NUMERICAL LINEAR ALGEBRA WITH APPLICATIONS
Numer. Linear Algebra Appl. 2006; **13**:173–192
Published online 7 February 2006 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/nla.476

Acceleration schemes for computing centroidal Voronoi tessellations

Qiang Du[1,‡] and Maria Emelianenko[2,*,†]

[1]*Department of Mathematics, Pennsylvania State University, University Park, PA 16802, U.S.A.*
[2]*Department of Mathematical Sciences, Carnegie Mellon University, Pittsburgh, PA 15213, U.S.A.*

SUMMARY

Centroidal Voronoi tessellations (CVT) have diverse applications in many areas of science and engineering. The development of efficient algorithms for their construction is a key to their success in practice. In this paper, we study some new algorithms for the numerical computation of the CVT, including the Lloyd–Newton iteration and the optimization based multilevel method. Both theoretical analysis and computational simulations are conducted. Rigorous convergence results are presented and significant speedup in computation is demonstrated through the comparison with traditional methods. Copyright © 2006 John Wiley & Sons, Ltd.

KEY WORDS: centroidal Voronoi tessellations; computational algorithms; Lloyd's method; Newton's method; multilevel method; uniform convergence

1. INTRODUCTION

Centroidal Voronoi tessellations (CVT) have broad applications and are linked to many important concepts defined in different contexts. Let us take the subject of vector quantization as an example [1]. A vector quantizer maps N-dimensional vectors in the domain $\Omega \subset \mathbb{R}^N$ into a finite set of vectors $\{\mathbf{z}_i\}_{i=1}^k$. Each vector $\mathbf{z}_i$ is called a code vector or a *codeword*, and the set of all the codewords is called a codebook. A special quantization scheme is given by the Voronoi tessellation which associates with each codeword $\mathbf{z}_i$, also called a *generator*, a nearest neighbour region that is called a Voronoi region $\{V_i\}_{i=1}^k$. That is, for each i, V_i consists of all points in the domain Ω that are closer to $\mathbf{z}_i$ than to all the other generating points, and a Voronoi tessellation refers to the tessellation of a given domain by the Voronoi regions $\{V_i\}_{i=1}^k$ associated with a set of given generating points $\{\mathbf{z}_i\}_{i=1}^k \subset \Omega$. For a given

*Correspondence to: M. Emelianenko, Department of Mathematical Sciences, Carnegie Mellon University, Pittsburgh, PA 15213, U.S.A.
†E-mail: masha@cmu.edu
‡E-mail: qdu@math.psu.edu

Contract/grant sponsor: NSF; contract/grant numbers: DMR-0205232, DMS-0409297

Received 15 May 2005
Revised 11 November 2005
Accepted 15 November 2005

density function ρ defined on Ω, we may define the centroids, or mass centres, of regions $\{V_i\}_{i=1}^k$ by

$$\mathbf{z}_i^* = \left(\int_{V_i} \mathbf{y}\rho(\mathbf{y})\,d\mathbf{y}\right)\left(\int_{V_i} \rho(\mathbf{y})\,d\mathbf{y}\right)^{-1} \tag{1}$$

An *optimal quantization* may then be defined through a CVT which is a special Voronoi tessellation whose generators coincide with the centroids of their respective Voronoi regions, i.e. $\mathbf{z}_i = \mathbf{z}_i^*$ for all i. Such a connection between CVTs and optimal quantization schemes has been explored extensively in the literature [2].

Given a set of points $\{\mathbf{z}_i\}_{i=1}^k$ and a tessellation $\{V_i\}_{i=1}^k$ of the domain, we may define the *energy functional* or the *distortion value* for the pair $(\{\mathbf{z}_i\}_{i=1}^k, \{V_i\}_{i=1}^k)$ by

$$\mathscr{H}(\{\mathbf{z}_i\}_{i=1}^k, \{V_i\}_{i=1}^k) = \sum_{i=1}^k \int_{V_i} \rho(\mathbf{y})\,|\,\mathbf{y} - \mathbf{z}_i|^2\,d\mathbf{y} \tag{2}$$

The minimizer of $\mathscr{H}$, that is, the optimal quantizer, necessarily forms a CVT which illustrates the optimization property of the CVT [2]. The terms optimal quantizer and CVT are thus to be used interchangeably in the sequel. We note that, besides providing an optimal least square vector quantizer design in electrical engineering applications [1, 3, 4], the CVT concept also has applications in diverse areas such as astronomy, biology, image and data analysis, resource optimization, sensor networks, geometric design, and numerical partial differential equations [2, 5–14]. We refer to Reference [2] for a more comprehensive review of the mathematical theory and diverse applications of CVTs.

In the seminal work of Lloyd on the least square quantization [15], one of the algorithms proposed for computing optimal quantizers is an iterative algorithm consisting of the following simple steps: starting from an initial quantization (a Voronoi tessellation corresponding to an old set of generators), a new set of generators is defined by the mass centres of the Voronoi regions. This process is continued until certain stopping criterion is met. It is easy to see that the Lloyd algorithm is an energy descent iteration of the energy functional (2), which gives strong indications to its practical convergence. We refer to Reference [16] for some discussion on the recent development of a rigorous convergence theory.

Lloyd's algorithms and their variants have been proposed and studied in many contexts for different applications [1, 7, 17, 18]. A particular extension using parallel and probabilistic sampling was given in Reference [6] which allows efficient and mesh free implementation of the Lloyd's algorithm. Still, Lloyd algorithm is at best linearly convergent, besides it slows down as the number of generators gets large.

For modern applications of the CVT concept in large scale scientific and engineering problems such as data communication and mesh generation, efficient algorithms for computing the CVTs play crucial roles. The objective of this paper is to present a number of different approaches for speeding up the convergence of Lloyd iteration. In Section 2 we consider a direct application of the Newton method and propose a coupled Lloyd–Newton scheme that can be used to accelerate the convergence of the original method. Both analytical and numerical results for scalar and vector quantization problems are discussed. Then in Sections 3 and 4 we introduce the ideas of multilevel algorithms and outline the two main strategies for applying them in the non-linear optimization context. In particular, we focus on the new multilevel

approach to the optimal quantization problem developed recently in References [19, 20]. In this paper we discuss the main characteristics of this scheme, such as the dynamic non-linear preconditioning and uniform convergence with respect to the problem size, and show some numerical results demonstrating its superiority over traditional methods.

2. LLOYD–NEWTON METHOD

2.1. Some technical lemmas

For a general non-linear system $\mathbf{f}(\mathbf{z})=0$ with vector argument $\mathbf{z}$, the Newton iteration is given by

$$\mathbf{z}_n = \mathbf{z}_{n-1} - \mathbf{df}|_{\mathbf{z}_{n-1}}^{-1}\mathbf{f}(\mathbf{z}_{n-1})$$

where $\mathbf{df}$ is the Jacobian matrix of the map $\mathbf{f}$. Applying Lloyd's algorithm to the computation of CVTs, we obtain the problem of solving $\mathbf{T}(\mathbf{z}_{n-1})=\mathbf{z}_n$, where $\mathbf{T}$ denotes the Lloyd map from generators to centroids, as discussed earlier. Newton's method in this setting takes on the form

$$\mathbf{z}_n = \mathbf{z}_{n-1} + (\mathbf{dT}|_{\mathbf{z}_{n-1}} - \mathbf{I})^{-1}(\mathbf{z}_{n-1} - \mathbf{T}(\mathbf{z}_{n-1}))$$

Here $\mathbf{T} : \mathbb{R}^{kN} \to \mathbb{R}^{kN}$ and the corresponding Jacobi matrix $\mathbf{dT}=(\partial \mathbf{T}_i/\partial \mathbf{z}_j)$ has dimensions $kN \times kN$

$$\mathbf{dT} = \begin{pmatrix} \frac{\partial \mathbf{T}_1^{(1)}}{\partial \mathbf{z}_1^{(1)}} & \cdots & \frac{\partial \mathbf{T}_1^{(1)}}{\partial \mathbf{z}_k^{(1)}} & \cdots & \frac{\partial \mathbf{T}_1^{(1)}}{\partial \mathbf{z}_1^{(N)}} & \cdots & \frac{\partial \mathbf{T}_1^{(1)}}{\partial \mathbf{z}_k^{(N)}} \\ \vdots & \cdots & \vdots & \cdots & \vdots & \cdots & \vdots \\ \frac{\partial \mathbf{T}_1^{(N)}}{\partial \mathbf{z}_1^{(1)}} & \cdots & \frac{\partial \mathbf{T}_1^{(N)}}{\partial \mathbf{z}_k^{(1)}} & \cdots & \frac{\partial \mathbf{T}_1^{(N)}}{\partial \mathbf{z}_1^{(N)}} & \cdots & \frac{\partial \mathbf{T}_1^{(N)}}{\partial \mathbf{z}_k^{(N)}} \\ \vdots & \cdots & \vdots & \cdots & \vdots & \cdots & \vdots \\ \frac{\partial \mathbf{T}_k^{(1)}}{\partial \mathbf{z}_1^{(1)}} & \cdots & \frac{\partial \mathbf{T}_k^{(1)}}{\partial \mathbf{z}_k^{(1)}} & \cdots & \frac{\partial \mathbf{T}_k^{(1)}}{\partial \mathbf{z}_1^{(N)}} & \cdots & \frac{\partial \mathbf{T}_k^{(1)}}{\partial \mathbf{z}_k^{(N)}} \\ \vdots & \cdots & \vdots & \cdots & \vdots & \cdots & \vdots \\ \frac{\partial \mathbf{T}_k^{(N)}}{\partial \mathbf{z}_1^{(1)}} & \cdots & \frac{\partial \mathbf{T}_k^{(N)}}{\partial \mathbf{z}_k^{(1)}} & \cdots & \frac{\partial \mathbf{T}_k^{(N)}}{\partial \mathbf{z}_1^{(N)}} & \cdots & \frac{\partial \mathbf{T}_k^{(N)}}{\partial \mathbf{z}_k^{(N)}} \end{pmatrix}$$

We arrive at a necessity to calculate the partial derivatives of $\mathbf{T}_i(\mathbf{z})$. The following result (see Reference [2]) is of use.

Lemma 2.1
Let $\Omega = \Omega(\mathbf{U})$ be a region that depends smoothly on $\mathbf{U}$ and that has a well-defined boundary. If $F = \int_{\Omega(\mathbf{U})} f(\mathbf{y})\,\mathrm{d}\mathbf{y}$, then

$$\frac{\mathrm{d}F}{\mathrm{d}\mathbf{U}} = \int_{\partial\Omega(\mathbf{U})} f(\mathbf{y})\dot{\mathbf{y}} \cdot \mathbf{n}\,\mathrm{d}\mathbf{y}$$

where $\mathbf{n}$ is the unit outward normal and $\dot{\mathbf{y}}$ denotes the derivative of the boundary points with respect to changes in $\mathbf{U}$.

Since

$$\mathbf{T}_i(\mathbf{z}) = \frac{\int_{V_i(\mathbf{z})} \mathbf{y}\rho(\mathbf{y})\,\mathrm{d}\mathbf{y}}{\int_{V_i(\mathbf{z})} \rho(\mathbf{y})\,\mathrm{d}\mathbf{y}}$$

we have that

$$\frac{\partial \mathbf{T}_i^{(m)}}{\partial \mathbf{z}_j^{(n)}} = \left(\int_{\partial V_i} \rho(\mathbf{y})\mathbf{y}^{(m)} \frac{\partial \mathbf{y}}{\partial \mathbf{z}_j^{(n)}} \cdot \mathbf{n}\,\mathrm{d}\mathbf{y}\right) \Bigg/ \int_{V_i(\mathbf{z})} \rho(\mathbf{y})\,\mathrm{d}\mathbf{y}$$
$$- \left(\int_{\partial V_i} \rho(\mathbf{y}) \frac{\partial \mathbf{y}}{\partial \mathbf{z}_j^{(n)}} \cdot \mathbf{n}\,\mathrm{d}\mathbf{y}\right) \int_{V_i(\mathbf{z})} \rho(\mathbf{y})\mathbf{y}^{(m)}\,\mathrm{d}\mathbf{y} \Bigg/ \left(\int_{V_i(\mathbf{z})} \rho(\mathbf{y})\,\mathrm{d}\mathbf{y}\right)^2$$

Here $1 \leqslant m \leqslant N$ and $1 \leqslant n \leqslant N$. An analytic representation of $\partial \mathbf{y}/\partial \mathbf{z}_j^{(n)}$ can be obtained from the following identity.

Lemma 2.2
If $\{\mathbf{u}_l\}$ are the vertices of the common face Δ_i^j between adjacent Voronoi regions generated by $\mathbf{z}_i$ and $\mathbf{z}_j$, then for any set of $\{\lambda_l\}, \lambda_l \geqslant 0$ with $\sum_{l\geqslant 0} \lambda_l = 1$, such that $\sum_{l\geqslant 0} \lambda_l \mathbf{u}_l \in \Delta_i^j$

$$\left(\sum_{l\geqslant 0} \lambda_l \mathbf{u}_l - \frac{\mathbf{z}_i + \mathbf{z}_j}{2}\right) \cdot (\mathbf{z}_j - \mathbf{z}_i) = 0$$

Differentiating the above expression with respect to $\mathbf{z}_i^{(n)}$, for any point $\mathbf{y} \in \Delta_i^j$ we get

$$\frac{\partial \mathbf{y}}{\partial \mathbf{z}_i^{(n)}} \cdot (\mathbf{z}_j - \mathbf{z}_i) = \frac{1}{2}\,\mathbf{e}_n \cdot (\mathbf{z}_j - \mathbf{z}_i) + \mathbf{e}_n \cdot \left(\mathbf{y} - \frac{\mathbf{z}_i + \mathbf{z}_j}{2}\right)$$

$$\frac{\partial \mathbf{y}}{\partial \mathbf{z}_j^{(n)}} \cdot (\mathbf{z}_j - \mathbf{z}_i) = \frac{1}{2}\,\mathbf{e}_n \cdot (\mathbf{z}_j - \mathbf{z}_i) - \mathbf{e}_n \cdot \left(\mathbf{y} - \frac{\mathbf{z}_i + \mathbf{z}_j}{2}\right)$$

where $\mathbf{e}_n = (0, \ldots, 1, \ldots, 0)^{\mathrm{T}}$ is a unit vector in $\mathbb{R}^N$. Since $\mathbf{n} = (\mathbf{z}_j - \mathbf{z}_i)/\|\mathbf{z}_j - \mathbf{z}_i\|$, the necessary expression for $(\partial \mathbf{y}/\partial \mathbf{z}_i^{(n)}) \cdot \mathbf{n}$ can be easily obtained and used for integration purposes.

2.2. Theoretical analysis

Classical convergence analysis of the Newton's method adopted in the current context relies on the following lemma (see for example Reference [21]).

Lemma 2.3
Suppose $\mathbf{F}(\mathbf{z}) = \mathbf{z} - \mathbf{T}(\mathbf{z}) : \mathbb{R}^{kN} \to \mathbb{R}^{kN}$ is continuously differentiable in an open convex set $D \subset \mathbb{R}^{kN}$. Assume that there exists a $\mathbf{z}^* \in \mathbb{R}^{kN}$, such that $\mathbf{F}(\mathbf{z}^*) = 0$ and there are constants $\beta, \gamma, r > 0$, such that

(1) $B(\mathbf{z}^*, r) \subset D$ is an open ball of radius r around $\mathbf{z}^*$;
(2) $(\mathbf{I} - \mathbf{dT}(\mathbf{z}^*))^{-1}$ exists and $\|(\mathbf{I} - \mathbf{dT}(\mathbf{z}^*))^{-1}\| < \beta$;
(3) $(\mathbf{I} - \mathbf{dT}) \in \mathrm{Lip}(\gamma, B(\mathbf{z}^*, r))$.

Then there is a radius $\varepsilon = \min\{r, 1/2\beta\gamma\}$, such that for any $\mathbf{z}_0 \in B(\mathbf{z}^*, \varepsilon)$, the sequence generated by the Newton's iteration $\mathbf{z}_n = \mathbf{z}_{n-1} - (I - \mathbf{dT})^{-1}\mathbf{F}(\mathbf{z}_{n-1})$ converges to $\mathbf{z}^*$ and obeys $\|\mathbf{z}_n - \mathbf{z}^*\| \leqslant \beta\gamma\|\mathbf{z}_{n-1} - \mathbf{z}^*\|^2$.

It is hard in general to produce criteria for the global convergence of the Newton's method. Here we discuss some of the results that help to further characterize the convergence radius of the Newton scheme in quantization context.

First let us denote $h_i(\mathbf{z}) = \mathrm{diam}(V_i)$ for each Voronoi cell V_i corresponding to the generators $\mathbf{z}$, and let D be a compact and convex set in the neighbourhood of a solution $\mathbf{z}^*$, such that $\mathbf{dT}$ is continuous in D and there are uniform bounds

$$H = \max_{\mathbf{z} \in D} h_i(\mathbf{z}), \quad h = \min_{\mathbf{z} \in D} h_i(\mathbf{z})$$

for all $1 \leqslant i \leqslant k$. Moreover, let

$$M = \max_{x \in \Omega} \rho(x), \quad m = \min_{x \in \Omega} \rho(x) \quad \text{and} \quad M' = \max_{x \in \Omega} |\nabla \rho(x)|$$

With these notations, we can claim the following Lipschitz continuity result for the Jacobian.

Lemma 2.4
In the 1-dimensional case, $I - \mathbf{dT} \in \mathrm{Lip}(\gamma, D)$, with $\gamma = 18M^2M'H^4/m^4h^4$.

Proof
The following relation was given in Reference [2]:

$$R_i^2(\mathbf{y})\left(1 - \sum_j \frac{\partial T_i}{\partial y_j}\right) = \frac{1}{2}\int_{V_i(\mathbf{y})}\int_{V_i(\mathbf{y})} (\rho(t)\rho'(s) - \rho(s)\rho'(t))(t - s)\,\mathrm{d}t\,\mathrm{d}s \tag{3}$$

where $R_i(\mathbf{y}) = \int_{V_i(\mathbf{y})} \rho(s)\,\mathrm{d}s$ and $V_i(\mathbf{y})$ denotes the ith Voronoi cell in the Voronoi tessellation generated by the generators $\mathbf{y} = (y_1, \ldots, y_k)$. So for any two sets of generators $\mathbf{x} = (x_1, \ldots, x_k)$ and $\mathbf{y} = (y_1, \ldots, y_k)$, we have

$$\left|\sum_j \left(\frac{\partial T_i}{\partial x_j} - \frac{\partial T_i}{\partial y_j}\right)\right| = \left|\sum_j \left(1 - \frac{\partial T_i}{\partial y_j} - \left(1 - \frac{\partial T_i}{\partial x_j}\right)\right)\right| = \frac{|R_i^2(\mathbf{x})Q_i(\mathbf{y}) - R_i^2(\mathbf{y})Q_i(\mathbf{x})|}{2R_i^2(\mathbf{x})R_i^2(\mathbf{y})}$$

where $Q_i(\mathbf{y}) = \int\int_{V_i(\mathbf{y})\times V_i(\mathbf{y})} \rho'(s)\rho(t)(t-s)\,\mathrm{d}t\,\mathrm{d}s$. Hence

$$\left|\sum_j \left(\frac{\partial T_i}{\partial x_j} - \frac{\partial T_i}{\partial y_j}\right)\right|$$

$$= \frac{|(R_i^2(\mathbf{x}) + R_i^2(\mathbf{y}))(Q_i(\mathbf{x}) - Q_i(\mathbf{y})) - (R_i^2(\mathbf{x}) - R_i^2(\mathbf{y}))(Q_i(\mathbf{x}) + Q_i(\mathbf{y}))|}{4R_i^2(\mathbf{x})R_i^2(\mathbf{y})}$$

$$\leqslant \frac{(R_i^2(\mathbf{x}) + R_i^2(\mathbf{y}))|Q_i(\mathbf{x}) - Q_i(\mathbf{y})| + (Q_i(\mathbf{x}) + Q_i(\mathbf{y}))|R_i^2(\mathbf{x}) - R_i^2(\mathbf{y})|}{4R_i^2(\mathbf{x})R_i^2(\mathbf{y})}$$

Let $V_i(\mathbf{x}-\mathbf{y}) = (V_i(\mathbf{x})\backslash V_i(\mathbf{y})) \cup (V_i(\mathbf{y})\backslash V_i(\mathbf{x}))$. Notice that there exists a constant such that $|V_i(\mathbf{x}-\mathbf{y})| \leqslant c\|\mathbf{x}-\mathbf{y}\|$. For the one-dimensional case, we can simply take $c=2$. We then have the following upper bounds:

$$Q_i(\mathbf{x}) \leqslant MM'H^3, \quad R_i(\mathbf{x}) \leqslant MH \text{ and}$$

$$|Q_i(\mathbf{x}) - Q_i(\mathbf{y})| \leqslant 2^{N+1}MM'H^{N+1}|V_i(\mathbf{x}-\mathbf{y})| \leqslant 2^{N+1}cMM'H^{N+1}\|\mathbf{x}-\mathbf{y}\|$$

$$|R_i^2(\mathbf{x}) - R_i^2(\mathbf{y})| = |R_i(\mathbf{x}) - R_i(\mathbf{y})|\,(R_i(\mathbf{x}) + R_i(\mathbf{y})) \leqslant 2cM^2H\|\mathbf{x}-\mathbf{y}\|$$

where N is the space dimension of the domain. Hence we end up with the following Lipschitz condition for $\mathbf{dT}$:

$$\|\mathbf{dT}(\mathbf{x}) - \mathbf{dT}(\mathbf{y})\| \leqslant \gamma\|\mathbf{x}-\mathbf{y}\|$$

where, for the 1-d case, by keeping track of the constants, we have $\gamma = 18M^2M'H^4/m^4h^4$. □

Proposition 2.1
For the computation of one-dimensional CVTs in the case of constant or log-concave densities, the Newton's method is quadratically convergent in a subset of D where $\|\mathbf{z}-\mathbf{z}^*\| < \theta/2\gamma$, with $(\mathbf{I}-\mathbf{dT}) \in \mathrm{Lip}(\gamma, D)$ and $\theta = 1 - \max_D \|\mathbf{dT}\| > 0$.

Proof
It was shown in Reference [16] that Lloyd's map is continuous in the neighbourhood D for any smooth density in 1-d. Notice also, that in the regions where the Lloyd's map is continuous, it is, in fact, continuously differentiable, so it remains to estimate the constants β and γ in this region. As shown in Reference [16], for constant and log-concave densities we have $\theta = 1 - \max_D \|\mathbf{dT}\| > 0$. Notice that $\|\mathbf{dT}\| < 1$ implies that $\mathbf{I}-\mathbf{dT}$ is invertible with $\|(\mathbf{I}-\mathbf{dT})^{-1}\| \leqslant \beta$ with $\beta = 1/(1-\|\mathbf{dT}\|) \leqslant 1/\theta$ (see Reference [22]). The conclusion then follows from Lemmas 2.3 and 2.4. □

With these ideas in mind, let us now design a new algorithm to accelerate the convergence of the Lloyd iteration. Knowing the issues associated with both Lloyd and Newton approaches, we will try to incorporate their best features into a coupled Lloyd–Newton scheme, as

described next. Note that round off and integration errors can affect theoretically established quadratic convergence of the Newton method. However, as we show later in numerical examples, with the acceleration scheme we are about to describe it is possible to preserve superlinear rate of convergence even in the case of non-linear densities.

2.3. The Lloyd–Newton acceleration scheme

The results mentioned above do not provide means of identifying the actual region of convergence for an arbitrary density function. In order to use the Newton's approach to speed up the fixed point Lloyd's iteration, we can deal with this problem by coupling the two algorithms into one hybrid scheme. For example, let us look at the following implementation of this idea.

Algorithm 2.1. Lloyd–Newton iteration

Input:
Ω, *the domain of interest*; ρ, *a probability distribution on* Ω;
k, *number of generators*; $\mathbf{z}=\{z_i\}_1^k$, *the initial set of generators*; ε-*tolerance.*
Output:
$\{V_i\}_1^k$, *a CVT with* k *generators* $\{z_i\}_1^k$ *in* Ω
Method:
1. Construct the Voronoi tessellation $\{V_i\}_1^k$ *of* Ω *with generators* $\mathbf{z}=\{z_i\}_1^k$.
2. Compute the mass centroids $\mathbf{x}=\{x_i\}_1^k$ *of* $\{V_i\}_1^k$.
3. If $\|\mathscr{H}(\mathbf{x},V_i)-\mathscr{H}(\mathbf{z},V_i)\|\geqslant\varepsilon$, *take* $\mathbf{z}=\mathbf{x}$ *and goto step 1.*
Otherwise fix $\alpha=1$, *let* $\mathbf{T}(\mathbf{x})=\mathbf{z}$ *and goto step 4.*
4. Perform a step of Newton's method: $\tilde{\mathbf{z}}=\mathbf{z}+\alpha(\mathbf{dT}|_{\mathbf{z}}-\mathbf{I})^{-1}(\mathbf{z}-\mathbf{T}(\mathbf{z}))$.
5. Let $I=\{i|1\leqslant i\leqslant k,\ \tilde{z}_i\notin\Omega\}$.
If $|I|=0$ *take* $\mathbf{z}=\tilde{\mathbf{z}}$ *and goto step 4.*
If $|I|=1$ *reduce Newton's step size*: $\alpha=\alpha/2$, *goto step 4.*
Otherwise take $\mathbf{z}$ *as generators and goto step 1.*
6. Repeat until some stopping criterion is met.

This hybrid method can be used to accelerate the Lloyd's scheme. As shown in Section 2.2, Newton iteration gives superlinear convergence, whenever the convergence region is reached. However, there are possible difficulties associated with this approach. Namely, the starting initial iteration may be too far from the solution, and out of the reach of the superlinear convergence region; the accurate and efficient solution of the linear system may be affected by the increase of the condition number with a large number of generators. In the examples presented in Section 2.5 we address these issues and present the numerical results that justify the use of the Newton approach as a local accelerator of the Lloyd iteration.

Let us note that in general, there is no guarantee for the Lloyd's method to converge to the global minimizer of the energy (see Reference [16] for a more recent study). The methods discussed above do not provide the means of reaching the global minimizer, but instead, they provide the acceleration of convergence in the local vicinity of any solution. It is possible, nevertheless, to couple these fast converging schemes with some global minimization methods to achieve the optimal performance. We will return to this discussion in later sections.

2.4. Computational complexity

Let us now briefly look at the complexity of the proposed algorithm. Each step of the adaptive Lloyd–Newton algorithm includes:

1. *Construction of Voronoi diagram*: For two dimensions, we use an embedded Matlab routine, which is of order $O(k)$ [23, 24]. In N-dimensions, the average complexity estimate of $O(k)$ with the running time of $O(k\log(k))$ is expected when the average number of Voronoi neighbours is bounded [25].
2. *Calculation of centroids for each region*: This involves calculation of two integrals per region, hence a total of $2k$ integrals, again $O(k)$.
3. *Calculation of the Jacobi matrix*: Each element of the matrix involves four new integrations, assuming we store the results of the previous centroidal calculations. There are k elements. Assume each one has on average m neighbours, $m<k$. Then we need a total of $4mk$ integrations. If we are sufficiently close to the optimal configuration, m does not exceed 8 (see Reference [26]), which makes this step worth $O(k)$.
4. *Solving the resulting linear system*: The complexity of the linear solver highly depends on the structure of the matrix. With a sparse banded matrix structure due to limited number of neighbours for each generator we can adopt fast inversion algorithms that minimize the fill in of the LU decomposition, for instance, the nested dissection methods for problems with bandwidth $\sqrt{k}$, that has complexity on the order of $O(k^{3/2})$ (see Reference [27]). Iterative methods with lower complexity can also be considered, we comment on this in the later discussions.
5. *Updating procedure for generators*: This is a simple element-by-element addition, requiring $2k$ operations.

Overall, it is clear that the total complexity depends critically on the linear solver and the algorithm for the construction of the Voronoi tessellations. For well-distributed points, however, it is reasonable to expect an optimistic linear time complexity.

2.5. Numerical results

We present the results of numerical computations on the square $\Omega=[0,1]^2$ for different types of density functions. All computations were made in Matlab 6.5. We used embedded Matlab functions *voronoi* and *voronoin* for diagram construction.

For the Lloyd's method, once the Voronoi construction is available, the only computational task left is to find the mass centroids of the Voronoi regions. For Newton's method, we also need to compute the entries of the Jacobian matrix, which adds up to the complexity of the problem. Computational properties of the problem heavily depend on the form of the density function used. One always needs to find a compromise between the accuracy and resource consumption for a particular algorithm. Since complexity of the quadrature is tightly bound with the computational cost of the algorithm, quadrature rules have to be tuned up depending on the form of the density function.

In the 1-d case, integrals can be computed more easily. In two-dimensional case, the Voronoi regions are of polygonal shape, so one may use triangle based integration rules or tensor-product based one-dimensional rules. This can be done using a triangulation of any kind.

We tested different integration rules for various types of density functions. For boundary integrals, we used Simpson's rule for polynomial densities of degree less than 3 and Gaussian

quadrature rules otherwise. For area integrals, midpoint Δ rule is used for all densities which refers to the triangular based quadrature rule: $\int_\Delta f = \frac{1}{3}|\Delta| \cdot (f(x_{12}) + f(x_{13}) + f(x_{23}))$, where x_{12}, x_{23}, and x_{13} are the midpoints of the sides of the triangle Δ. This rule is exact for polynomials of degree no greater than 2.

There are several criteria one can adopt for this algorithm: $\|\mathbf{z}_n - \mathbf{z}_{n-1}\| < \varepsilon$, i.e. when the distance between two consecutive configurations becomes small enough; $\|E_n - E_{n-1}\| < \varepsilon$, i.e. when changes in energy become sufficiently small or $N\text{step} > \text{MaxNumSteps}$, i.e. maximum number of steps is reached. It is also possible to base the stopping criterion on the relative decrease of the norm for the Jacobian that is conveniently available in our formulation. For each practical application, the user has a freedom to choose the most suitable approach. Quite often it is a combination of the above conditions that makes a good stopping criterion. In our examples, the algorithm was stopped whenever a failure of either conditions was discovered with $\varepsilon = 10^{-8}$. In the examples given below, we used the asymptotic formula $r \approx \log e_{n+1} / \log e_n$ to compute the convergence rate, where $e_n = \|\mathbf{z}_n - \mathbf{z}^*\|$ is the error at the nth iteration. Since for non-linear densities the exact solution is often hard to determine, we used the ratio $\log|\mathbf{z}_{n+1} - \mathbf{z}_n| / \log|\mathbf{z}_n - \mathbf{z}_{n-1}|$ to approximate the asymptotic convergence rate in this case.

One-dimensional examples: For one-dimensional intervals, since finding the Voronoi regions is trivial, most of the computation is associated with finding centroids. Numerical errors for such tasks are negligible, so the algorithm converges in several steps.

In Figure 1 we plot the asymptotic convergence rate for both Newton (top) and Lloyd (bottom) iterations. It can be readily seen that the limit is 2 in the Newton case, which justifies the quadratic convergence. Lloyd's method converges at a linear rate.

Two-dimensional examples: In the two-dimensional case, the effect of the roundoff and numerical integration errors becomes more pronounced. In case of a constant density, we are still able to get almost flawless performance. Figure 2 shows convergence of both methods for a random 5 generator configuration. Here dots denote positions of the generators at each step of the iteration and lines are used to separate the corresponding Voronoi regions. Lloyd–Newton iteration converged after 7 Newton steps, and convergence became quadratic as soon as the convergence region was reached, as shown in Figure 3.

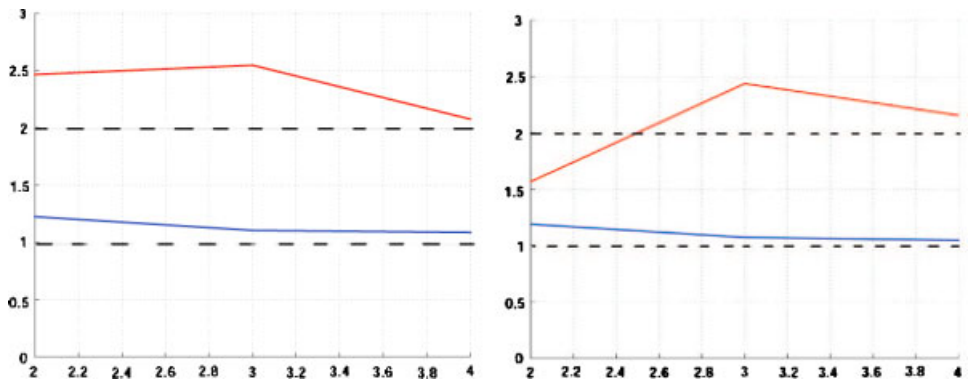

Figure 1. 1-d convergence rates comparison for $k = 4$ (left) and $k = 64$ (right) with $\rho(x) = 1 + x^4 \cos(\pi(x - 0.5))$. Top curves are for Newton iteration and the bottom ones are for Lloyd.

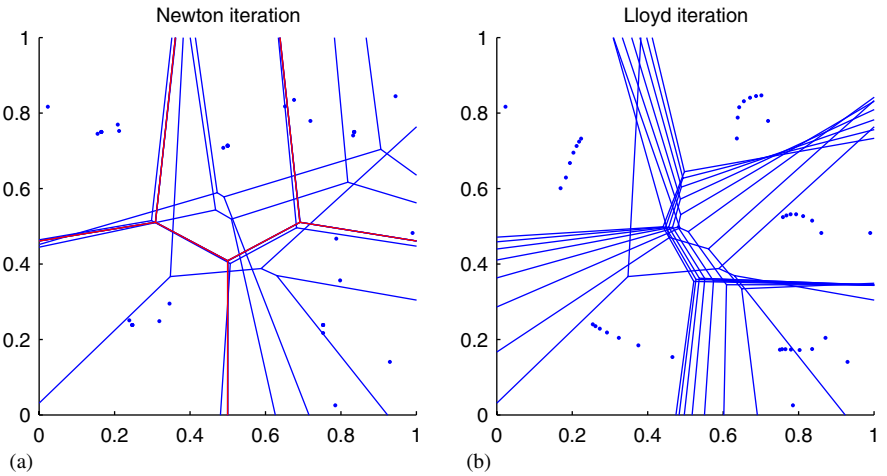

Figure 2. Iteration history of (a) Lloyd–Newton vs (b) Lloyd method for $\rho(x)=1, k=5$.

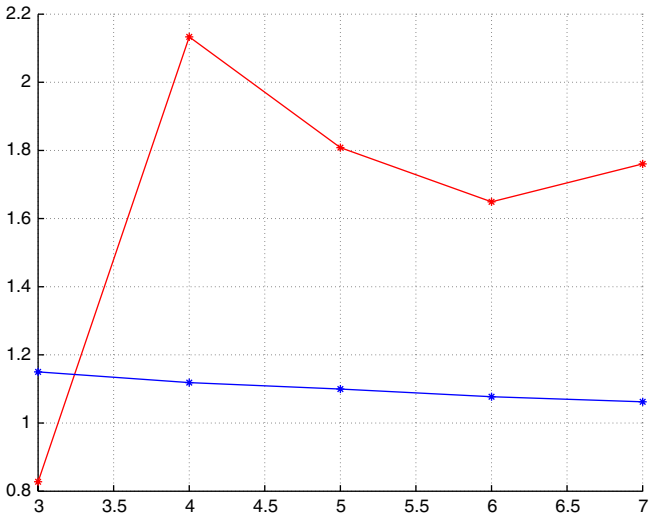

Figure 3. Convergence rate of Lloyd–Newton (top graph) vs Lloyd iteration (bottom graph) in $[0,1]^2$ for $\rho(x)=1, k=5$.

The next two pictures (Figures 4 and 5) demonstrate the performance of both methods in non-constant density cases. The Lloyd–Newton method converged after 6 Newton steps for $\rho(x)=1+x+0.1x^2$ and after 9 Newton steps for $\rho(x)=1+x^4$.

For a more precise comparison, Table I below shows the decrease of the error for Lloyd–Newton and Lloyd methods in the case of a quadratic density function $\rho(x)=1+x+0.1x^2$ after 5 consecutive iterations, respectively.

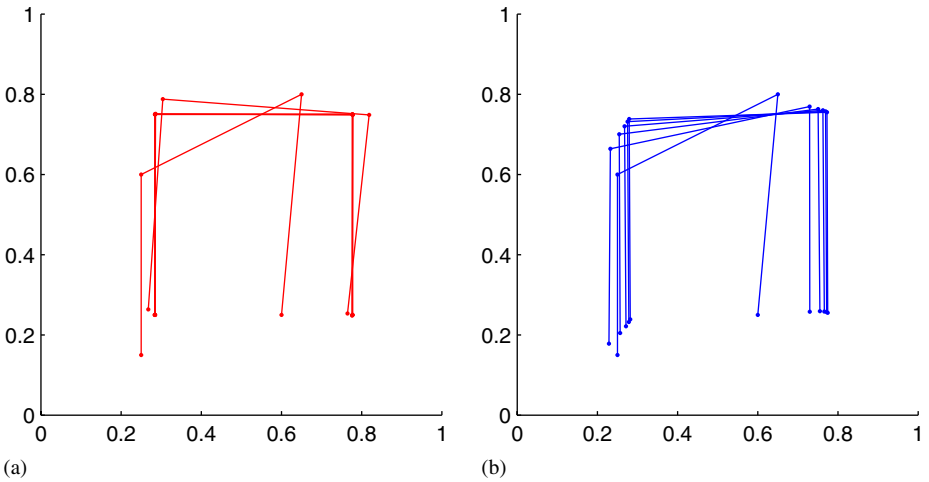

Figure 4. Iteration history of (a) Lloyd–Newton vs (b) Lloyd method for $\rho(x) = 1 + x + 0.1x^2$, $k = 4$. Here lines connecting the generators are drawn.

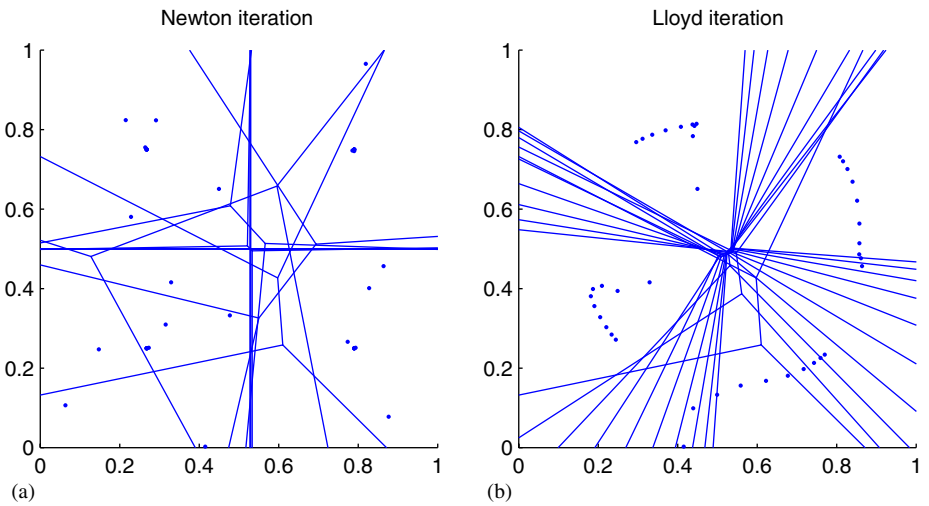

Figure 5. Iteration history of (a) Lloyd–Newton vs (b) Lloyd method, $\rho(x) = 1 + x^4, k = 4$.

Adding higher order terms to the density function introduces numerical error in the calculation of both boundary and area integrals. Here we compare the exact and inexact calculations made using Simpson's rule for line integrals and midpoint triangle rule for the area. Figure 6(a) shows results we got for a quadratic function, for which integration is exact, whereas the graph in Figure 6(b) shows convergence for a quartic polynomial density function.

Table I. Error reduction of the Lloyd–Newton and the Lloyd iterations.

Iteration	Lloyd's iteration error	Lloyd–Newton's iteration error
1	0.08641081909378	0.16571484289620
2	0.03313222925306	0.03144575914202
3	0.01849005608503	0.00159901251769
4	0.01041059669286	0.00000571605675
5	0.00599684938138	0.00000000572324

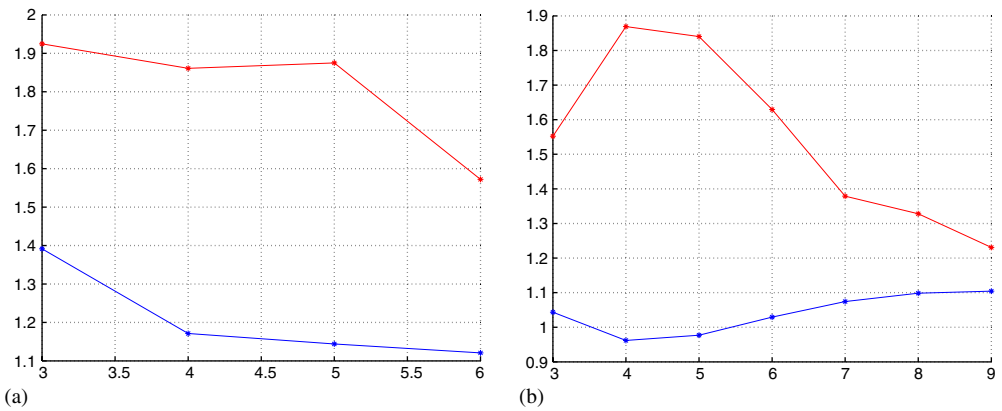

Figure 6. Comparison of convergence factors for Newton–Lloyd iteration (top) vs Lloyd (bottom) for different densities: (a) $\rho(x) = 1 + x + 0.1x^2$, $k = 4$; and (b) $\rho(x) = 1 + x^4$, $k = 4$.

Clearly, the integration errors do have an effect on the convergence of the overall scheme, so for the best performance of the algorithm the optimal tradeoff of the integration scheme accuracy and overall complexity should be made. As mentioned above, for the density functions up to certain order it is possible to nullify the numerical integration error by picking a more accurate quadrature rule. However, this might not be possible for a large class of functions, e.g. functions with singularities of a much higher order. Despite these natural restrictions, the results shown above clearly justify the fact that for an adequately chosen quadrature Lloyd–Newton method outperforms the Lloyd iteration and allows to reach the desired solution significantly faster.

3. NEWTON-BASED MULTILEVEL ALGORITHM

While the results presented in the previous section show that the Lloyd–Newton scheme generally performs better than the traditional fixed-point iteration, improvement can be made, in particular, by reducing the computational cost of solving the linear system for the Newton increment. In this regard, one possibility is to use multilevel techniques to solve the linear system inside the Newton iteration framework. We refer to this approach as the *inner* multigrid scheme. Naturally, the other possibility is to rely on the non-linear multigrid solver with the

Lloyd–Newton scheme being part of the inner relaxation procedure which would give an *outer* multigrid approach. We now first discuss the former case while leave the latter to the next section.

Recall that the non-linear problem under consideration is to find the fixed points of the Lloyd map $\mathbf{T}(\mathbf{z}) = \mathbf{z}$. As shown above, the Newton linearization

$$(\mathbf{I} - \mathbf{dT}|_{\mathbf{z}_{n-1}})(\mathbf{z}_n - \mathbf{z}_{n-1}) = \mathbf{T}(\mathbf{z}_{n-1}) - \mathbf{z}_{n-1}$$

gives a fast convergent iterative scheme in the neighbourhood of the solution. The performance can be greatly enhanced if some fast sparse solvers are used to reduce the computational complexity associated with the solution of linear systems. For instance, let us outline an algorithm that uses algebraic multigrid techniques for these purposes.

Algorithm 3.1. AMG–BGS–Newton iteration

Input:
Ω, *the domain of interest*; ρ, *a probability distribution on* Ω;
k, *number of generators*; $\mathbf{z} = \{z_i\}_1^k$, *the initial set of generators*;
Output:
$\{V_i\}_1^k$, *a CVT with* k *generators* $\mathbf{z} = \{z_i\}_1^k$ *in* Ω.
Method:
1. Given n-th iterate $\mathbf{z}_n$, *calculate* $\mathbf{T}(\mathbf{z}_n)$, $\mathbf{dT}(\mathbf{z}_n)$.
2. Put

$$A = \mathbf{I} - \mathbf{dT}(\mathbf{z}_n) = \begin{pmatrix} \mathbf{I} - \mathbf{T}_{xx} & \mathbf{T}_{xy} \\ \mathbf{T}_{yx} & \mathbf{I} - \mathbf{T}_{yy} \end{pmatrix}, \quad M = \begin{pmatrix} \mathbf{I} - \mathbf{T}_{xx} & 0 \\ \mathbf{T}_{yx} & \mathbf{I} - \mathbf{T}_{yy} \end{pmatrix}$$

and $b = \mathbf{T}(\mathbf{z}_n) - \mathbf{z}_n$.
3. Solve $M\mathbf{z}_{n+1} = b - (A - M)\mathbf{z}_n$, *where the system for each of the diagonal blocks involving* $(\mathbf{I} - \mathbf{T}_{xx})$ *and* $(\mathbf{I} - \mathbf{T}_{yy})$ *is solved using AMG.*
4. Repeat the procedure 1 to 3 until some stopping criterion is met.

Let us now discuss the key elements of the scheme introduced above. First key observation is related to the choice of a triangular iteration matrix

$$M = \begin{pmatrix} \mathbf{I} - \mathbf{T}_{xx} & 0 \\ \mathbf{T}_{yx} & \mathbf{I} - \mathbf{T}_{yy} \end{pmatrix}$$

for solving the linearized system. In making this choice, we relied on the fact that the matrix $A = \mathbf{I} - \mathbf{dT}$ has a block structure with the contribution of the off-diagonal blocks being relatively small. To solve the corresponding linear system, one can either perform the GMRES iteration with M being a preconditioner or resort to the block Gauss–Seidel (BGS) method taking M to be the corresponding iteration matrix.

The next key feature of this algorithm is the use of the algebraic multigrid method (AMG) [28–30] to solve the linear systems corresponding to each of the diagonal blocks of M. Indeed, such an approach is justified by the fact that both of the blocks $\mathbf{I} - \mathbf{T}_{xx}$ and $\mathbf{I} - \mathbf{T}_{yy}$ are

symmetric and often share diagonal dominance properties. An example of using the *classical* AMG approach based on the standard coarse-grid correction scheme is given as follows:

1. Perform relaxation of the fine grid until the error is smooth: $A^h u^h = b^h$.
2. Compute residual $r^h = b^h - A^h b^h$ and transfer to the coarse grid $r^{2h} = I_h^{2h} r^h$.
3. Solve the coarse-grid residual equation in terms of the error $A^{2h} e^{2h} = r^{2h}$.
4. Interpolate the error to the fine grid and correct the fine-grid solution: $u^h = u^h + I_{2h}^h e^{2h}$.

Here the restriction operator I_h^{2h} is dependent on the solution at the current iteration and represents a coarsening procedure, while the iteration dependent operator I_{2h}^h represents the standard interpolation. Naturally, a setup phase has to be implemented first based on the entries of A so that these operators are suitably defined [30]. Combining these considerations, we can design the AMG–BGS–Newton scheme, as shown in Algorithm 3.1.

The efficiency of such an algebraic multigrid implementation relies on the observation that each of the diagonal blocks of the M matrix become diagonally dominant in the vicinity of the solution. Theoretical arguments leading to this conclusion have been carried out in 1-d for the class of strongly logarithmically concave densities in Reference [16]. In fact, in this case the Lloyd map was shown to be a local contraction, implying diagonal dominance for the matrix $\mathbf{I} - \mathbf{dT}$. For these densities, a multilevel scheme designed this way outperforms regular Newton iteration in its convergence.

Another possible approach, as mentioned above, consists of taking a Newton iteration as part of the relaxation within the *outer* framework provided by some type of non-linear multigrid procedure. The next section is dedicated to a possible implementation of this type of algorithm.

4. OPTIMIZATION-BASED NON-LINEAR MULTILEVEL ALGORITHM

Since the original concept of centroidal Voronoi tessellations is related to the solution of a non-linear optimization problem, and the monotone energy descent property is preserved by the Lloyd's fixed point iteration [2], we may thus investigate whether monotone energy reduction can be achieved in a multilevel procedure which would also improve the performance of the simple-minded fixed point iteration.

The problem of constructing a CVT is non-linear in nature, hence standard linear multigrid theory cannot be directly applied. There are still several ways one could implement a non-linear multilevel scheme in this context (see References [19, 20, 31, 32]). The Newton type acceleration methods described earlier are based on some global linearization as the outer loop, coupled with other fast solvers in the inner loop. Alternatively, we now study an approach that overcomes the difficulties of the non-linearity by essentially relying on the direct energy minimization without any type of global linearization.

We note that the optimality property implies that at the CVT (or optimal quantizer), we have $\nabla \mathscr{H} = 0$. This is the key characterization to be used in the later discussion.

4.1. Space decomposition

Since the energy functional is in general non-convex, it turns out to be very effective to relate our problem to a convex optimization problem through a technique that mimics the role of a dynamic non-linear preconditioner. More precisely, denote $R = \mathtt{diag}\{R_i^{-1}\}, i = 1, \ldots, k+1$

where $R_i = \int_{V_i} \rho(\mathbf{y})\,\mathrm{d}\mathbf{y}$ are the masses of the corresponding Voronoi cells. We arrive at an equivalent formulation of the minimization problem: $R\nabla\mathscr{H} = 0$, or $\min \|R\nabla\mathscr{H}\|^2$. A key observation is that as R varies with respect to the generators, the above transformation or *dynamic preconditioning* makes the modified energy functional convex in a large neighbourhood of the minimizer and therefore makes the new formulation more amenable than the original problem. Hence, let us define the set of iteration points $\mathbf{W}$ by

$$\mathbf{W} = \{(w_i)|_{i=0}^{k+1} \mid 0 = w_0 \leqslant w_i \leqslant w_{i+1} \leqslant w_{k+1} = 1,\ \forall 0 \leqslant i \leqslant k\}$$

and let us design a new multilevel algorithm based on the following non-linear optimization problem:

$$\min_{\mathbf{z}\in\mathbf{W}} \tilde{\mathscr{H}}(\mathbf{z}), \quad \text{where } \tilde{\mathscr{H}}(\mathbf{z} = \{\mathbf{z}_i\}_{i=0}^{k+1}) = \|R\nabla\mathscr{H}(\{\mathbf{z}_i\}_{i=1}^{k}, \{V_i\}_{i=1}^{k})\|^2 \tag{4}$$

Here $\{V_i\}_{i=1}^{k}$ is the Voronoi tessellation corresponding to the generators $\{\mathbf{z}_i\}_{i=1}^{k}$. Let us take $\mathscr{T} = \mathscr{T}_J$ as a finite element mesh corresponding to $\mathbf{W}$. Consider a sequence of nested quasi-uniform finite element meshes $\mathscr{T}_1 \subset \mathscr{T}_2 \subset \cdots \mathscr{T}_J$, where $\mathscr{T}_i$ consists of all finite element meshes $\{\tau_j^i\}_{j=1}^{n_i}$ with mesh parameter h_i, such that $\bigcup_{j=1}^{n_i} \tau_j^i = \Omega$. Corresponding to each finite element partition $\mathscr{T}_i$ there is a finite element space $\mathbf{W}_i$ defined by

$$\mathbf{W}_i = \{v \in H^1(\Omega) \mid v|_\tau \in \mathscr{P}_1(\tau),\ \forall \tau \in \mathscr{T}_i\}$$

For each $\mathbf{W}_i$ there corresponds a nodal basis $\{\psi_j^i\}_{j=1}^{n_i}$, such that $\psi_j^i(x_k^i) = \delta_{jk}$, where $\{x_k^i\}_{k=1}^{n_i}$ is the set of all nodes of the elements of $\mathscr{T}_i$ and $x_1^J = 0, x_{n_J}^J = 1$. Define the corresponding one-dimensional subspaces $\mathbf{W}_{i,j} = \mathrm{span}\{\psi_j^i\}$. Then the decomposition can be regarded as

$$\mathbf{W}_J = \sum_{i=1}^{J} \sum_{j=1}^{n_i} \mathbf{W}_{i,j} = \bigoplus_{i=1}^{J} \bar{\mathbf{W}}_i$$

where $\bar{\mathbf{W}}_i = \mathbf{W}_i / \mathbf{W}_{i-1}$ for $i > 1$ and $\bar{\mathbf{W}}_1 = \mathbf{W}_1$. Now clearly for every function $\psi_j^i \in \mathbf{W}_i$ we can find a vector $\bar{\psi}_j^i = \{\bar{\psi}_{jm}^i\} \in \mathbb{R}^{n_J}$, such that $\psi_j^i(x) = \sum_{m=1}^{n_J} \bar{\psi}_{jm}^i \psi_m^J(x),\ \forall x \in \Omega$.

We note that in the 1-dimensional case, the set of basis functions

$$Q_i = [\bar{\psi}_1^i, \ldots, \bar{\psi}_{n_i}^i]^{\mathrm{T}} \in \mathbb{R}^{n_i \times k}$$

used at each iteration can be pre-generated using the recursive procedure: $Q_J = I_{k\times k}$ and $Q_{J-s} = (\prod_{i=1}^{s} P_{J-i}) Q_J$ where P_i is the basis transformation from space $\mathbf{W}_{i+1}$ to $\mathbf{W}_i$ which plays a role of a restriction operator.

4.2. Description of the algorithm

Using the above notations, we design a multilevel successive subspace correction algorithm (Algorithm 4.1). Each step of the procedure outlined below involves solving a system of non-linear equations which plays the role of relaxation. We use the Newton iteration to solve this non-linear system, similarly to the method described in Section 2. Solution at current iterate is updated after each non-linear solve by the Gauss–Seidel type procedure, hence the resulting scheme is successive in nature. The algorithm uses a procedure *CoarseGridSolve*($\mathbf{Z}$), which, as the names indicates, refers to finding the solution at the coarsest level. In our

implementation, this procedure consists of applying Lloyd method for a few steps or until saturation. In general, other efficient optimization methods, as well as Newton's method, can be used in order to quickly damp the error, since the number of unknowns on the coarsest grid remains relatively small. The overall algorithm essentially only depends on the proper space decompositions and the correspondence with the set of generators thus is applicable in any dimension. The more general forms will be discussed in our subsequent works.

Algorithm 4.1. Successive correction $V(\nu_1, \nu_2)$ scheme

Input:
Ω, *the domain of interest*; ρ, *a probability distribution on* Ω;
k, *number of generators*;
$\mathbf{z} = \{z_i\}_{i=0}^{k+1} \in \mathbf{W}$, *the ends plus the initial set of generators.*
Output:
$\mathbf{z} = \{z_i\}_{i=0}^{k+1}$, *the ends plus the set of generators for CVT* $\{V_i\}_{i=1}^{k}$.
Method:
1. For $i = J:-1:2$
Repeat ν_1 *times*: *given* $\mathbf{z}$, *find* $\mathbf{z} = \mathbf{z} + \alpha_j^0 \bar{\psi}_j^i \in \mathbf{W}$ *sequentially for* $1 \leqslant j \leqslant n_i$
such that $\tilde{\mathscr{H}}(\mathbf{z} + \alpha_j^0 \bar{\psi}_j^i) = \min_{\alpha_j} \tilde{\mathscr{H}}(\mathbf{z} + \alpha_j \bar{\psi}_j^i)$,
endfor
2. At the coarsest level, update $\mathbf{z}$ *by* $\mathbf{z} \leftarrow$ *CoarseGridSolve*$(\mathbf{z})$.
3. For $i = 2:1:J$
Repeat ν_2 *times*: *given* $\mathbf{z}$, *find* $\mathbf{z} = \mathbf{z} + \alpha_j^0 \bar{\psi}_j^i \in \mathbf{W}$ *sequentially for* $1 \leqslant j \leqslant n_i$,
such that $\tilde{\mathscr{H}}(\mathbf{z} + \alpha_j^0 \bar{\psi}_j^i) = \min_{\alpha_j} \tilde{\mathscr{H}}(\mathbf{z} + \alpha_j \bar{\psi}_j^i)$,
endfor
4. Repeat the procedure 1 to 3 until some stopping criterion is met.

First, for $y = u - v$ where $u, v \in \mathbf{W}$, we supply with the following norm:

$$\|y\|_{1,\mathbf{W}}^2 = \frac{1}{k} \sum_{i=1}^{k+1} (y_i - y_{i-1})^2$$

Note that $y_0 = y_{k+1} = 0$.

We can state the following convergence result.

Theorem 4.1
Algorithm 3.1 converges uniformly in $\mathbf{W}$ for any density of the type $\rho(x) = 1 + \varepsilon g(x)$, where $g(x)$ is smooth and ε is small. Moreover, $d_n = \tilde{\mathscr{H}}(u_n) - \tilde{\mathscr{H}}(u)$ satisfies

$$d_n \leqslant r d_{n-1}, \quad r \in (0, 1)$$

for some constant $r = C/(1 + C)$, where $C = C_1^2 C_2^2 L / K^3$, independent of the number of generators or the number of layers.

The proof of the above result can be constructed following the framework of Reference [33] and is rather technical, so it appears in a separate work [19], while extensions of this proof

to higher dimensions are discussed in Reference [20]. Roughly, the key steps of the proof include demonstrating that for all densities of the given type there exist constants $K>0, L>0$ such that

$$K\|w-v\|_{1,\mathbf{W}}^2 \leqslant (\tilde{\mathscr{H}}'(w)-\tilde{\mathscr{H}}'(v), w-v) \leqslant L\|w-v\|_{1,\mathbf{W}}^2, \quad \forall w, v \in \mathbf{W}$$

Moreover, the space decomposition (e.g. for the hierarchical basis) satisfies:

(1) for any $v \in \mathbf{W}$, there exist $v_i \in \bar{\mathbf{W}}_i$ such that

$$\sum_{i=1}^{J} v_i = v, \quad \left(\sum_{i=1}^{J} \|v_i\|_{1,\bar{\mathbf{W}}_i}^2\right)^{1/2} \leqslant C_1 \|v\|_{1,\mathbf{W}}$$

(2) for any $w_{ij} \in \mathbf{W}, u_i \in \bar{\mathbf{W}}_i, v_j \in \bar{\mathbf{W}}_j$, we have

$$\sum_{i,j=1}^{J} (\tilde{\mathscr{H}}'(w_{ij}+u_i)-\tilde{\mathscr{H}}'(w_{ij}), v_j) \leqslant C_2 \left(\sum_{i=1}^{J} \|u_i\|_{1,\bar{\mathbf{W}}_i}^2\right)^{1/2} \left(\sum_{j=1}^{J} \|v_j\|_{1,\bar{\mathbf{W}}_j}^2\right)^{1/2}$$

The complete proof is given in Reference [19] and is omitted here.

Corollary 4.2
For the constant density function in 1-d, we have $C=4$.

It follows that for a suitable choice of decomposition the asymptotic convergence factor of our multilevel algorithm is independent of the size of the problem and the number of grid levels, which gives a significant speedup compared to other methods, like the traditional Lloyd iteration. This claim can be justified by the following numerical examples, computed using the Matlab 6.5 implementation of the new algorithm on a Pentium IV with 512 MB RAM. We compare the results of our $V(1,0)$ multilevel implementation with the regular Lloyd method, and we also present some results for a two-dimensional test problem in a parallelogram domain.

The one-dimensional implementation is very straightforward. Here, we take the unit interval and test a couple of different density functions $\rho(x)=1$ and $\rho(x)=1+x$. For the energy functional $\tilde{\mathscr{H}}$ defined in (4), we plot the convergence factor $\rho \approx |\tilde{\mathscr{H}}(u_{n+1})-\tilde{\mathscr{H}}(u_n)|/|\tilde{\mathscr{H}}(u_n)-\tilde{\mathscr{H}}(u_{n-1})|$ for each $V(1,0)$ cycle with respect to the total number of generators (grid points) involved.

Figure 7 substantiates the fact that the speed of convergence for the proposed scheme remains nearly constant as the number of generators increases. The graph shows that for the 1-d examples, the computational time *scales almost linearly* with the problem size. More statistics on the implementation using other multilevel cycles can be found in Reference [19]. The geometric rate of the energy and error reduction asserted by Theorem 4.1 are also confirmed by the experiments. Indeed, Figure 8 shows the convergence history (that is, the error reduction during the iteration) of a $V(1,0)$-cycle against the total number of relaxations for the $k=129$ case.

Finally, the convergence factors for some two-dimensional problems on a parallelogram domain are compared in Figure 9. The graph on the left shows the convergence factor for

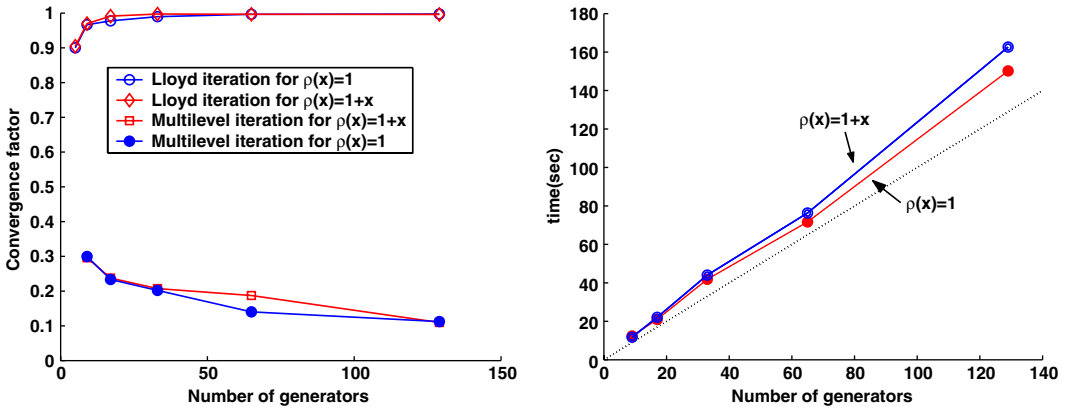

Figure 7. The left plot shows the convergence factor vs the number of generators for Lloyd (upper) and multilevel (lower curves) iterations with $\rho(x)=1$ and $\rho(x)=1+x$. The right plot shows the computational time needed for the $V(1,0)$ implementation of the multilevel method vs the problem size.

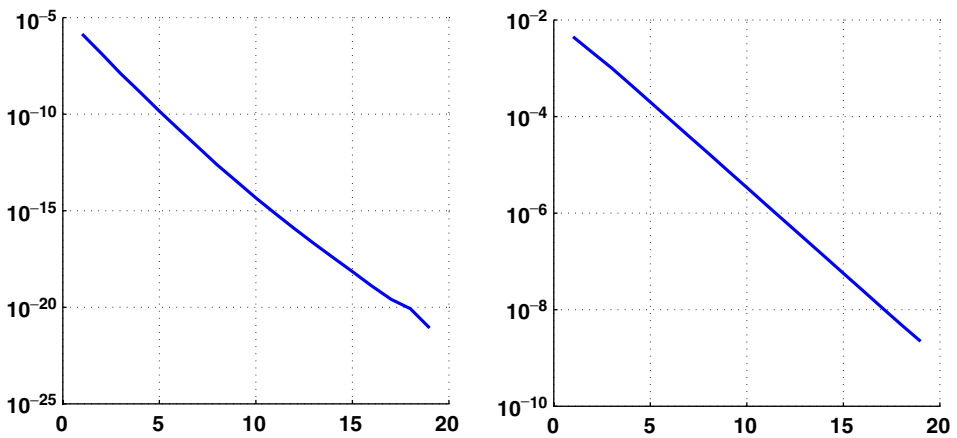

Figure 8. The energy reduction (left) and the convergence history (right) for 129 generators in the log-normal scale.

the compatible relaxation, that is, with the exact solution given at the coarse nodes (see Reference [34] for further discussion). The result gives an indication of the effectiveness of the coarsening procedure. On the right of Figure 9 is the convergence history plot for the 2-d example. The top curves there depicts the error reductions given by the Lloyd iteration, while the graphs below correspond to the convergence of the multigrid scheme for various problem sizes. We see that even though our theoretical results are only proved in 1-d here, it is clear that they remain valid in the higher dimensional implementations.

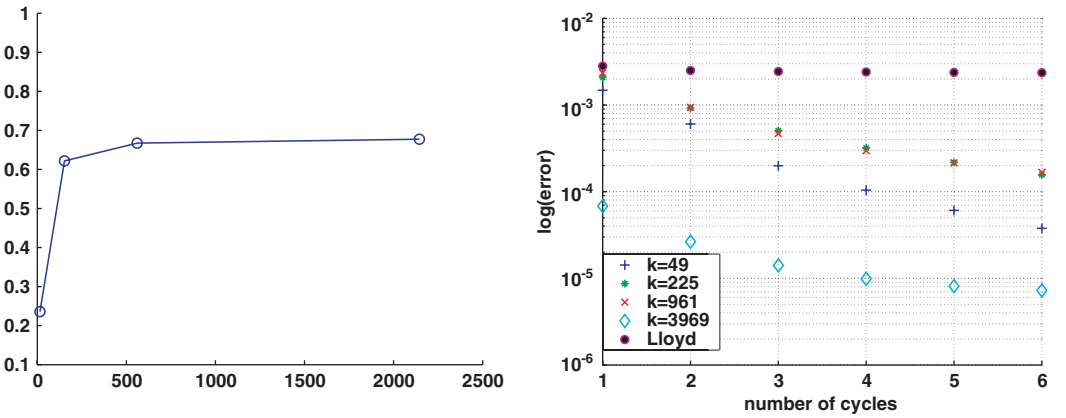

Figure 9. Left: Convergence factors of a compatible relaxation against the number of generators. Right: convergence history for the multigrid scheme compared to the Lloyd scheme (top curve).

5. CONCLUSION

In this paper, several methods are proposed for accelerating the convergence of the classical Lloyd iteration commonly used in the context of quantization and in the construction of centroidal Voronoi tessellations. The coupling of the Lloyd method with a Newton-like iteration is introduced and studied both analytically and numerically. Some possible extensions that use multilevel techniques to accelerate the convergence of the CVTs are suggested and their implementations demonstrate enhancement of both the robustness and the efficiency of the original algorithm. One of the extensions uses algebraic multigrid solver as a preconditioner to accelerate the solution of the linear system at every Newton iteration, while the other adopts a novel energy based non-linear multigrid approach with the use of a dynamic non-linear preconditioning. Some analysis of the convergence properties and numerical experiments for these methods are carried out, with the more in-depth studies of the new algorithms as well as various application problems left to the future works.

ACKNOWLEDGEMENTS

We thank the referees for their careful reading and valuable suggestions. This research was supported by the NSF grants DMR-0205232 and DMS-0409297.

REFERENCES

1. Gray R, Neuhoff D. Quantization. *IEEE Transactions on Information Theory* 1998; **44**:2325–2383.
2. Du Q, Faber V, Gunzburger M. Centroidal Voronoi tessellations: applications and algorithms. *SIAM Review* 1999; **41**:637–676.
3. Gersho A. Asymptotically optimal block quantization. *IEEE Transactions on Information Theory* 1979; **25**: 373–380.
4. Trushkin A. On the design of an optimal quantizer. *IEEE Transactions on Information Theory* 1993; **39**: 1180–1194.

5. Du Q, Gunzburger M. Grid generation and optimization based on centroidal Voronoi tessellations. *Applied Mathematics and Computation* 2002; **133**:591–607.
6. Ju L, Du Q, Gunzburger M. Probabilistic methods for centroidal Voronoi tessellations and their parallel implementations. *Parallel Computing* 2002; **28**:1477–1500.
7. Ju L, Gunzburger M, Du Q. Meshfree, probabilistic determination of points, support spheres, and connectivities for meshless computing. *Computational Methods in Applied Mechanics and Engineering* 2002; **191**:1349–1366.
8. Du Q, Gunzburger M, Ju L. Constrained centroidal Voronoi tessellations on general surfaces. *SIAM Journal on Scientific Computing* 2003; **24**:1488–1506.
9. Cappellari M, Copin Y. Adaptive spatial binning of integral-field spectroscopic data using Voronoi tessellations. *Monthly Notices of the Royal Astronomical Society* 2003; **2**:345–354.
10. Cortes J, Martinez S, Karata T, Bullo F. Coverage control for mobile sensing networks. *IEEE Transactions on Robotics and Automation* 2004; **20**:243–255.
11. Mendes A, Themido I. Multi-outlet retail site location assessment. *International Transactions in Operational Research* 2004; **11**:1–18.
12. Hiller S, Hellwig H, Deussen O. Beyond stippling methods for distributing objects on the plane. *Computer Graphics Forum* 2003; **22**:515–522.
13. Valette S, Chassery J. Approximated centroidal Voronoi diagrams for uniform polygonal mesh coarsening. *Computer Graphics Forum* 2004; **23**:381–390.
14. Wager C, Coull B, Lange N. Modelling spatial intensity for replicated inhomogeneous point patterns in brain imaging. *Journal of Royal Statistical Society B* 2004; **66**:429–446.
15. Lloyd S. Least square quantization in PCM. *IEEE Transactions on Information Theory* 1982; **28**:129–137.
16. Du Q, Emelianenko M, Ju L. Convergence properties of the Lloyd algorithm for computing the centroidal Voronoi tessellations. *SIAM Journal on Numerical Analysis* 2006, in press.
17. Linde Y, Buzo A, Gray R. An algorithms for vector quantizer design. *IEEE Transactions on Communications* 1980; **28**:84–95.
18. MacQueen J. Some methods for classification and analysis of multivariate observations. In *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, *I*, LeCam L, Neyman J (eds). 1967; 281–297.
19. Du Q, Emelianenko M. Uniform convergence of a nonlinear energy-based multilevel quantization scheme via centroidal Voronoi tessellations. 2006, preprint.
20. Du Q, Emelianenko M, Zikatanov L. An energy-based multilevel quantization scheme in multidimension. 2005, preprint.
21. Dennis J, Schnabel R. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Prentice-Hall: Englewood Cliffs, NJ, 1983; 86–110.
22. Golub G, Van Loan C. *Matrix Computations*. The John Hopkins University Press: Baltimore, MD, 1989.
23. Aurenhammer F. Voronoi diagrams. A survey of a fundamental geometric data structure. *ACM Computing Surveys* 1990; **23**:345–405.
24. Fortune S. A sweepline algorithm for Voronoi diagrams. *Algorithmica* 1987; **2**:153–174.
25. Dwyer R. Higher-dimensional Voronoi diagrams in linear expected time. *Discrete and Computational Geometry* 1991; **6**:343–367.
26. Okabe A, Boots B, Sugihara K. *Spatial Tessellations*; *Concepts and Applications of Voronoi Diagrams*. Wiley: Chichester, 1992.
27. George A. Nested dissection of a regular finite element mesh. *SIAM Journal on Numerical Analysis* 1973; **10**:345–363.
28. Brezina M, Cleary A, Falgout R, Henson H, Jones J, Manteuffel T, McCormick S, Ruge J. Algebraic multigrid based on element interpolation. *SIAM Journal on Scientific Computing* 2000; **22**:1570–1592.
29. Chang Q, Huang Z. Efficient algebraic multigrid algorithms and their convergence. *SIAM Journal on Scientific Computing* 2002; **24**:597–618.
30. Ruge J, Stuben K. *Algebraic Multigrid*. Multigrid Methods. Fronties in Applied Mathematics. SIAM: Philadelphia, PA, 1987; 73–130.
31. Koren Y, Yavneh I, Spira A. A multigrid approach to the 1-D quantization problem. *IEEE Transactions on Information Theory* 2005; **51**(8):2993–2998.
32. Koren Y, Yavneh I. Adaptive multiscale redistribution for vector quantization. *SIAM Journal on Scientific Computing* 2005, in press.
33. Tai X, Xu J. Global convergence of subspace correction methods for some convex optimization problems. *Mathematics of Computation* 2002; **71**:105–124.
34. Brandt A. General highly accurate algebraic coarsening. *Electronic Transactions on Numerical Analysis* 2000; **10**:1–20.

