TRUNCATED NEWTON-BASED MULTIGRID ALGORITHM FOR CENTROIDAL VORONOI DIAGRAM CALCULATION

ZICHAO DI*, MARIA EMELIANENKO†, AND STEPHEN G. NASH‡

Abstract. In a variety of modern applications there arises a need to tessellate the domain into representative regions, called Voronoi cells. A particular type of such tessellations, called centroidal Voronoi tessellations or CVTs, are in big demand due to their optimality properties important for many applications. The availability of fast and reliable algorithms for their construction is crucial for their successful use in practical settings. This paper introduces a new multigrid algorithm for constructing CVTs that is based on the MG/Opt algorithm that was originally designed to solve large nonlinear optimization problems. Uniform convergence of the new method and its speedup comparing to existing techniques are demonstrated for linear and nonlinear densities for several 1d and 2d problems, and $O(k)$ complexity estimation is provided for a problem with k generators.

Key words. Optimal quantization, centroidal Voronoi tessellations, Lloyd's algorithm, multilevel method, uniform convergence

1. Introduction. A Voronoi diagram can be thought of as a map from the set of N-dimensional vectors in the domain $\Omega \subset \mathbb{R}^N$ into a finite set of vectors $\{\mathbf{z}_i\}_{i=1}^k$ called generators. It associates with each $\mathbf{z}_i$ a nearest neighbor 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 into the Voronoi regions $\{V_i\}_{i=1}^k$ associated with a set of given generating points $\{\mathbf{z}_i\}_{i=1}^k \subset \Omega$ [**?**, **?**]. With a suitably defined distortion measure, an optimal tessellation is given by a centroidal Voronoi tessellation, which is constructed as follows. For a given density function ρ defined on Ω, we may define the centroids, or mass centers, of regions $\{V_i\}_{i=1}^k$ by

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

A *centroidal Voronoi tessellation* (CVT) is then a tessellation for which the generators of the Voronoi diagram coincide with the centroids of their respective Voronoi regions, in other words, $\mathbf{z}_i = \mathbf{z}_i^*$ for all i.

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

$$\mathcal{F}\Big(\{\mathbf{z}_i\}_{i=1}^k, \{V_i\}_{i=1}^k\Big) = \sum_{i=1}^k \int_{V_i} \rho(\mathbf{y})|\mathbf{y}-\mathbf{z}_i|^2\,d\mathbf{y}.$$

If the Voronoi tessellation $\{V_i\}_{i=1}^k$ is determined from $\{\mathbf{z}_i\}_{i=1}^k$ then we write

$$\mathcal{G}\Big(\{\mathbf{z}_i\}_{i=1}^k\Big) \equiv \mathcal{F}\Big(\{\mathbf{z}_i\}_{i=1}^k, \{V_i\}_{i=1}^k\Big). \tag{1.1}$$

The minimizer of $\mathcal{G}$ necessarily forms a CVT which illustrates the optimization property of the CVT [**?**]. This functional appears in many engineering applications and the

*Department of Mathematical Sciences, George Mason University, Fairfax, VA 22030. (zdi@gmu.edu)

†Department of Mathematical Sciences, George Mason University, Fairfax, VA 22030. (memelian@gmu.edu).

‡Systems Engineering and Operations Research Department, George Mason University, Fairfax, VA 22030. (snash@gmu.edu).

relation of its minimizers with CVTs is studied, for instance, in [**?**, **?**, **?**]. For instance, it provides optimal least-squares vector quantizer design in electrical engineering applications. 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 [**?**, **?**, **?**, **?**, **?**, **?**, **?**, **?**, **?**, **?**, **?**, **?**]. In [**?**, **?**], extensive reviews of the modern mathematical theory and diverse applications of CVTs are provided, and this list is constantly growing.

The most widely used method for computing CVTs is the algorithm developed by Lloyd in the 1960s [**?**]. Lloyd's algorithm represents a fixed-point type iterative algorithm consisting of the following simple steps: starting from an initial configuration (a Voronoi tessellation corresponding to an old set of generators), a new set of generators is defined by the mass centers of the Voronoi regions. The domain is re-tessellated and a new set of centroids is taken as generators. This process is continued until some stopping criterion is met. For other types of algorithms for computing CVTs we refer to [**?**, **?**, **?**, **?**]. It was shown that Lloyd's algorithm decreases the energy functional $\mathcal{G}(\{\mathbf{z}_i\}_{i=1}^k)$ at every iteration, which gives strong indications of its practical convergence. Despite its simplicity, proving convergence of Lloyd's algorithm is not a trivial task. Some recent work [**?**, **?**] has substantiated earlier claims about global convergence of Lloyd's algorithm, although single-point convergence for a general density function ρ is still not rigorously justified.

For modern applications of the CVT concept in large scale scientific and engineering problems such as data communication, vector quantization and mesh generation, it is crucial to have fast and memory-efficient algorithms for computing the CVTs. Variants of Lloyd's algorithm have been recently proposed and studied in many contexts for different applications [**?**, **?**, **?**, **?**]. A particular extension using parallel and probabilistic sampling was given in [**?**] which allows efficient and mesh free implementation of Lloyd's algorithm. However, the issue of finding a better alternative remains critical for many applications, since Lloyd's algorithm and its variants are at best linearly convergent. Moreover, the standard Lloyd algorithm slows down as the number of generators gets large, which renders many practical calculations prohibitively expensive. Several alternatives have been proposed, including Newton-based methods [**?**, **?**] and GPU extensions [**?**, **?**].

Multilevel algorithms present a desirable framework in the context of large-scale applications, since they do not suffer from the deterioration of convergence as the problem size grows—a typical pitfall of most iterative methods. The idea of multigrid and multilevel implementation of Lloyd's algorithm (referred to later as "Multilevel-Lloyd method") has been recently introduced in [**?**], where multigrid method was used as an "outer" scheme, with Lloyd's algorithm playing the role of a relaxation at each level. The method was rigorously shown to be uniformly convergent for all smooth perturbations of a constant density in [**?**] and was extended to the 2-dimensional setting in [**?**], where it was successfully applied to a physical data binning application problems. Despite its uniform convergence, the method has a rather big computational cost for medium-sized problems and the construction of the interpolation operators is far from straightforward. An alternative approach was introduced by Yavneh et al. in [**?**] that used a FAS (full approximation scheme) implementation of the Lloyd-Max scheme method based on minimizing the residual between generators and centroids. We refer to this algorithm as "Multigrid Lloyd-Max method" later in the text. Although successful in the 1-dimensional setting, this approach was not generalizable to higher dimensions [**?**].

Here we propose a new multilevel formulation for constructing CVTs in the 1-dimensional and 2-dimensional cases that is based on a different type of FAS scheme. Our approach is distinctive because the algorithm uses a fine-tuned version of the"off the shelf" techniques instead of special purpose approaches. A distinctive feature of the new method is its close relationship with optimization problems. This is accomplished by formulating Lloyd's algorithm as a minimization problem in terms of the energy functional $\mathcal{G}(\{\mathbf{z}_i\}_{i=1}^k)$. We then apply a multilevel optimization framework called MG/Opt to this functional.

MG/Opt is a general framework for optimization when a hierarchy of approximate models is available. MG/Opt was originally developed in the context of optimal control problems with PDE constraints, but is applicable to a broader class of problems. By choosing appropriate interpolation operators, we design a new scheme which enjoys uniform convergence with respect to the problem size similar to the aforementioned algorithms. The advantage comparing to the Multilevel-Lloyd method comes in a form of significant reduction of computational costs, while lower convergence factors and generalizability to higher dimensions favorably distinguishes it from the Multigrid Lloyd-Max method.

Although MG/Opt is inspired by FAS, it is not equivalent to FAS. In the simplest case, i.e., when MG/Opt is applied to an unconstrained problem, the update to the variables is the same as if FAS were applied to the first-order optimality conditions. However MG/Opt includes a line search which guarantees convergence in the sense that $\lim_{i\to\infty} \| \nabla\mathcal{G} \| = 0$. Even if a line search is added to FAS, then FAS is only guaranteed to find a local minimizer of $\| \nabla\mathcal{G} \|$, which is not guaranteed to be zero. Thus MG/Opt has stronger convergence properties than FAS. In addition, MG/Opt (but not FAS) can be applied to optimization problems with constraints, such as bounds on the variables or more complicated constraints [**?**]. We expect this to be an important factor when applying MG/Opt to two- and higher-dimensional problems. Additional reasons for preferring MG/Opt to FAS are discussed in [**?**].

The rest of the paper is organized as follows. In section 2, we discuss the MG/Opt algorithm which forms the basis for the new CVT construction method. Its connection with the CVT formulation is the subject of Section 3, where we also introduce the new FAS implementation and define the corresponding operators. The results of numerical experiments and comparison with other methods are presented in Section 4, and final conclusions and discussion are found in Section 5.

2. The Multilevel Optimization Algorithm. We are applying a multilevel optimization algorithm called MG/Opt to the CVT problem. MG/Opt was originally developed to solve unconstrained optimization problems [**?**], with an emphasis on discretized optimization models. It has been extended to constrained problems [**?**] and the approach can be applied to problems not based on discretization.

MG/Opt is designed to accelerate a traditional optimization algorithm applied to a high-fidelity problem by exploiting a hierarchy of coarser models. In its raw form it is an optimization framework, not an algorithm, since it depends on an underlying (traditional) optimization algorithm, here labeled "OPT". There is considerable flexibility in selecting the underlying optimization algorithm. Of course, the performance of MG/Opt will depend on the choice of OPT. MG/Opt is based on the ideas of the full approximation scheme (see, e.g., [**?**]) but is not equivalent to the full approximation scheme.

Before giving a description of MG/Opt, it is necessary to say more about the underlying optimization algorithm OPT. It is assumed that OPT is a convergent

optimization algorithm in the sense that, if appropriate assumptions on the objective function $\mathcal{G}$ are satisfied then

$$\lim_{j\to\infty} \| \nabla\mathcal{G}(\mathbf{z}_j) \| = 0$$

where $\{ \mathbf{z}_j \}$ are the iterates computed by OPT. (See, for example, [**?**].) We will write OPT as a function of the form

$$\mathbf{z}^+ \leftarrow \mathrm{Opt}(\mathcal{G}(\cdot), \mathbf{v}, \bar{\mathbf{z}}, k)$$

which applies k iterations of the convergent optimization algorithm to the problem

$$\min_{\mathbf{z}} \mathcal{G}(z) - \mathbf{v}^T\mathbf{z}$$

with initial guess $\bar{\mathbf{z}}$ to obtain $\mathbf{z}^+$. If the parameter k is omitted, the optimization algorithm continues to run until its termination criteria are satisfied. If OPT is applied by itself to the high-fidelity model, then $v = 0$. In the context of the multilevel algorithm MG/Opt, non-zero choices of v will be used.

To describe an iteration of MG/Opt we make reference to a high-fidelity model $\mathcal{G}_h$ and a low-fidelity model $\mathcal{G}_H$. The letters h and H are used repeatedly to identify the high- and low-fidelity information. Also required are a downdate operator I_h^H and an update operator I_H^h that map sets of generators $\mathbf{z}$ from one level to another. Unlike the algorithm in [**?**], the version of MG/Opt here uses a separate downdate operator $\hat{I}_h^H$ to map values of $\nabla\mathcal{G}$ from the fine level to the coarse level.

Here is a description of MG/Opt: Given an initial estimate of the solution $\mathbf{z}_h^0$ on the fine level, set $\mathbf{v}_h = 0$. Select non-negative integers k_1 and k_2 satisfying $k_1 + k_2 > 0$. Then for $j = 0, 1, \ldots$, set

$$\mathbf{z}_h^{j+1} \leftarrow \mathrm{MG/Opt}(\mathcal{G}_h(\cdot), \mathbf{v}_h, \mathbf{z}_h^j)$$

where the function MG/Opt is defined as follows:

- *Coarse-level solve:* If on the coarsest level, then solve the optimization problem:

  $$\mathbf{z}_h^{j+1} \leftarrow \mathrm{Opt}(\mathcal{G}_h(\cdot), \mathbf{v}_h, \mathbf{z}_h^j).$$

  Otherwise,
- *Pre-smoothing:*

  $$\bar{\mathbf{z}}_h \leftarrow \mathrm{Opt}(\mathcal{G}_h(\cdot), \mathbf{v}_h, \mathbf{z}_h^j, k_1)$$

- *Recursion:*
  - Compute

    $$\begin{aligned} \bar{z}_H &= I_h^H \bar{z}_h \\ \bar{v} &= \hat{I}_h^H \mathbf{v}_h + \nabla\mathcal{G}_H(\bar{z}_H) - \hat{I}_h^H \nabla\mathcal{G}_h(\bar{z}_h) \end{aligned}$$

  - Apply MG/Opt recursively to the surrogate model:

    $$z_H^+ \leftarrow \mathrm{MG/Opt}(\mathcal{G}_H(\cdot), \bar{v}, \bar{z}_H)$$

  - Compute the search directions $e_H = z_H^+ - \bar{z}_H$ and $e_h = I_H^h e_H$.

– Use a line search to determine $z_h^+ = \bar{z}_h + \alpha e_h$ satisfying $\mathcal{G}_h(z_h^+) \leq \mathcal{G}_h(\bar{z}_h)$.

- *Post-smoothing:*

$$\mathbf{z}_h^{j+1} \leftarrow \mathrm{Opt}(\mathcal{G}_h(\cdot), v_h, \mathbf{z}_h^+, k_2)$$

Because the integers k_1 and k_2 satisfy $k_1 + k_2 > 0$, each iteration of MG/Opt includes at least one iteration of the convergent optimization algorithm OPT. This fact, in combination with the line search to determine z_h^+, makes it possible to prove that MG/Opt is guaranteed to converge in the same sense as OPT [**?**].

In this paper, we have chosen OPT to be the truncated-Newton algorithm TN [**?**]. When applied to an optimization problem of the form

$$\min_{\mathbf{z}} f(\mathbf{z}),$$

at the j-th iteration a search direction p is computed as an approximate solution to the Newton equations

$$\nabla^2 f(\mathbf{z}_j) p = -\nabla f(\mathbf{z}_j)$$

where $\mathbf{z}_j$ is the current approximation to the solution of the optimization problem. The search direction p is computed using the linear conjugate-gradient algorithm. The necessary Hessian-vector products are estimated using finite differencing. The TN algorithm only requires that values of $f(\mathbf{z})$ and $\nabla f(\mathbf{z})$ are computed. TN has low storage requirements, and has low computational costs per iteration, and hence is suitable for solving large optimization problems [**?**].

3. Applying MG/Opt to the CVT Formulation. To make sure we are dealing with the same underlying problem on different levels, we scaled the objective function value by k^2 so that the optimal energy value would be same on every level for the uniform density. This is also approximately true for other density functions. The coarser grids were obtained by standard coarsening, i.e. doubling grid size at every level: $N_H = \frac{1}{2} N_h$, where N_h and N_H are the numbers of grid points at the finer and coarser grids, respectively.

To define our algorithm for solving the CVT problem, we need to adapt the formulation above to the CVT context and specify the appropriate choices of the operators and algorithm parameters.

3.1. MG/Opt setup. It turns out that in 1-dimensional case the following choice of the transfer operators works best in the CVT MG/Opt context. Transfer of the solution from finer to coarser grid (error restriction) is done via simple *injection*, i.e. given a vector $\mathbf{v}_h$ on the fine level, the downdate operator I_h^H samples $\mathbf{v}_h$ at the even indices:

$$[I_h^H v_h]^i = v_h^{2i}, \quad i = 1, 2, \ldots, k/2.$$

The corresponding downdate operator $\hat{I}_h^H$ for the gradient (gradient restriction) that maps values of $\nabla\mathcal{G}$ from the fine level to the coarse level is given by

$$[\hat{I}_h^H v_h]^i = \frac{1}{2} v_h^{2i-1} + v_h^{2i} + \frac{1}{2} v_h^{2i+1}, \quad i = 1, 2, \ldots, k/2.$$

This form corresponds to the a standard choice of a full weighting (FW) restriction operator scaled by a factor of 2, which comes from scaling the objective function at

each level. The update (interpolation) operator I_H^h is then given by a standard bilinear interpolation operator, which satisfies $I_H^h = 2(FW)^T$ (see p.61 of [**?**] for example), as is the case when solving an optimization problem based on a discretized PDE in one dimension. For our choice of the gradient restriction we obtain the relationship $I_H^h = (\hat{I}_h^H)^T$, that we rely on in our implementation.

Similarly to the 1D, transfer of the solution from finer to coarser grid in 2D can be computed by injection. However, the other transfer operators have to be chosen differently due to the differences in the CVT geometry. For the gradient downdate operator the following choice has been shown to work:

$$[\hat{I}_h^H v_h]^i = \sum_j \alpha_j^i v_h^j,$$

where $\alpha_i^i = 1$ and $\alpha_j^i = \frac{1}{2}$ for any j s.t. z_j is a fine node sharing an edge with z_i in the fine level triangulation. In the calculations provided in Section **??**, we fix these weights according to the equilibrium configuration. Although this setup is specific to the situation when the exact solution has a hierarchical structure with a nested set of grids (like in the case of triangular domain with $\rho = 1$), it can be generalized to arbitrary domains via recomputing the connectivity matrix at each step and re-assigning the interpolation weights accordingly. The exact procedure will be provided in a forthcoming publication. The corresponding update operator for the error can be provided by the formula analogous to the 1-D case: $I_H^h = 4(\hat{I}_h^H)^T$.

In both dimensions, for a $V_{1,1}$ cycle implemetation, one pre-smoothing and one post-smoothing relaxations were used ($k_1 = k_2 = 1$), which corresponds to applying one iteration of OPT in each of these steps. We have tested several convergence criteria. As we demonstrate in Section **??**, the best convergence was achieved when both OPT and MG/Opt were terminated due to saturation of the energy gradient $|\mathcal{G}(\mathbf{z}) - \mathcal{G}(\mathbf{z}^*)| \leq 10^{-8}$, where $\mathbf{z}^*$ is the solution to the CVT problem. In 1-dimensional nonlinear density case, the estimation of the exact solution $\mathbf{z}^*$ was obtained by running OPT with a tight convergence tolerance. Exact solution was computed analytically for the case of constant density in both dimensions.

It is also worth mentioning that we reordered the generators at each energy evaluation, since the order is not necessarily preserved by the optimization procedure described above. The symmetry of the CVT energy $\mathcal{G}$ with respect to re-orderings of the points $\{\mathbf{z}_i\}_{i=1}^k$ assures that the objective function is not suffering from this modification.

4. Numerical Experiments. All numerical experiments have been performed on 2-core Intel Duo CPU E8400 3.00GHz platform with 3.25 GB of RAM and the running times might differ for other configurations.

We did not use any preprocessing to generate the starting configuration $\mathbf{z}_h^0$ for our algorithm, which was simply obtained from a random sampling in the interval $[0, 1]$. In Matlab, we reset the random generator seed each time and supplied the initial generators by means of the commands $\texttt{rand}('\texttt{state}', 0), \texttt{v0} = \texttt{sort}(\texttt{rand}(\texttt{k}, 1))$ to eliminate the effect of initial configuration when benchmarking algorithm performance.

4.1. 1-dimensional examples. All 1-dimensional examples have been computed on the $[0, 1]$ domain. In the 1-dimensional case, the objective function (**??**) and its gradient can be integrated analytically in the case of $\rho = 1$ and require numerical approximation for more complicated cases. For the types of nonlinear densities

chosen in our numerical experiments we have tested several quadratures, including Simpson's and 15-point Gauss-Kronrod rules, with no significant change in algorithm performance.

In the numerical experiments below we compare the multilevel algorithm MG/Opt with the single level counterpart (OPT) given by the Truncated Newton (TN) method. Since the TN algorithm is minimizing the energy $\mathcal{G}$ at every step, the approach is comparable to the Lloyd algorithm (see [**?**]), which is the usual benchmark used for CVT algorithms. We measure the computational cost of OPT by counting the number of fine-level gradient evaluations of $\mathcal{G}$, which estimates the dominant cost of using OPT. For MG/Opt we count the number of equivalent fine-level gradient evaluations. That is, we determine (for each level) the relative cost of a gradient evaluation compared to an evaluation on the fine level.

Our first test uses the uniform density $\rho(y) = 1$ with 512 variables. This is an easy problem. MG/Opt converges at a fast linear rate, about 12 times as fast as OPT (see Figure **??**). Next we use the density $\rho(y) = 6y^2e^{-2y^3}$. The results are in Figure **??**. In this case the performance of MG/Opt is almost the same, but OPT converges more slowly. Again we are able to achieve fast linear convergence using the multilevel method. The convergence factor here is computed as follows:

$$C = \Big(\frac{|\mathcal{G}(\mathbf{z}^{k+1}) - \mathcal{G}(\mathbf{z}^*)|}{|\mathcal{G}(\mathbf{z}^1) - \mathcal{G}(\mathbf{z}^*)|} \Big)^{\frac{1}{k+1}}, \tag{4.1}$$

where $\mathcal{G}(\mathbf{z}^*)$ is the approximation of the exact solution precomputed by running OPT until saturation.

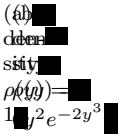


Fig. 4.1: Comparison of MG/Opt versus OPT: (a) for density $\rho(y) = 1$, (b) density $\rho(y) = 6y^2e^{-2y^3}$. Circle: MG/Opt; Star: OPT; Dot: Lloyd.

In Figure **??** we analyze the performance of MG/Opt and OPT as the size of the problem increases. In these tests we used the uniform density $\rho(y) = 1$ and averaged the results over 6 independent runs with random initial guesses. In the left plot we display the number of iterations needed for MG/Opt and OPT to compute the objective function value to a specified accuracy. In the middle figure we display the time needed to do this. Both MG/Opt and TN are expected to converge at a linear rate, and the rate constant ("convergence factor") can be estimated from the results of each test. The right plot displays the rate constants for both algorithms for different problem sizes. As can be seen, for MG/Opt the rate constant is insensitive to the problem size, whereas for OPT the rate constant deteriorates as the problem size increases. Exact numerical values for both number of cycles, convergence factor and elapsed time are given in Table **??**. Both MG/Opt and OPT algorithms have been stopped when the residual reaches a tolerance threshold of 10^{-8}.

Since the CVT algorithms in general show sensitivity to the choice of the initial configuration, we ran several tests with an intentionally "bad" choice of the initial

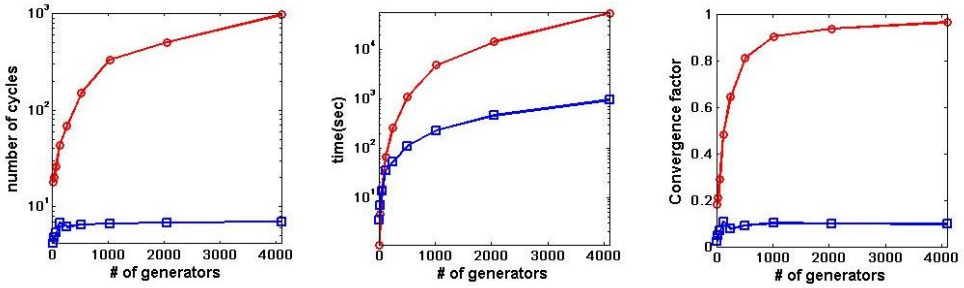


Fig. 4.2: Solving problems of increasing size, MG/Opt versus OPT ($\rho(y) = 1$). Blue-Square: MG/Opt, Red-Circle: OPT; Cycle numbers and Time elapsed are log scaled

	convergence factor		number of cycles		time(sec)	
k	MG/Opt	OPT	MG/Opt	OPT	MG/Opt	OPT
16	0.0234	0.1803	4	18	2.63	1.08
32	0.0440	0.2094	5	20	6.02	4.44
64	0.0952	0.2897	6	26	12.06	14.11
128	0.0604	0.4839	5	43	20.59	67.31
256	0.1174	0.6474	6	69	44.30	256.67
512	0.0932	0.8129	6	150	91.73	1120.31
1024	0.0796	0.9076	6	330	178.44	4831.66
2048	0.0728	0.9403	6	500	398.56	14568.47
4096	0.0757	0.9679	6	970	820.28	55370.94

Table 4.1: Comparing performance of MG/Opt with 1-level optimization for $\rho = 1$

guess, with all generators clustered near the origin in the $[0, 1]$ region. As Figure **??** demonstrates, no significant difference in performance has been noted.

Figure **??** shows the performance of MG/Opt when different stopping criteria are used. Again we use the uniform density $\rho(y) = 1$. From the plot, we can see that the common stopping criterion based on the improvement in the residual $\nabla\mathcal{G}$ (which is related to the difference between the positions of generators in the approximated and exact solution) is not the ideal choice here since it leads to a slight increase in the iteration count as the problem size increases. For the other two criteria based on the energy function $\mathcal{G}$, both the number of cycles and the convergence factors in MG/Opt are not sensitive to problem size. The other reason to prefer the criterion basen on energy values rather than positions is the abundance of local minima in the large-scale CVT problems, which makes it impossible to pinpoint the exact solution the method converges to. The energy value provides a reasonable measure for the closeness to a particular exact solution. However, as demonstrated by Figure **??**, both criteria are acceptable and performance differences are minor. It is also worth noting that based on the time complexity shown here the algorithm is linearly scalable — a desirable feature of the multigrid formulation that allows for an efficient parallelization for modern large-scale computing applications.

Figure **??** demonstrates the performance of MG/Opt when applied to problems

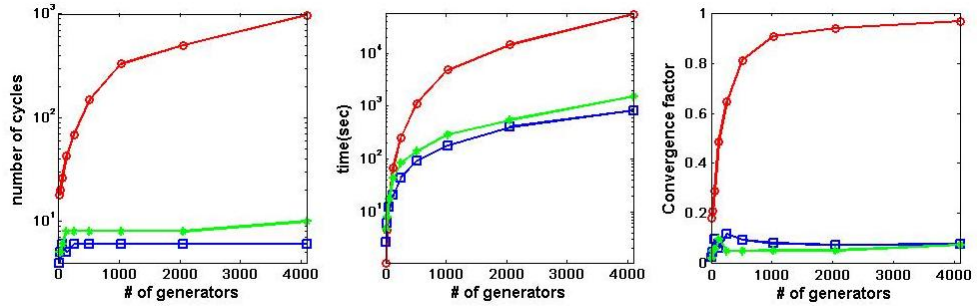


Fig. 4.3: Performance of MG/Opt with different initial configurations. Red-Circle: OPT; Green-Star: MG/Opt with "bad" initial condition; Blue-Square: MG/Opt with random initial condition. Number of cycles and elapsed time axes are log-scaled.

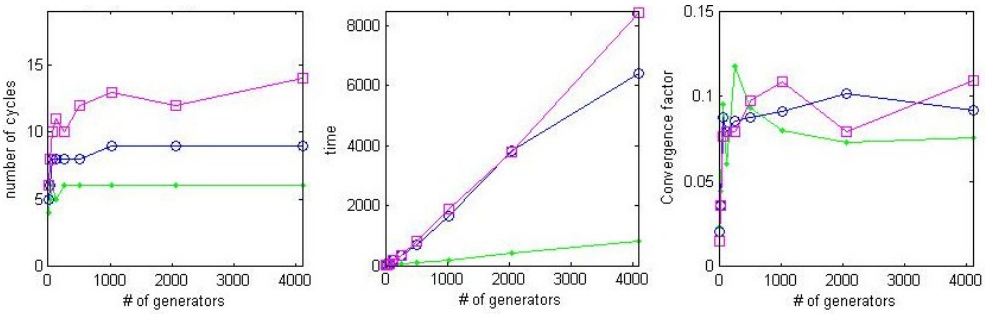


Fig. 4.4: Performance of MG/Opt with different stopping criteria ($\rho(y) = 1$). Blue-Dot: $|\mathcal{G}(\mathbf{z}) - \mathcal{G}(\mathbf{z}^*)| \leq 10^{-8}$; Red-Circle: $|\mathcal{G}(\mathbf{z}) - \mathcal{G}(\mathbf{z}^*)| \leq 10^{-11}$; Green-Square: $\| \nabla \mathcal{G} \|_\infty \leq 10^{-6}$.

with different density functions. In particular we consider $\rho_1(y) = 1 + 0.1y$ and $\rho_2(y) = 6y^2 e^{-2y^3}$. In the left plot, we can see that the iteration count for ρ_1 stays constant with the increase of the problem size, and increases only slightly for ρ_2. For both densities the convergence factor exhibits insensitivity to the problem size, although the convergence factor is slightly larger for ρ_2, while remaining bounded from above by 0.2.

Figure **??** compares the performance of MG/Opt to that of the multilevel optimization based method introduced in [**?**]. Although the latter was also based on the idea of formulating an optimization problem based on the CVT energy (**??**), it has significant differences with the MG/Opt approach. Essentially, the previous scheme represented a successive correction algorithm with a suitably defined domain decomposition, with the Lloyd iteration used as a relaxation on each level — hence the name Multilevel-Lloyd we use to refer to it in this work. Comparison of performance of both methods for constant and linear densities shows that MG/Opt yields slightly lower convergence factors, while showing significant advantage in terms of the time complexity.

Table **??** compares the performance of MG/Opt with that of the OPT algorithm

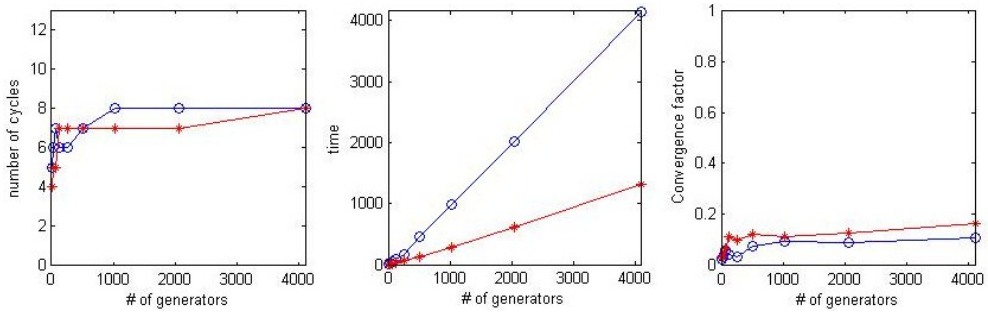
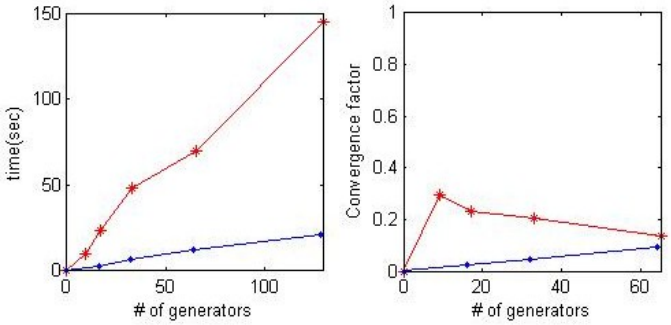
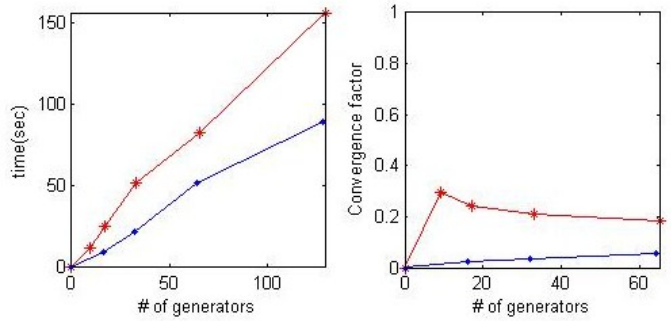


Fig. 4.5: Performance of MG/Opt with different density functions. Circle: $\rho(y) = 1 + 0.1y$. Star: $\rho(y) = 6y^2e^{-2y^3}$



(a) density $\rho(y) = 1$



(b) density $\rho(y) = 1 + 0.1y$

Fig. 4.6: Convergence factors for MG/Opt vs. Multilevel-Lloyd [**?**] for (top) uniform density $\rho(y) = 1$; (bottom) linear density $\rho(y) = 1 + 0.1y$. Notations: Dot:MG/Opt; Star:Multilevel/Lloyd

and the Multigrid Lloyd-Max (MLM) algorithm introduced in [**?**], which is based on a FAS formulation of the Lloyd-Max iterative process. The same form of the nonlinear density $y = 6x^2e^{-2x^3}$ as the one tested in [**?**] has been chosen for fair comparison. The column MG[**?**] shows the convergence factors reported in [**?**], where the convergence

	convergence factor				number of cycles	
k	MG/Opt^2	MG/Opt^1	MLM[?]	OPT	MG/Opt^2	OPT
16	0.0350	0.0023	0.1725	0.8060	4	108
32	0.0314	0.0177	0.1782	0.9024	5	228
64	0.0673	0.0551	0.1847	0.9381	5	354
128	0.1120	0.0673	0.1962	0.9736	7	846
256	0.0980	0.0983	———	0.9877	7	1818
512	0.1230	0.1927	———	0.9943	7	4276
1024	0.1105	0.1504	———	0.9971	7	8388
2048	0.1253	0.2890	———	0.9983	7	14297

Table 4.2: Comparing performance of MG/Opt with 1-level optimization (OPT) and the Multilevel Lloyd-Max (MLM) method of [?] for a nonlinear density: $y = 6x^2e^{-2x^3}$. MG/Opt^1 denotes the MG/Opt algorithm with the convergence factor computed according to (??), MG/Opt^2 uses the geometric mean convergence factor (??); MLM[?] is the result from [?] with the same convergence factor formula as MG/Opt^1 , '——' refers to data not provided in [?]

factors were computed using

$$C = \frac{|\mathcal{G}(\mathbf{z}^{k+1}) - \mathcal{G}(\mathbf{z}^*)|}{|\mathcal{G}(\mathbf{z}^k) - \mathcal{G}(\mathbf{z}^*)|} \tag{4.2}$$

The column MG/Opt^1 lists the convergence factors for our algorithm MG/Opt computed the save way. The column MG/Opt^2 lists the convergence factors for MG/Opt computed according to the formula (??), as we did in our earlier discussion. This latter formula is not as sensitive to the performance of the algorithm at the final iteration. It is also the formula used in our other numerical experiments when computing the convergence factor. Notice that the convergence factors for MG/Opt are significantly better than for MLM [?]. The fact that convergence factors stay very small even for much larger problems suggests that the new method has the potential to outperform earlier formulations, even for highly nonlinear densities.

4.2. 2-dimensional examples. Below are some preliminary results that have been obtained in the 2-dimensional case for the triangular domain. The choice of the domain is motivated by the availability of easily computable exact solution and the absence of boundary effects inherent to rectangular shaped regions.

In Figure ??, we provide convergence results for MG/Opt and its competitors when the initial guess is taken to be relatively close to the local minimizer. It is clear that MG/Opt maintains its superiority over 1-level methods. Moreover, MG/Opt has a significantly lower convergence factors independent of the problem size, similar to the behavior observed in 1-dimensional case.

Table ?? provides numerical data for the performance of MG/Opt, OPT and Lloys methods for the case of initial guess taken further away from the minimizer. Again, there is a clear advantage in using MG/Opt in this case. It has also been noted that the tendency to move points outside of the domain typical for the regular 1-level OPT method is significantly reduced in the 2-dimensional MG/Opt implementation.

Overall, while the above estimates are only a first step in a comprehensive analysis of the 2-dimensional version of the algorithm, and more work needs to be done to

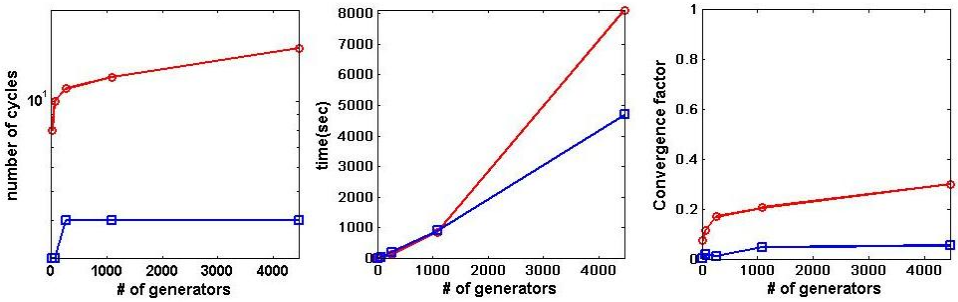


Fig. 4.7: Comparison of performance of MG/Opt and OPT for the case of a triangular 2-d region with $\rho = 1$. Red-Circle: OPT, Blue-Square: MG/Opt.

	convergence factor			number of cycles		
k	MG/Opt	OPT	Lloyd	MG/Opt	OPT	Lloyd
10	0.0127	0.0871	0.8323	2	6	54
55	0.0231	0.1378	0.9554	4	10	244
253	0.0121	0.1957	0.9891	4	14	1143
1081	0.0092	0.4055	0.9973	4	28	5265

Table 4.3: Comparing performance of 2-dimensional MG/Opt with 1-level optimization (OPT) and Lloyd method for constant density in triangular domain.

make the method work efficiently for general 2-d domains and density functions, they give a reason to believe MG/Opt will be competitive to other methods. Analysis of the performance of the 2-d method for different choices of transfer operators, and the creation of a robust and versatile general-purpose solver are the focus of current investigations.

5. Complexity of MG/OPT for 1-d CVT problem. In this section, we analyze the computational complexity of a $V_{1,1}$ cycle of the MG/Opt algorithm for CVT, i.e. $k_1 = k_2 = 1$. At each level, MG/Opt consists of Pre-smoothing and Post-smoothing which amount to two iterations of the Truncated Newton (TN) method, plus the cost of transferring the solution from one grid to another (downdate and update) and line search. Leaving the TN part aside, let us first compute the work required for all other parts of the algorithm, similar to the analysis performed in [**?**]. Notice that: (a) downdating the variables is carried out by means of simple injection, so no additional computational cost is involved; (b) the update operator is applied to half of the points in the grid during the refinement process; (c) the process of downdating and finding local search directions is only applied to half of the points in the coarsening process; (d) these steps are performed in the beginning and at the end of each cycle. The setup is performed once per outer iteration. With these observations in mind, we obtain the estimate of computational cost for all these operations to be less than $9k$ for the problem of size k, as shown in Table **??**.

To estimate the complexity of the TN part, notice that if we only do one iteration of TN in pre-smoothing and post-smoothing, respectively, the following computations

Operations	per coordinate	on the fine level
Downdate(variable)	0	0
Downdate(residual)	3	$3k/2$
Search Direction	1	$k/2$
Update	3	$3k/2$
Line search	1	k

Table 5.1: The complexity of various parts of MG/Opt measured in floating point operations.

are to be performed:

- 1 infinity-norm calculation, 1 vector addition and 2 function-gradient evaluations
- Conjugate-gradient (CG) iteration with cost per iteration given by:
  1 function-gradient evaluation,
  4 inner product and
  5 "vector + constant·vector" operations.

On average, we do 5 CG iterations per TN iteration. Based on the above estimate, one TN iteration requires 1 infinity norm, 20 inner product, 25 "vector + constant·vector", 1 vector add calculations and 7 function/gradient evaluations. If the cost of infinity norm calculation is estimated to be no more that k flops (floating-point operations, which include additions, subtractions, products and divisions), with k being the number of grid points at the finest level, and the cost of each the inner product and "vector + constant·vector" calculations is taken to be $2k$ flops, the overall cost of one TN iteration amounts to $k(1+20+25+1) = 47k$ flops, plus 7 function-gradient evaluations. Since in each $V_{1,1}$ cycle we do 2 TN iterations at each level and the contribution from coarser grids is estimated from above as $\sum_{i=0}^{\log k} 2^{-i} < 2$, we multiply this estimate by a factor of 4, and add the additional costs as specified above. We conclude that the typical cost of a MG/Opt V-cycle is $(4 \cdot 47 + 9)k = 197k$, plus $28k$ function-gradient evaluations. This proves the fact that the above algorithm has $O(k)$ complexity. This bound can be lowered by performing less safeguarding as part of the TN algorithm at each step, however, we feel these steps are needed in order to obtain the most robust implementation.

6. Comments and Conclusions. In conclusion, we have introduced a novel way of computing CVTs by means of an $O(k)$ complexity algorithm based on the MG/Opt strategy originally used only for large-scale nonlinear optimization problems. The main advantage of the new method is its superior convergence speed when compared to other existing approaches. Even though the computational complexity of each cycle is not the lowest, the fact that the convergence factors are on average much lower than those of its counterparts makes this formulation attractive when it comes to solving large CVT problems. The simplicity of its design and the results of preliminary tests suggest that the method is generalizable to higher dimensions, which is the subject of current investigations. Rigorous convergence theory development for this type of a method is nontrivial but possible, and will be addressed in a future publication. Future work also includes application of this technique to various scientific and engineering applications, including image analysis and grid generation.

7. Acknowledgements. The work of Zichao Di and Stephen G. Nash was supported by the U.S. Department of Energy under Award DE-SC-0001691. Maria Emelianenko acknowledges the support from the ORAU Ralph E. Powe Junior Faculty Enhancement Award.