast computation of the integrals required. When using non-constant ρ we employ quadrature: the cells are triangulated and each triangle mapped to a reference triangle on which an N-point (we use $N = 31$) quadrature rule is applied.

It is worth noting that the generalised Lloyd algorithm converges to local minima and a strategy must be adopted to find global minima. In our simulations we start with a large number of random initial configurations, apply the algorithm and periodically sort the results. We then continue using the algorithm on a subset of configurations that are the lowest energy states. In this way we search for global minima, although we cannot guarantee to find them with this heuristic method.

3 Useful implementation formulas for the case $\rho =$ constant

In two dimensions in the special case that $\rho(\boldsymbol{x}) = \rho_0$, a constant, the integrals defining the mass and centroid of a polygonal cell can be computed without using a quadrature rule; they can be expressed explicitly as functions of the vertices of the cell.

Consider a polygon P, lying in the plane with normal $\boldsymbol{k}$ with vertices $\boldsymbol{v}_k$ for $k = 0, \ldots, N-1$. We use the notation $k \oplus m = (k+m) \mod N$. The vertices are numbered anti-clockwise around the boundary (with respect to the normal $\boldsymbol{k}$). We denote the outward normal to the polygon by $\boldsymbol{\nu}$. On edge k (with end points $\boldsymbol{v}_k$ and $\boldsymbol{v}_{k\oplus 1}$ and length L_k) $\boldsymbol{\nu}$ can be written in terms of the vertices as

$$\boldsymbol{\nu} = \frac{1}{L_k} (\boldsymbol{v}_{k\oplus 1} - \boldsymbol{v}_k) \times \boldsymbol{k}.$$

As the density is constant, we can use the Divergence Theorem to express integrals over polygons as edge integrals:

$$\int_P \rho_0 \, d\boldsymbol{x} = \frac{1}{2}\rho_0 \int_{\partial P} \boldsymbol{x} \cdot \boldsymbol{\nu} \, ds = \frac{1}{2}\rho_0 \sum_{\text{edges } k} \int_{\boldsymbol{v}_k}^{\boldsymbol{v}_{k\oplus 1}} \boldsymbol{x} \cdot \boldsymbol{\nu} \, ds.$$

Along an edge, using an arc-length parametrization,

$$\boldsymbol{x} = \frac{s}{L_k}\boldsymbol{v}_{k\oplus 1} + \frac{(L_k - s)}{L_k}\boldsymbol{v}_k, \quad \boldsymbol{x}\cdot\boldsymbol{\nu} = \frac{1}{L_k}\left[\boldsymbol{k}\cdot(\boldsymbol{v}_k \times \boldsymbol{v}_{k\oplus 1})\right].$$

Since $\boldsymbol{x}\cdot\boldsymbol{\nu}$ is independent of s we obtain

$$\int_P \rho_0 \, d\boldsymbol{x} = \frac{1}{2}\rho_0 \sum_{\text{edges } k} \left[\boldsymbol{k}\cdot(\boldsymbol{v}_k \times \boldsymbol{v}_{k\oplus 1})\right]. \tag{3}$$

Similarly

$$\int_P \rho_0 \boldsymbol{x} \, d\boldsymbol{x} = \frac{1}{6}\rho_0 \sum_{\text{edges } k} (\boldsymbol{v}_k + \boldsymbol{v}_{k\oplus 1})\left[\boldsymbol{k}\cdot(\boldsymbol{v}_k \times \boldsymbol{v}_{k\oplus 1})\right] \tag{4}$$

and

$$\int_P \rho_0 \boldsymbol{x}\otimes\boldsymbol{x} \, d\boldsymbol{x} = \tag{5}$$
$$\frac{1}{24}\rho_0 \sum_{\text{edges } k} \left[2\boldsymbol{v}_k\otimes\boldsymbol{v}_k + 2\boldsymbol{v}_{k\oplus 1}\otimes\boldsymbol{v}_{k\oplus 1} + \boldsymbol{v}_k\otimes\boldsymbol{v}_{k\oplus 1} + \boldsymbol{v}_{k\oplus 1}\otimes\boldsymbol{v}_k\right]\left[\boldsymbol{k}\cdot(\boldsymbol{v}_k \times \boldsymbol{v}_{k\oplus 1})\right].$$

4 Proofs of auxiliary results

Proof of Proposition 2.11.
(2) $\Longrightarrow$ (1): Suppose that there exists an admissible f satisfying $f'(m_i) = -w_i$. Suppose for contradiction that there exists i, j with $w_i > w_j$, $m_i < m_j$. Then

$$(f'(m_j) - f'(m_i))(m_j - m_i) = (w_i - w_j)(m_j - m_i) > 0,$$

which contradicts the fact that f' is non-increasing. The second condition is trivial; if $m_i = m_j$, then $w_i = -f'(m_i) = -f'(m_j) = w_j$.
(1) $\Longrightarrow$ (2): We construct an example of an admissible f. There are infinitely many possibilities. Without loss of generality we can assume that $w_i \leq 0$ for all i (by subtracting a constant from all the weights if necessary). We can also assume that $m_i \neq m_j$ for $i \neq j$ so that we have N distinct constraints $f'(m_i) = -w_i$ (otherwise simply relabel N). Furthermore we can assume that

$$w_1 \leq w_2 \leq \cdots \leq w_N \leq 0$$

by relabelling the w_i if necessary. Then by assumption

$$0 < m_1 < m_2 < \cdots < m_N.$$

We define a sequence $\{f_i\}_{i=1}^N$ and an admissible function f such that $f(m_i) = f_i$, $f'(m_i) = -w_i$. Choose any $f_0 > 0$. This will be the value of f at 0. Define $f_1 = f_0 - w_1 m_1$. If $w_2 = w_1$, define $f_2 = f_1 - w_1(m_2 - m_1)$. Otherwise define $f_2 = f_1 - w_1(m_2 - m_1) - \varepsilon_2$ for some $\varepsilon_2 > 0$ to be determined. Note that (m_2, f_2) lies below the line through (m_1, f_1) with slope $-w_1 \geq 0$. To ensure that (m_1, f_1) lies below the line through (m_2, f_2) with slope $-w_2$ we require that

$$f_1 < f_2 - w_2(m_1 - m_2) \quad \Longleftrightarrow \quad \varepsilon_2 < (m_2 - m_1)(w_2 - w_1).$$

Repeat this process: If $w_3 = w_2$ define $f_3 = f_2 - w_2(m_3 - m_2)$. Otherwise define $f_3 = f_2 - w_2(m_3 - m_2) - \varepsilon_3$ for some $0 < \varepsilon_3 < (m_3 - m_2)(w_3 - w_2)$, etc. Define $\tilde{f} : [0, \infty) \to \mathbb{R}$ by

$$\tilde{f}(m) = \min_i (f_i - w_i(m - m_i)).$$

Then $\tilde{f}$ satisfies $\tilde{f}(m_i) = f_i$, $\tilde{f}'(m_i) = -w_i$, $\tilde{f}(0) > 0$. Also $\tilde{f}$ is a minimum of concave functions and so is concave. By modifying $\tilde{f}$ in some small regions away from $\{m_i\}$ we obtain a C^1 function f with the same properties.

Note that for computational purposes, when checking the stability of a centroidal power diagram, it is sufficient to take $f = \tilde{f}$ even though $\tilde{f} \notin C^1$ since stability is determined by the behaviour of f near the critical point, where $\tilde{f}$ is smooth. □

Proof of Proposition 3.6. Recall that

$$m_i^n = \int_{P_i(\boldsymbol{X}^n, \boldsymbol{w}^n)} \rho(\boldsymbol{x})\, d\boldsymbol{x}$$

and $\hat{\boldsymbol{M}}_n = \operatorname{diag}(m_1^n \boldsymbol{I}_d, \dots, m_N^n \boldsymbol{I}_d)$. Equation (2.12) implies that

$$\begin{pmatrix} \nabla_{\boldsymbol{X}} E^n \\ \nabla_{\boldsymbol{w}} E^n \end{pmatrix} = \begin{pmatrix} 2\hat{\boldsymbol{M}}_n & \nabla_{\boldsymbol{X}} \boldsymbol{m}^n \\ \boldsymbol{0} & \nabla_{\boldsymbol{w}} \boldsymbol{m}^n \end{pmatrix} \begin{pmatrix} \boldsymbol{X}^n - \boldsymbol{X}^{n+1} \\ \boldsymbol{w}^n - \boldsymbol{w}^{n+1} \end{pmatrix}, \tag{6}$$

where $\boldsymbol{0}$ is the N-by-(Nd) zero matrix. By Lemma 2.4, the matrix on the right-hand side has a one-dimensional nullspace. Therefore rewriting these equations in the form (3.2) requires some care.

Let $\boldsymbol{e}_1, \dots, \boldsymbol{e}_N$ be the standard basis vectors for $\mathbb{R}^N$. We introduce the new basis

$$\boldsymbol{f}_1 := \boldsymbol{e}_1 - \boldsymbol{e}_2, \quad \boldsymbol{f}_2 := \boldsymbol{e}_2 - \boldsymbol{e}_3, \quad \dots, \quad \boldsymbol{f}_{N-1} := \boldsymbol{e}_{N-1} - \boldsymbol{e}_N, \quad \boldsymbol{f}_N := \boldsymbol{e}_1 + \cdots + \boldsymbol{e}_N.$$

Note that $\boldsymbol{f}_N$ spans the null space of $\nabla_{\boldsymbol{w}} \boldsymbol{m}^n$. Let P be the invertible change-of-basis matrix satisfying $P\boldsymbol{f}_i = \boldsymbol{e}_i$. Let $\Pi : \mathbb{R}^N \to \mathbb{R}^{N-1}$ be the projection onto $\{\boldsymbol{f}_N\}^\perp$: $\Pi = (\boldsymbol{I}_{N-1} | \boldsymbol{0})$ where $\boldsymbol{0}$ is the $(N-1)$-by-1 zero vector. Observer that for all $\boldsymbol{y} \in \mathbb{R}^N$

$$\nabla_{\boldsymbol{w}} \boldsymbol{m}^n\, \boldsymbol{y} = \nabla_{\boldsymbol{w}} \boldsymbol{m}^n\, P^{-1} \Pi^T \Pi P\, \boldsymbol{y} \tag{7}$$

since $\nabla_{\boldsymbol{w}} \boldsymbol{m}^n\, \boldsymbol{f}_N = \boldsymbol{0}$, $\Pi P\, \boldsymbol{f}_N = \boldsymbol{0}$, and $\Pi^T \Pi\, \boldsymbol{e}_i = \boldsymbol{e}_i$ for all $i \in \{1, \dots, N-1\}$. We check that the following $(N-1)$-by-$(N-1)$ matrix is invertible:

$$A_n := \Pi P\, \nabla_{\boldsymbol{w}} \boldsymbol{m}^n P^{-1} \Pi^T. \tag{8}$$

If $A_n \boldsymbol{x} = \boldsymbol{0}$, then $P\, \nabla_{\boldsymbol{w}} \boldsymbol{m}^n P^{-1} \Pi^T\, \boldsymbol{x} = c\, \boldsymbol{e}_N$ for some $c \in \mathbb{R}$, and so $\nabla_{\boldsymbol{w}} \boldsymbol{m}^n P^{-1} \Pi^T\, \boldsymbol{x} = c \boldsymbol{f}_N$. But $\boldsymbol{f}_N^T \nabla_{\boldsymbol{w}} \boldsymbol{m}^n = (\nabla_{\boldsymbol{w}} \boldsymbol{m}^n\, \boldsymbol{f}_N)^T = \boldsymbol{0}$ and thus $c = 0$. Therefore, by Lemma 2.4, $P^{-1} \Pi^T\, \boldsymbol{x} = a \boldsymbol{f}_N$ for some $a \in \mathbb{R}$. It follows from the definitions of P and Π that $a = 0$ and $\boldsymbol{x} = \boldsymbol{0}$.

Using equations (7) and (8), we see that the equation

$$\nabla_{\boldsymbol{w}} E^n = \nabla_{\boldsymbol{w}} \boldsymbol{m}^n (\boldsymbol{w}^n - \boldsymbol{w}^{n+1})$$

can be inverted to give

$$A_n^{-1} \Pi P\, \nabla_{\boldsymbol{w}} E^n = \Pi P (\boldsymbol{w}^n - \boldsymbol{w}^{n+1}).$$

Therefore

$$\boldsymbol{w}^{n+1} = \boldsymbol{w}^n - P^{-1} \Pi^T A_n^{-1} \Pi P\, \nabla_{\boldsymbol{w}} E^n + c \boldsymbol{f}_N \tag{9}$$

for some $c \in \mathbb{R}$. We conclude from equations (6) and (9) that

$$\begin{pmatrix} \boldsymbol{X}^{n+1} \\ \boldsymbol{w}^{n+1} \end{pmatrix} = \begin{pmatrix} \boldsymbol{X}^n \\ \boldsymbol{w}^n \end{pmatrix} - \boldsymbol{B}_n \begin{pmatrix} \nabla_{\boldsymbol{X}} E^n \\ \nabla_{\boldsymbol{w}} E^n \end{pmatrix} + \begin{pmatrix} \boldsymbol{0} \\ c \boldsymbol{f}_N \end{pmatrix}$$

where $\boldsymbol{B}_n$ is the matrix

$$\boldsymbol{B}_n = \begin{pmatrix} \frac{1}{2} \hat{\boldsymbol{M}}_n^{-1} & -\frac{1}{2} \hat{\boldsymbol{M}}_n^{-1} \nabla_{\boldsymbol{X}} \boldsymbol{m}^n P^{-1} \Pi^T A_n^{-1} \Pi P \\ \boldsymbol{0} & P^{-1} \Pi^T A_n^{-1} \Pi P \end{pmatrix},$$

where $\boldsymbol{0}$ is the N-by-(Nd) zero matrix. This completes the proof. □

Proof of Proposition 3.8. Since $\omega_i = -f'(m_i)$, then the partial derivatives of $\boldsymbol{\omega}$ are obtained immediately from Lemma 2.4. Obtaining the partial derivatives of $\boldsymbol{\xi} = \frac{1}{m_i}\int_{P_i} \boldsymbol{x}\rho\, dx$ requires a bit more work. Observe that

$$\frac{\partial \boldsymbol{\xi}_i}{\partial \boldsymbol{x}_j} = \frac{1}{m_i}\left(\frac{\partial(m_i\boldsymbol{\xi}_i)}{\partial \boldsymbol{x}_j} - \boldsymbol{\xi}_i \otimes \frac{\partial m_i}{\partial \boldsymbol{x}_j}\right), \quad \frac{\partial \boldsymbol{\xi}_i}{\partial w_j} = \frac{1}{m_i}\left(\frac{\partial(m_i\boldsymbol{\xi}_i)}{\partial w_j} - \frac{\partial m_i}{\partial w_j}\boldsymbol{\xi}_i\right). \tag{10}$$

Lemma 2.4 gives $\partial m_i/\partial \boldsymbol{x}_j$, $\partial m_i/\partial w_j$ and so we just need to compute $\partial(m_i\boldsymbol{\xi}_i)/\partial \boldsymbol{x}_j$, $\partial(m_i\boldsymbol{\xi}_i)/\partial w_j$, i.e., compute the partial derivatives of

$$(m_i\boldsymbol{\xi}_i)(\boldsymbol{X}, \boldsymbol{w}) = \int_{P_i(\boldsymbol{X},\boldsymbol{w})} \boldsymbol{x}\rho(\boldsymbol{x})\, d\boldsymbol{x}.$$

The computation is similar to the proof of Lemma 2.4 and so we just sketch the details. Consider the same 1-parameter family of power diagrams used in the proof of Lemma 2.4: $\{P_i^t\} = \{\varphi^t(P_i)\}$. As for equation (2.19),

$$\left.\frac{d}{dt}\right|_{t=0} (m_i\boldsymbol{\xi}_i)(\boldsymbol{X}^t, \boldsymbol{w}^t) = \sum_{j=1}^{N} \frac{\partial(m_i\boldsymbol{\xi}_i)}{\partial \boldsymbol{x}_j}\tilde{\boldsymbol{x}}_j + \frac{\partial(m_i\boldsymbol{\xi}_i)}{\partial w_j}\tilde{w}_j = \sum_{k\in J_i}\int_{F_{ik}} \boldsymbol{x}\rho(\boldsymbol{x}) V\cdot \boldsymbol{n}_{ik}\, dS$$

where $V(\boldsymbol{x}) = \frac{d}{dt}\varphi^t(\boldsymbol{x})|_{t=0}$. Combining this with equation (2.21) gives

$$\begin{aligned}
&\sum_{j=1}^{N} \frac{\partial(m_i\boldsymbol{\xi}_i)}{\partial \boldsymbol{x}_j}\tilde{\boldsymbol{x}}_j + \frac{\partial(m_i\boldsymbol{\xi}_i)}{\partial w_j}\tilde{w}_j \\
&= \sum_{k\in J_i}\int_{F_{ik}} \boldsymbol{x}\rho(\boldsymbol{x})\left[\frac{(\boldsymbol{x}-\boldsymbol{x}_i)\cdot\tilde{\boldsymbol{x}}_i - (\boldsymbol{x}-\boldsymbol{x}_k)\cdot\tilde{\boldsymbol{x}}_k}{d_{ik}} + \frac{\tilde{w}_i - \tilde{w}_k}{2d_{ik}}\right] dS \\
&= \sum_{k\in J_i}\frac{m_{ik}}{d_{ik}}\left[(\mathcal{S}(F_{ik}) - \overline{\boldsymbol{x}}_{ik}\otimes \boldsymbol{x}_i)\tilde{\boldsymbol{x}}_i - (\mathcal{S}(F_{ik}) - \overline{\boldsymbol{x}}_{ik}\otimes \boldsymbol{x}_k)\tilde{\boldsymbol{x}}_k + \frac{\overline{x}_{ik}(\tilde{w}_i - \tilde{w}_k)}{2}\right]
\end{aligned} \tag{11}$$

where the matrix $\mathcal{S}(F_{ik})$ was defined in the statement of the proposition. By combining equations (10) and (11) (with suitable choices of $\tilde{\boldsymbol{X}}$ and $\tilde{\boldsymbol{w}}$) and Lemma 2.4 we obtain the desired expressions for $\partial\boldsymbol{\xi}_i/\partial\boldsymbol{x}_j$ and $\partial\boldsymbol{\xi}_i/\partial w_j$. □

References

[1] F. Aurenhammer, *Power diagrams: properties, algorithms, and applications*, SIAM J. Comput., 16 (1987), pp. 78–96.

[2] F. Aurenhammer, R. Klein, and Lee D.-T., *Voronoi Diagrams and Delaunay Triangulations*, World Scientific, 2013.

[3] E. S. Barnes and N. J. A. Sloane, *The optimal lattice quantizer in three dimensions*, SIAM Journal on Algebraic and Discrete Methods, 4 (1983), pp. 30–41.

[4] D.P. Bourne, M.A. Peletier, and S.M. Roper, *Hexagonal patterns in a simplified model for block copolymers*, SIAM J. Appl. Math., 74 (2014), pp. 1315–1337.

[5] Q. Du and D. S. Wang, *The optimal centroidal Voronoi tessellations and the Gersho's conjecture in the three-dimensional space*, Comput. Math. Appl., 49 (2005), pp. 1355–1373.

[6] A. Okabe, B. Boots, K. Sugihara, and S. N. Chiu, *Spatial Tesselations. Concepts and Applications of Voronoi Diagrams*, Wiley, second edition ed., 2000.

[7] C.H. Rycroft, *Voro++: A three-dimensional Voronoi cell library in C++*, Chaos, 19 (2009).

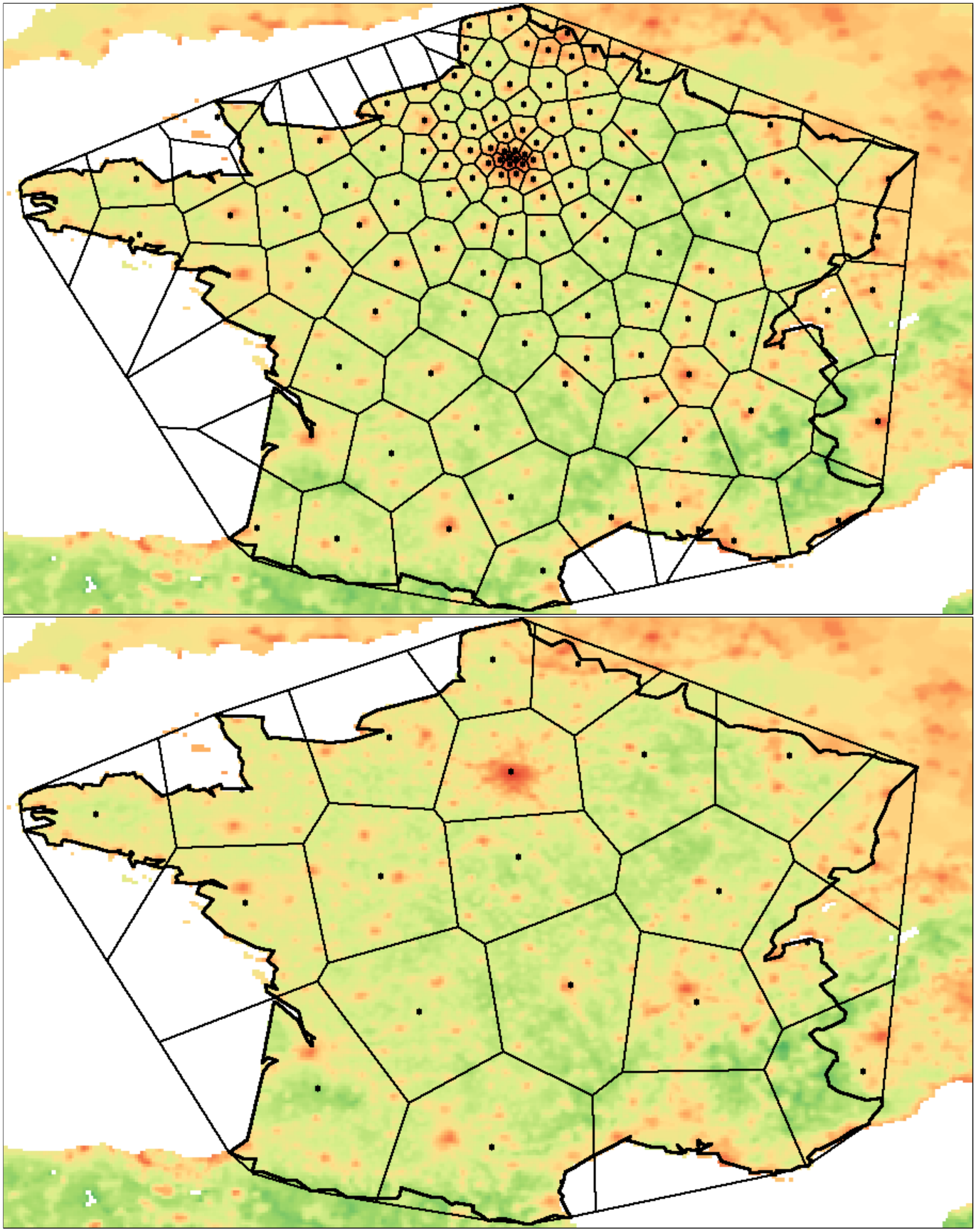

Figure 1: An example illustrating the algorithm applied to the convex hull of metropolitan France (recall that we require the region Ω to be convex). The colour represents the population density ρ, red is high density and green is low density (population data obtained from Eurostat). Areas in which no data was available have been assigned a population density of zero, for example in the seas and oceans. The figures were produced using the generalised Lloyd algorithm with an initial condition of 5000 random generators and 9000 iterations. The top figure has $\lambda = 0.01$ and the bottom figure has $\lambda = 20$. In the particular instances here, the final local minimum of the energy has 128 cells when $\lambda = 0.1$ and 21 cells when $\lambda = 20$.

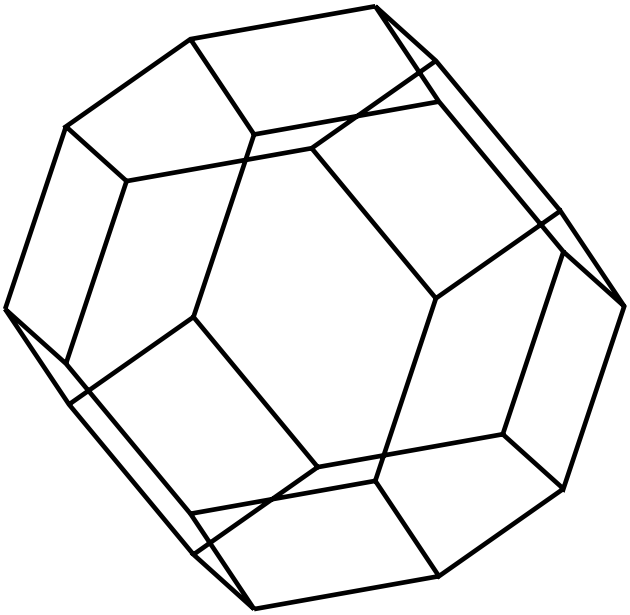

Figure 2: A Voronoi cell generated by the BCC lattice.

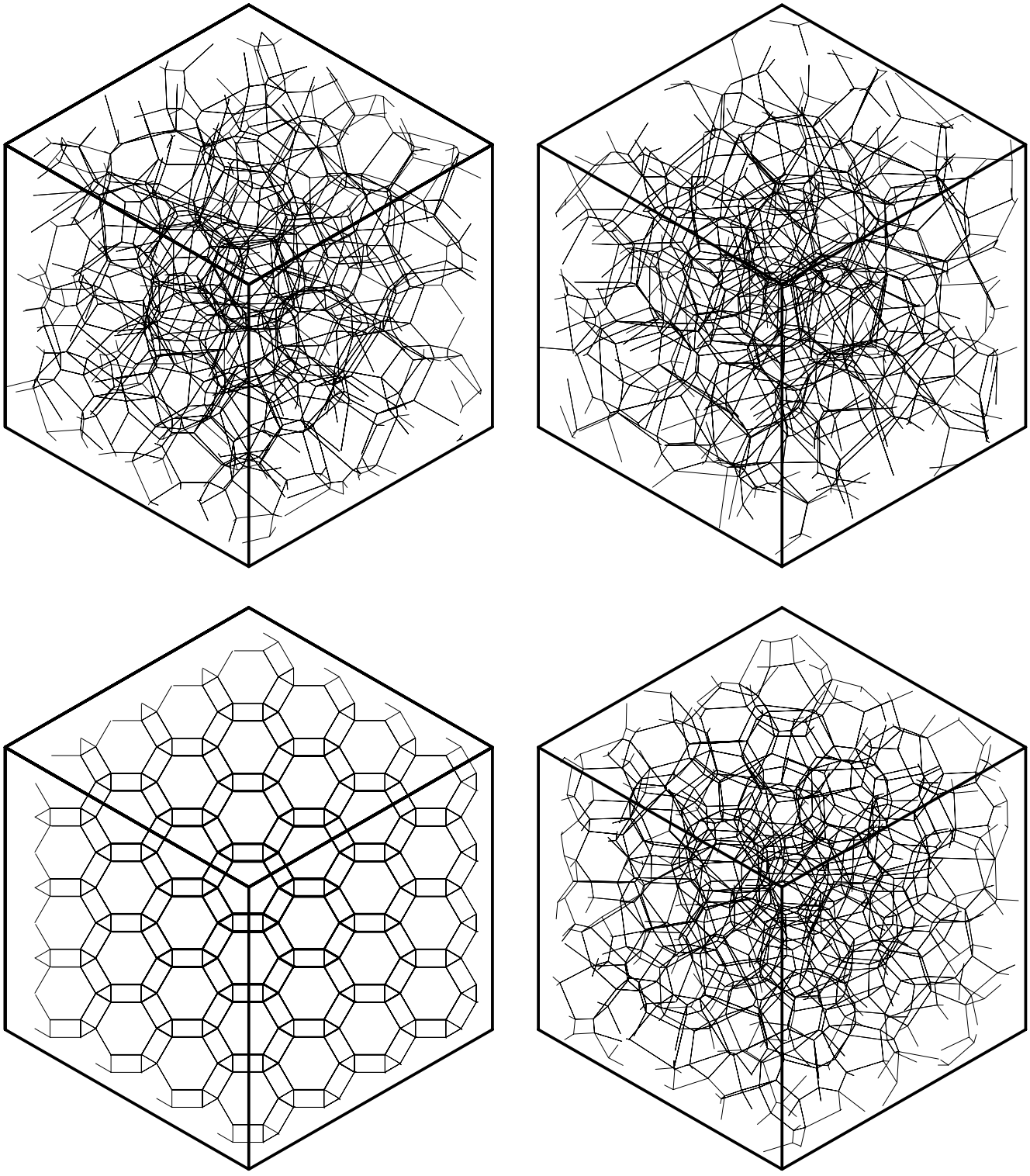

Figure 3: Numerical evidence that the BCC lattice is a local minimizer of (1) for $\lambda = 10^{-3}$. Left column: the initial condition (top-left) is a perturbation of the BCC lattice (a perturbation of both the generator locations and weights). The perturbation is small enough that the algorithm converges to the BCC lattice (bottom-left, configuration after 2000 iterations). Right column: the initial condition (top-right) is a large enough perturbation of the BCC lattice to cause the algorithm to converge to a different local minimum (bottom-right, configuration after 2000 iterations). In all figures $N = 128$.