A MULTIGRID OPTIMIZATION FRAMEWORK FOR VECTOR QUANTIZATION

by

Zichao Di
A Dissertation
Submitted to the
Graduate Faculty
of
George Mason University
In Partial fulfillment of
The Requirements for the Degree
of
Doctor of Philosophy
Mathematics

Committee:

________________________  Dr. Stephen Nash, Dissertation Director

________________________  Dr. Maria Emelianenko, Dissertation co-Director

________________________  Dr. Padmanabhan Seshaiyer, Committee Member

________________________  Dr. Ariela Sofer, Committee Member

________________________  Dr. David Walnut, Department Chair

________________________  Dr. Timothy L. Born, Associate Dean for
                              Student and Academic Affairs, College of Science

________________________  Dr. Peggy Agouris, Dean,
                              College of Science

Date: _____________________  Summer Semester 2013
George Mason University
Fairfax, VA
A Multigrid Optimization Framework for Vector Quantization

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at George Mason University

By

Zichao Di
Bachelor of Science
Dalian University of Technology, 2008

Director: Dr. Stephen Nash, Professor
Department of Systems Engineering & Operations Research
Co-Director: Dr. Maria Emelianenko, Associate Professor
Department of Mathematical Science

Summer Semester 2013
George Mason University
Fairfax, VA
This dissertation is dedicated to my immediate family members: my mom, dad, and my grandparents. Thank you for all of your love, support, and sacrifice throughout my life.
Acknowledgments

Foremost, I would like to express my sincere gratitude to my advisors, Dr. Stephen Nash and Dr. Maria Emelianenko, for their continuous support for me to proceed through the doctoral program and complete my dissertation. It has been an honor to be their Ph.D. student. They served as role models to me and provided their patience, motivation, enthusiasm and immense knowledge to me without reservation. I appreciate all their contributions of time, ideas, and funding to make my Ph.D. experience productive and stimulating.

Besides my advisors, I would like to thank the rest of my thesis committee: Dr. Ariela Sofer and Dr. Padmanabhan Seshaiyer, for their time, interest, and insightful comments.

My sincere thank also goes to my friend: Carmen Caiseda, for her continuous support and encouragement during my way of pursuing my degree. She is the first foreign friend I made in America and I always feel lucky being able to meet her at the first place.

I gratefully acknowledge the funding sources that made my Ph.D. study possible. I was funded by the U.S. Department of Energy project under Award DE-SC-0001691 for three years and funded by the Department of Mathematical Sciences in George Mason university for 1 year. My work was also supported by the National Science Foundation under Award DMS-0915013.

Last but not the least, I wish to thank my parents, Zhennan Di and Xiaomei Zhang. Their love provided my inspiration and was my driving force. I hope this work makes them proud.
5.3 1D advection problem ................................................. 81
5.4 2D linear control problem ............................................. 82
5.5 Summary of constrained version of MG/OPT .......................... 85
6 Conclusion and future work ............................................ 86
A Definitions and Background Information ................................ 88
  A.1 Elliptic PDE .......................................................... 88
  A.2 Relaxation method .................................................... 88
  A.3 Stencil notation ....................................................... 89
  A.4 Jacobi method ......................................................... 89
  A.5 Gauss-Seidel method .................................................. 90
  A.6 red-black Gauss-Seidel method ...................................... 91
  A.7 Newton’s method ...................................................... 92
  A.8 Linear Interpolation in stencil notation ............................ 92
  A.9 Bilinear Interpolation ............................................... 93
  A.10 Full-weighting Restriction in stencil notation ..................... 93
  A.11 Nested Iteration ..................................................... 93
  A.12 Convergence factor .................................................. 93
  A.13 Interior point Method ............................................... 93
  A.14 Linear Conjugate gradient method ................................ 94
  A.15 Delaunay Neighboring ............................................... 94
  A.16 Truncated Newton method ......................................... 95
  A.17 Maximal Independent set algorithm ................................ 95
  A.18 Barycentric Interpolation .......................................... 95
  A.19 Schur complement method ......................................... 95
  A.20 Richardson iteration ............................................... 96
  A.21 Karush-Kuhn-Tucker (KKT) conditions ............................ 96
  A.22 Triangular number ................................................... 97
  A.23 Discrete Laplace operator: five point stencil .................... 97
Bibliography .................................................................. 98
# List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Measured convergence factors of the red-black GS based V cycle for problem (2.1) on grids of different mesh size. In each case the coarsest grid has a mesh size of $h_0 = 1/2$.</td>
<td>8</td>
</tr>
<tr>
<td>4.1</td>
<td>Comparing performance of MG/OPT with 1-level optimization for $\rho = 1$.</td>
<td>50</td>
</tr>
<tr>
<td>4.2</td>
<td>Comparing performance of MG/OPT with 1-level optimization (OPT) and the Multilevel Lloyd-Max (MLM) method of [KYS03] for a nonlinear density: $y = 6x^2e^{-2x^3}$. MG/OPT$^1$ denotes the MG/OPT algorithm with the convergence factor computed according to (4.2), MG/OPT$^2$ uses the geometric mean convergence factor (4.1); MLM[KYS03] is the result from [KYS03] with the same convergence factor formula as MG/OPT$^1$, ‘——’ refers to data not provided in [KYS03].</td>
<td>54</td>
</tr>
<tr>
<td>4.3</td>
<td>The complexity of various parts of MG/OPT measured in floating point operations.</td>
<td>55</td>
</tr>
<tr>
<td>4.4</td>
<td>Comparing performance of 2-dimensional MG/OPT with 1-level optimization (OPT) and Lloyd method for constant density in triangular domain.</td>
<td>59</td>
</tr>
<tr>
<td>4.5</td>
<td>Comparing performance of 2-dimensional REFINE with 1-level optimization (OPT) for constant density in triangular domain.</td>
<td>63</td>
</tr>
<tr>
<td>4.6</td>
<td>Performance of a combination of MG/OPT and OPT v.s. OPT for constant density in a rectangular domain.</td>
<td>72</td>
</tr>
</tbody>
</table>
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>representation of regular hexagon in Cartesian coordinate system, $r$ is the radius of the circumscribed circle of the hexagon enclosed by $[x_{j1}, \ldots, x_{j6}]$, $r_1$ is the radius of the circumscribed circle of the hexagon enclosed by the neighboring generators to $x_i$ labeled as *</td>
</tr>
<tr>
<td>3.2</td>
<td>convergence factors for $\rho = 1$ with different number of generators $K$</td>
</tr>
<tr>
<td>4.1</td>
<td>initial error with a random initial guess</td>
</tr>
<tr>
<td>4.2</td>
<td>final error from OPT with a random initial guess</td>
</tr>
<tr>
<td>4.3</td>
<td>initial error with a special initial guess slightly perturbed from exact solution</td>
</tr>
<tr>
<td>4.4</td>
<td>final smoothed error from OPT with a special initial guess slightly perturbed from exact solution</td>
</tr>
<tr>
<td>4.5</td>
<td>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</td>
</tr>
<tr>
<td>4.6</td>
<td>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</td>
</tr>
<tr>
<td>4.7</td>
<td>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.</td>
</tr>
<tr>
<td>4.8</td>
<td>Performance of MG/OPT with different stopping criteria ($\rho(y) = 1$). Blue-Dot: $</td>
</tr>
<tr>
<td>4.9</td>
<td>Performance of MG/OPT with different density functions. Circle: $\rho(y) = 1 + 0.1y$. Star: $\rho(y) = 6y^2e^{-2y^3}$</td>
</tr>
</tbody>
</table>
4.10 Convergence factors for MG/OPT vs. Multilevel-Lloyd [DE08] for (top) uniform density $\rho(y) = 1$; (bottom) linear density $\rho(y) = 1 + 0.1y$. Notations: Dot:MG/OPT; Star:Multilevel/Lloyd

4.11 hierarchy of CVT with number of generators as a subset of triangular number: $[1, 10]$, $\lambda$ is the weight of the fine-to-coarse node, this pattern can be extended to larger set

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

4.13 demonstration of triangular numbers forming equilateral triangles [Wik11].

4.14 Hierarchy of CVT with 3 generators at coarse level, 6 generators at fine level, and $\lambda$ is the weight of coarse node to Delaunay neighboring fine node.

4.15 Optimal function value obtained by OPT minus optimal function value obtained by REFINE: $f_{opt} - f_{REFINE}^*$.

4.16 MG/OPT search direction versus error: dashed line: interpolated coarse level search direction $e_h^*$; solid line: error on fine level $e_h^*$.

4.17 wrong scale between step to solution and search direction.

4.18 the search direction matches the step to solution perfectly.

4.19 performance of MG/OPT with triangle number of generators.

4.20 first scheme to cover $R_s$ by equilateral triangles.

4.21 second scheme to cover $R_s$ by equilateral triangles.

4.22 further decomposition of $I$ into $I_1, A_{11}$ and $I_l$.

4.23 compare the performance of OPT two different initial guess. The red line represents the performance of OPT using the special initial guess obtained by PMG, and the blue line represents the performance of OPT using random initial guess.

5.1 1D Laplacian problem with lower bound, the bound constraints on coarse levels are the same as the fine level.

5.2 1D Laplacian problem with lower bound.

5.3 1-D Laplacian problem with upper and lower bounds.

5.4 Advection problem with upper bound.

5.5 2D Linear control problem with lower bound constants.

5.6 MG/OPT search direction versus error for problem (5.3).
5.7 Equivalent gradient evaluations comparison between MG/OPT and OPT for problem (5.3) ................................. 84
5.8 Equivalent gradient evaluations comparison between MG/OPT and OPT for problem (5.3) after rescaling the transform operators ........................ 85
Abstract

A MULTIGRID OPTIMIZATION FRAMEWORK FOR VECTOR QUANTIZATION
Zichao Di, PhD
George Mason University, 2013
Dissertation Director: Dr. Stephen Nash
Dissertation co-Director: Dr. Maria Emelianenko

In this thesis, we apply the multilevel optimization framework (MG/OPT) to a class of vector quantization problem. Specifically, we focus on a problem called centroidal Voronoi tessellations (CVTs).

MG/OPT is a generalization of traditional multigrid technique for solving partial differential equations. This framework assumes that there exists a hierarchy of models of the underlying optimization problem, corresponding to levels of detail in the problem, and the goal is to find the solution at the finest level. In other words, the intent of MG/OPT is to use calculations on coarser levels to accelerate the progress of the optimization on the finest level. In [Nas00,NL09,Nas10,LN00], it is emphasized that under appropriate assumptions, MG/OPT provides a descent direction, therefore, combining this fact with a line search and an underlying minimizing scheme, the global convergence of MG/OPT is guaranteed.

A Voronoi tessellation is a special kind of decomposition of a given space into Voronoi cells determined by distances to a set of objects, i.e., generators. A CVT is a Voronoi tessellation whose generators are the centroids of the corresponding Voronoi cell. There are two reasons we are particularly interested in this topic. First, contrary to other
existing common applications of MG/OPT whose solutions are based on regularly discretized grids, e.g., optimal control problems, CVT has an irregular underlying geometric structure which extends the use of MG/OPT to another type of applications. Second, due to the natural optimization properties enjoyed by CVTs, they have many applications in diverse fields, therefore, it is critical to have a fast and reliable scheme for constructing them. The Lloyd algorithm is one of the most popular iterative schemes for computing the CVTs but its theoretical analysis is far from complete.

This work to some extent fills this gap by showing that for 1D CVT with uniform density, the eigenvalues for the Jacobian of Lloyd map are \( \sin^2 \left( \frac{(2k+1)\pi}{2K+1} \right) \), \( k = 0, \ldots, K - 1 \) where \( K \) is the number of generators. In order to know better the nature of CVT, we also analyze the Hessian matrix obtained from its corresponding energy function. We show that the Hessian matrices for 1D and 2D CVTs are both positive definite.

In computational experiments, we study the performance of MG/OPT applied to 1D and 2D CVT separately. Uniform convergence and speedup achieved by MG/OPT comparing to other existing techniques are demonstrated for linear and nonlinear densities for several 1D and 2D problems, and \( O(K) \) complexity estimation is provided for problems with \( K \) generators.

Finally, we extend MG/OPT to solve certain types of constrained optimization problems. We consider PDE-constrained problems such as optimal control problems and advection problems. As a result, we are able to show the dramatic improvement of convergence speed by applying constrained version of MG/OPT.
Chapter 1: Introduction

1.1 Motivation

Multigrid methods are typically used to accelerate the convergence of a basic iterative method. Due to their superior convergence factor, they are accepted as among the most efficient modern techniques for solving large scale algebraic systems arising from the discretization of partial differential equations especially for elliptic partial differential equations [Hac85]. Furthermore, they are regarded as among the fastest methods for many other problems, like other types of partial differential equations, integral equations, etc. [Hac86]. A generalization of this multigrid idea is also able to successfully solve other type of structures other than grids, like multilevel, multiscale or multiresolution methods [DE07,AH01,KY06]. Inspired by these facts, researchers have extended the multigrid idea to solve optimization problems with PDE-constraints because such problems appear in many applications. One way is to recast the optimization problem as a system of equations derived from the Karush-Kuhn-Tucker (KKT) stationarity conditions, and then apply the traditional multigrid approach [BK06]. Another way is to apply linear multigrid to the Newton equations [JW01]. However, since the optimization models usually have more complicated nature than the systems of equations, to remodel them in terms of equations will consume considerably more effort. Especially for the constrained optimization problems, it may be impossible to deduce even qualitative information about the relevant operator. Therefore, a more general technique to utilize this multigrid idea is highly desirable.

One approach is to use the optimization-based multigrid framework (MG/OPT) which relies explicitly on optimization models as subproblems on coarser grids [Nas00]. There are several advantages of MG/OPT: first of all, the optimization perspective is broader
than systems of equations since it can handle the constraints in a more natural way. Also, MG/OPT has stronger guarantees of convergence than traditional multigrid. Furthermore, in many cases the reduced Hessian of the optimization model is an elliptic operator even if the PDE constraints are not elliptic.

So far, MG/OPT has been successfully applied to many PDE-constrained optimization problems which have regular geometric structures. Besides that, the current implementations of MG/OPT are all based on an unconstrained setting. Theoretically speaking, MG/OPT is available to solve problems under a constrained setting [Nas10]. Obviously, this capability is more attractive since, in reality, optimization problems with general constraints are more practical and most of them are difficult to transform to an unconstrained setting. However, in practice, more sophisticated implementations that provide an easier way to apply MG/OPT to specific problems are still under exploration. In order to develop one such practical version of this framework, we are particularly interested in a class of vector quantization problems: centroidal Voronoi tessellation (CVT).

A Voronoi diagram is a special kind of tessellation to decompose the given space $\Omega \subset \mathbb{R}^N$ based on a specified set of points $\{z_i\}_{i=1}^K$ called generators. It is defined as

$$V_i = \{ w \in \Omega \mid d(w, z_i) < d(w, z_j), \ j = 1, \ldots, K, i \neq j \}$$

for a given density function $\rho$ and distance function $d(z, w)$ defined on $w \in \Omega$ [Aur90, OBS92]. The centroids of regions $\{V_i\}_{i=1}^K$ are defined as

$$z_i^* = \frac{\int_{V_i} y \rho(y) \, dy}{\int_{V_i} \rho(y) \, dy}.$$ 

A 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, $z_i = z_i^*$ for all $i$. 
Correspondingly, we may define the energy function for \( \{ z_i \}_{i=1}^K \) as

\[
G(\{ z_i \}_{i=1}^K) = \sum_{i=1}^{K} \int_{V_i} \rho(y) |y - z_i|^2 \, dy.
\]

The minimizer of \( G \) forms a CVT which illustrates the optimization property of the CVT.

Due to the fact that the construction of higher dimensional CVT has to be constrained by a boundary of a chosen domain, it may be able to provide some insight into the implementation of the constrained version of MG/OPT. Moreover, CVT has an irregular underlying geometric structure contrary to the existing applications of MG/OPT. CVTs are in a big demand due to their optimality properties important for many applications. In particular, the CVT concept is getting more and more popular in large scale scientific and engineering problems such as data communication, vector quantization and mesh generation. Accordingly, the availability of fast and reliable algorithms for their construction is crucial for their successful use in practical settings.

In what follows, we focus our attention on CVT as an important application of the MG/OPT methodology. We explore the performance of MG/OPT applied to 1D and 2D CVTs separately. As a result, the multigrid efficiency is achieved on several types of test problems. In addition, we study theoretical properties of CVT such as the properties of the corresponding Hessian matrix so that more insight can be provided for the performance of MG/OPT in this setting.

The MG/OPT framework has been broadly applied for solving unconstrained optimization problems [Nas00, NL09, Nas10, LN00]. Diagnostic tests to measure the suitability of MG/OPT for specific problems have been developed in [NL09]. From the point of view of practical applications, it is highly desirable to extend MG/OPT to constrained problems. It has been theoretically shown that this extension is applicable in regard to convergence.
and descent. Despite that, to achieve rapid convergence for MG/OPT comparable to traditional multigrid scheme, there is still a need for more sophisticated implementations which would provide an easier way to apply MG/OPT to specific problems, e.g., what is an appropriate underlying solver, and how to wisely construct the coarse grid model and transfer operators. Therefore, another research effort has been directed towards developing a constrained version of MG/OPT to handle bound constraints for a wide variety of constrained problems. We choose the underlying solver designed for constrained problems to explicitly solve problems at each level of the hierarchy. This distinguishes our approach from other previous strategies since it avoids the commonly taken step of transforming the original constrained setting to an unconstrained form which is complicated and costly for large and complex problems. As a consequence, multigrid efficiency is shown for various types of PDE-constrained problems.
Chapter 2: An overview of MG/OPT and CVT

2.1 From Traditional Multigrid to MG/OPT

Before we start considering multigrid methods, for the purpose of intuition, we give an example of ‘multigrid phenomena’.

Consider the finite-difference discretization of

\[ -u'' = f \]  

on the interval \([0, 1]\) with Dirichlet boundary conditions \(u(0) = u(1) = 0\). On a uniform grid with interior points \(x_i = \frac{i}{n+1}\), \(i = 1, \ldots, n\), the corresponding algebraic problem for \(n = 7\) is given by the following linear system

\[
\begin{bmatrix}
2 & -1 & 0 & 0 & 0 & 0 & 0 \\
-1 & 2 & -1 & 0 & 0 & 0 & 0 \\
0 & -1 & 2 & -1 & 0 & 0 & 0 \\
0 & 0 & -1 & 2 & -1 & 0 & 0 \\
0 & 0 & 0 & -1 & 2 & -1 & 0 \\
0 & 0 & 0 & 0 & -1 & 2 & -1 \\
0 & 0 & 0 & 0 & 0 & -1 & 2 \\
\end{bmatrix} \begin{bmatrix}
u_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \\
\end{bmatrix} = h^2 \begin{bmatrix}
f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ f_6 \\ f_7 \\
\end{bmatrix}
\]

where \(h = \frac{1}{n+1}\), \(u_i = u(x_i)\), \(f_i = f(x_i)\), \(i = 1, \ldots, n\). Write the above system as \(A_h u_h = f_h\), and let \(E_i\) be row \(i\) of the corresponding system. Now we would like to approximate this
system on a grid coarser than the original grid by defining the coarsening steps

\[ \tilde{E}_i = E_{i-1} + 2E_i + E_{i+1}, \quad i = 2, 4, 6 \]

The system after one step of coarsening is

\[
\begin{bmatrix}
2 & -1 & 0 \\
-1 & 2 & -1 \\
0 & -1 & 2
\end{bmatrix} \cdot \begin{bmatrix}
u_2 \\
u_4 \\
u_6
\end{bmatrix} = (2h)^2 \begin{bmatrix} f_1 + 2f_2 + f_3 \\
f_3 + 2f_4 + f_5 \\
f_5 + 2f_6 + f_7
\end{bmatrix} = (2h)^2 \begin{bmatrix} \tilde{f}_2 \\
\tilde{f}_4 \\
\tilde{f}_6
\end{bmatrix}
\]

which is a smaller algebraic system with the same structure. Similarly, we can write it as \( A_H u_H = f_H \) where \( H = 2h \). In this sense we have ‘coarsening invariance’. Consequently, solving a system of 3 unknowns instead of 7 will give us part of the exact solution, and to repeat this process recursively, we can obtain the full solution, and meanwhile the computation costs are saved. This is the advantage for a system possessing coarsening invariance because, in fact, we can obtain an approximate solution of this system by solving its coarsened version.

As a consequence, to describe a solution of a given problem, the use of many scales of discretization is appropriate to represent all components of the solution. For example, a smooth function can be represented by a few of its values, whereas relatively highly oscillating functions require high resolution. This is one essential aspect of the multigrid strategy.

In the 1960s, R.P. Fedorenko developed the first multigrid scheme for the solution of the Poisson equation in a unit square. However, a better understanding of the efficiency of the multigrid approach was realized after the works of A. Brandt [Bra73, Bra77a] and W. Hackbusch [Hac83]. Briefly speaking, the fundamental idea of multigrid is to reduce the high frequency components of the error by smoothing procedures and to take care of the low frequency error components by coarse grid corrections.
Recall the previous boundary value problem. In order to use multigrid scheme to solve the equation system $A_h u_h = f_h$, we need to provide the prolongation operator $I_{2h}^h$ to map the search direction from coarse grid into fine grid, and the restriction operator $I_h^{2h}$ to map the defect $f_h - A_h u_h$ for the current approximate solution $u_h$ from the fine grid to the coarse grid. Initialize the estimate of the solution as $u_h^0$, use $u_h = S(u_h, f_h, k)$ to denote $k$ iterations of relaxation [A.2]. For the linear problem, $S$ is frequently chosen as a Jacobi-type [A.4] or a Gauss-Seidel-type [A.5] method. For nonlinear problems, $S$ is usually chosen as a Newton-type method [A.7]. Then the overall algorithm $u_k^h \leftarrow MG(u_{k-1}^h, A_h, f_h, k_1, k_2)$ corresponds to a V-cycle algorithm denoted as $V(k_1, k_2)$ consisting of a pre-smoothing with $k_1$ iterations of $S$, a coarse-grid correction and a post-smoothing with $k_2$ iterations of $S$. One step of such an iterative two-grid method proceeds as follows:

**Algorithm of multigrid(MG) scheme**

- Pre-smoothing on the fine grid: $\bar{u}_h = S(u_h^k, f_h, k_1)$
- Calculate the residual: $r_h = f_h - A_h \bar{u}_h$
- Restriction of the residual: $r_H = I_H^h r_h$
- Set $u_H = 0$
- on the coarse level
  - If on the coarsest level, solve $A_H u_H = r_H$ exactly to get $u_H^+$
  - Otherwise, apply MG recursively to the residual equation:
    $$u_H^+ \leftarrow MG(u_H, A_H, r_H, k_1, k_2)$$
- Coarse-grid correction: $u_h^+ = \bar{u}_h + I_H^h u_H^+$
- Post-smoothing on the fine grid: $u_h^{k+1} = S(u_h^+, f_h, k_2)$
As a numerical experiment on the 2-D version of the above boundary value problem, in [TOS01], the authors choose red-black Gauss-Seidel method [A.6] as relaxation, bilinear interpolation [A.9] as $I^h_H$, and full-weighting [A.10] as $I^H_h$. As a result, the numerically measured convergence factor of MG is essentially independent of the size of the finest grid which is known as the $h$-independent convergence factors as shown in table 2.1. Furthermore, if a concrete multigrid algorithm has this property, we speak of the typical multigrid efficiency.

<table>
<thead>
<tr>
<th>Cycle</th>
<th>$h = 1/512$</th>
<th>$h = 1/256$</th>
<th>$h = 1/128$</th>
<th>$h = 1/64$</th>
<th>$h = 1/32$</th>
<th>$h = 1/16$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$V(1,1):$</td>
<td>0.10</td>
<td>0.10</td>
<td>0.10</td>
<td>0.10</td>
<td>0.11</td>
<td>0.12</td>
</tr>
</tbody>
</table>

Table 2.1: Measured convergence factors of the red-black GS based $V$ cycle for problem (2.1) on grids of different mesh size. In each case the coarsest grid has a mesh size of $h_0 = 1/2$.

**Remark 2.1.** It has been shown in [BM+00] that, a commonly used, and highly useful requirement for transform operators is

$$I^h_H = C(I^H_h)^T, \quad \text{for } C \in \mathbb{R}$$

So far, we have considered linear problems. We will now discuss how multigrid methods can be used to solve nonlinear problems. An important achievement has been to introduce the multigrid method to nonlinear systems of equations via the full approximation scheme (FAS) [Bra77a, Hac85].

Initialize the estimate of solution as $u^0_h$, choose $I^h_H$ to map the search direction from coarse grid into fine grid, $I^H_h$ to restrict the residual from fine grid to coarse grid, and correspondingly, choose $I^H_h$ to restrict the solution from fine grid to coarse grid. Then $u^{k+1}_h \leftarrow \text{FAS}(u^k_h, L_h, f_h, k_1, k_2)$ refers to one iteration of the FAS scheme to solve nonlinear equation $L_h(u_h) = f_h$, and is summarized below:
Algorithm of nonlinear FAS scheme

- Pre-smoothing on the fine grid: $\bar{u}_h = S(u_h^k, f_h, k_1)$
- Calculate the residual: $r_h = f_h - L_h(\bar{u}_h)$
- Restriction of the residual: $r_H = I^H r_h$
- Set $u_H = I^H \bar{u}_h$
- Set $f_H = r_H + L_H(u_H)$
- on the coarse level
  - If on the coarsest level, solve $L_H(u_H) = f_H$ exactly to get $u^+_H$
  - Otherwise, apply FAS recursively to the coarse grid model:
    $$u^+_H \leftarrow FAS(u_H, L_H, f_H, k_1, k_2)$$
- Coarse-grid correction: $u^+_h = \bar{u}_h + I^h_H (u^+_H - u_H)$
- Post-smoothing on the fine grid: $u^{k+1}_h = S(u^+_h, f_h, k_2)$

Another possible approach to solve nonlinear problems is based on Newton’s method and uses the multigrid scheme as an inner solver for the linearized Newton equations. This is referred as a MG-Newton scheme. In general, although the FAS and MG-Newton may look quite different at first sight, they often show similar convergence under a specific implementation. In [Hen83], a comparison between FAS and MG-Newton schemes is presented. First, in terms of computing time, it is still an open question how many interior MG cycles are possibly needed in the MG-Newton method to match the FAS efficiency, and also, it is not necessary to compute and store the Jacobian in the FAS process, as is necessary in the MG-Newton methods. However, if considering the relative expense of evaluating the nonlinear function versus the cost of forming the Jacobian matrix, if the Jacobian matrix can
be evaluated with relative ease, MG-Newton may be preferable. In all, these determinations are highly dependent on the nature of the specific problem at hand.

In order to deal with sophisticated problems it is often important to have a good initial approximation to start the iterative procedure. Especially for any nonlinear method, it is well-known that the iteration is much more reliable and much faster if the initial guess is good. Hence, it is well worth the effort to obtain the highest quality initial guess possible. Therefore, A. Brandt and W. Hackbusch also developed another formulation of multigrid methods called the full multigrid (FMG) scheme [Bra77a,Hac85], based on the combination of nested iteration techniques and multigrid methods. The basic idea is to use nested iteration to provide a better initial guess to the MG scheme. Because of the improvement on the initial solution at each starting level, the FMG scheme results in a cheaper algorithm than the iterative application of the multigrid cycle without FMG initialization.

**Algorithm of FMG scheme**

- FMG method for solving \( L_h(u_h) = f_h \)

1. Compute \( u_H \) on the current grid \( H \) with initialization \( u_H^0 \)

2. If not on the finest grid, then interpolate to the next finer working grid: \( u_h = I_H^h u_H \)

3. Apply FAS or MG scheme to \( L_h(u_h) = f_h \)

4. If still not on the finest grid, go to 2

As is pointed out in [Hen83], combining the MG-Newton and FAS scheme with FMG has the effect of further decreasing the residual so as to improve both speed of convergence and robustness.

A more recent multigrid-based strategy is algebraic multigrid (AMG) developed by Brandt, McCormick and Huge in the early 1980s [Bra86]. AMG constructs the levels of the hierarchy by only utilizing the information from the algebraic system to be solved. The purpose of this approach is to overcome the issues that a uniform coarsening pattern will not
be available if the problem arises from unstructured meshes. The first fairly general AMG program, AMG1R5, was developed in the mid 1980s [RSH90]. AMG1R5 uses a hierarchical algorithm like geometric multigrid, however, it does not require the objective problem to be defined on a grid. The necessary components for the hierarchical algorithm, such as the coarse system matrices and the transfer operators are created only from information obtained in the algebraic equations.

An argument for combining multigrid with an acceleration technique is that problems become more and more complex if we treat real-life applications. For such complicated applications, it is far from trivial to choose optimal multigrid components uniformly for a large class of problems. In this case, the fundamental idea of multigrid doesn’t work optimally if straightforward multigrid approaches are used. Therefore, multigrid as a preconditioner [TOS01, chapter 7] is particularly suited with respect to robustness for some problems, for example, on unstructured grids, for problems with small geometric details and for problems with geometric singularities. Especially, the robustness of multigrid-like methods are enhanced by coupling them with conjugate gradient (CG) or other Krylov subspace acceleration techniques. In [Ket82], for positive semi-definite problems, the robustness of multigrid as a preconditioner for CG has been demonstrated. So the question is whether multigrid should be used as a solver or as a preconditioner. In [OW98], an evaluation of parallel multigrid as a solver and a preconditioner for singularly perturbed problems is given. The authors compared the convergence of three multigrid methods as preconditioners and solvers separately by analyzing their eigenvalue spectra. The behavior is much more robust when these three multigrid methods were used as preconditioners, since then the convergence was not sensitive to parameter changes.

The idea of adaptivity combined with multigrid was realized by Brandt who introduced the multilevel adaptive technique (MLAT) [Bra77a,Bra77b]. Adaptive multigrid exhibits adaptive mesh refinement, that is, it adjusts the grid as the computation proceeds, in a manner dependent upon the computation itself. The basic idea is to increase resolution of the grid only in regions of the solution where it is needed. See also [TOS01, chapter 9].
Starting in the early 1950s, Pontryagin, et al. [Pon62] introduced another modern field of research in applied mathematics which is to relate optimization to partial differential equations. For example, one popular class of such problem is as follows:

$$\min_{a} \quad F(a) = f(a, u(a))$$
subject to  \quad S(a, u(a)) = 0

where $a$ is a set of design variables, and $u = u(a)$ is a set of state variables. The purpose of this is to optimally change real-world systems to meet a given target. However, the increasing complexity in practical applications arising from data size, dimension, accuracy, discreteness, etc. requires the solution of large-scale optimization problems in an accurate and computationally efficient way. For this reason, the use of multigrid methods for optimization purposes is obtaining more and more interest from researchers.

Since traditional multigrid methods aim to solve system of equations, the most intuitive way to relate multigrid to optimization is to transform the optimization problem to system of equations obtained from Karush-Kuhn-Tucker (KKT) conditions. A simple example is to consider a quadratic optimization problem

$$\min_{u} \quad \frac{1}{2} u^T A u - b^T u.$$

Its minimizer happens to be the solution of setting gradient equal to zero, which is equivalently to solve $Au - b = 0$ provided that $A$ is a positive semi-definite matrix. Motivated by this observation, the earliest multigrid optimization approaches are based on the Schur-complement smoothing concept which intents to give an explicit reformulation of a decomposed Hessian matrix. In [BM+00], the nonlinear interior point methods used to solve topology optimization problems lead to linear subproblems under Schur-complement approaches. And this structure is well suited to apply multigrid methods. In [HAO04], a
multigrid method is used as a preconditioner for Krylov methods for the optimality conditions. And the convergence of the resulting performance is shown in [SW99]. For more detail about using multigrid as an inner solver for the optimality condition, see [BS09].

In [EG11], Engel and Griebel describe an efficient numerical solution of linear-quadratic optimal control problems with additional constraints on the control. They treat the complementarity conditions by a primal-dual active-set method as an outer iteration. At each iteration, they solve a KKT system within a multigrid framework using preconditioned Richardson iteration as the underlying smoother. The numerical experiments show that superlinear convergence of the outer iteration is obtained and the total cost for the solution of optimal control problems is just a small multiple of the cost for the solution of the constraint equations only. In their multigrid framework, the basic idea is to apply traditional multigrid to the equation system of the discretized PDE constraints. The difference is, instead of requiring that $I_H^H = C(I_H^H)^T$, they choose the downdate operator as the four point average restriction and the update operator as bilinear interpolation. In stencil notation [TOS01]:

\[
I_H^H = \frac{1}{4} \begin{bmatrix}
1 & 1 \\
1 & 1
\end{bmatrix}
\]

\[
I_H^h = \frac{1}{16} \begin{bmatrix}
1 & 3 & 3 & 1 \\
3 & 9 & 9 & 3 \\
3 & 9 & 9 & 3 \\
1 & 3 & 3 & 1
\end{bmatrix}
\]

Furthermore, when applying bilinear interpolation to obtain a fine grid value, the stencil entries for the active nodes are set to zero.

In [IV04], the authors described two multilevel schemes, the Multilevel Subspace Minimization Algorithm for linear systems and the FAS Constrained Optimization Algorithm.
for non-linear systems, for solving inequality constrained finite element second-order elliptic problems. The first important feature of these schemes is that they utilize element agglomeration coarsening away from the constraint set, which allows for easy construction of coarse level approximations that straightforwardly satisfy the fine grid constraints. Second is that they provide monotone reduction of the energy functional throughout the multilevel cycles by using monotone smoothers such as projected Gauss-Seidel iteration.

Around 2000, a new approach to use multigrid as an outer optimization scheme to treat unconstrained optimization problems was proposed by Nash [Nas00]. It is called MG/OPT. Instead of solving the KKT system, MG/OPT explicitly uses the optimization model on the fine and coarse levels.

2.2 The MG/OPT method for optimization problems

Consider the following convex optimization problem

\[
\min_{u_h} f(u_h)
\]

with an available low-resolution model \( f_H \) where \( h \) and \( H \) separately denote high- and low-resolution information. To describe one iteration of MG/OPT, first, we need the underlying optimization algorithm labeled as “OPT” which is assumed to be convergent in the sense that, under appropriate assumptions on the objective function \( f \),

\[
\lim_{j \to \infty} \| \nabla f(u_j) \| = 0
\]

where \( \{u_j\} \) are the iterates computed by OPT.

Define OPT as a function of the form

\[
u^+ \leftarrow \text{OPT}(f, v, \bar{u}, k)
\]
which applies \( k \) iterations of OPT to the problem

\[
\min_u f(u) - v^T u
\]

with initial guess \( \bar{u} \) to obtain \( u^+ \). If OPT is required to proceed until converged, we rewrite this as \( u^+ \leftarrow \text{OPT}(f, v, \bar{u}) \). Also given a downdate operator \( I_h^H \) and an update operator \( I_h^H \) to transform the information from one level to another, then MG/OPT is described as follows:

Given an initial estimate of the solution \( u_h^0 \) on the fine level, set \( v_h = 0 \). Select non-negative integers \( k_1 \) and \( k_2 \) satisfying \( k_1 + k_2 > 0 \). Then for \( j = 0, 1, \ldots \), set

\[
u_h^{j+1} \leftarrow \text{MG/OPT}(f_h, v_h, u_h^j)\]

where the function MG/OPT is defined as follows:

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

\[
u_h^{j+1} \leftarrow \text{OPT}(f_h, v_h, u_h^j).
\]

Otherwise,

- **Pre-smoothing:**

\[
\bar{u}_h \leftarrow \text{OPT}(f_h, v_h, u_h^j, k_1)
\]

- **Coarse-grid correction:**

  - Compute

\[
\bar{u}_H = I_h^H \bar{u}_h
\]

\[
\bar{v} = I_h^H v_h + \nabla f_H (\bar{u}_H - I_h^H \nabla f_h (\bar{u}_h))
\]
– Apply MG/OPT recursively to the surrogate model:

\[ u^+_H \leftarrow \text{MG/OPT}(f_H, \bar{v}, \bar{u}_H) \]

– Compute the search directions \( e_H = u^+_H - \bar{u}_H \) and \( e_h = I^H_H e_H \).

– Use a line search to determine \( u^+_h = \bar{u}_h + \alpha e_h \) satisfying \( f_h(u^+_h) \leq f_h(\bar{u}_h) \).

• Post-smoothing:

\[ u^{i+1}_h \leftarrow \text{OPT}(f_h, v_h, u^+_h, k_2) \]

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 \( u^+_h \), makes it possible to prove that MG/Opt is guaranteed to converge in the same sense as OPT [Nas10].

Theorem 4 in [Nas10] states that the coarse-grid model provides a descent direction under appropriate assumptions.

Roughly speaking, MG/OPT is inspired by nonlinear FAS, but they are not equivalent. The line search provided in MG/OPT is necessary to guarantee the convergence in the sense that \( \lim_{k \to \infty} \| \nabla f \| = 0 \) while FAS is not guaranteed to converge. Even if a line search is added to FAS, it is only guaranteed to find a local minimizer of \( \| \nabla f \| \). As is pointed out in [Nas00], without the line search, MG/OPT may diverge for some cases. Thus, MG/OPT has stronger convergence properties than FAS.

In [LN00], another benefit of MG/OPT is demonstrated via analyzing the optimization of systems governed by differential equations which represent a family of optimization problems. The evidence shows that even if the constraints governed by the differential equations are not elliptic, MG/OPT can still be effective.

In summary, Lewis and Nash [LN00] argued that, comparing to applying traditional multigrid schemes to the system of equations derived from KKT conditions, MG/OPT has
the following advantages:

- MG/OPT can deal with more general problems in an optimization perspective, in particular, it is able to handle inequality constraints in a natural way.

- MG/OPT has a better guarantees of convergence than using traditional multigrid for a system of equations.

- for a class of optimization problems governed by differential equations, multigrid will be better suited to the explicit optimization model rather than to the underlying differential equation when it is not elliptic.

In order to explore more on the suitability of MG/OPT, these authors also developed diagnostic tests [NL09] to automatically measure if the optimization model is proper for the multilevel algorithm. Basically, it is sufficient to examine the following four properties of the optimization problems: degree of the nonlinearity, complementarity of the optimization problems and the solver, consistency of the optimization problems across the levels, and separability of the reduced Hessians of the optimization problems across the levels.

Worth to be noticed about these diagnostic tests is that first, the computational costs are relatively cheap since most of the required information has been provided by the optimization process itself. Second, the tests identify relevant properties of the optimization problem so as to guide the user wisely in choosing the underlying components of the MG/OPT solver.

Later, Nash [Nas10] developed a framework analogous to the unconstrained version of MG/OPT to handle constrained optimization problem. The general formula for this kind of problem is:

\[
\begin{align*}
\min_{u_h} & \quad f_h(u_h) \\
\text{subject to} & \quad a_h(u_h) = 0 \\
& \quad c_h(u_h) \leq 0
\end{align*}
\]

The major differences from the unconstrained MG/OPT are as follows:
• In addition to shifting the original optimization problem on the coarse grid, MG/OPT algorithm requires to shift the constraints as well.

• The necessary line search is based on an augmented Lagrangian merit function

The global convergence and descent properties of this algorithm are also provided in this paper.

Inspired by MG/OPT, the idea of optimization-based multigrid has been widely applied to different areas. In [Haz08b], Hazra presents the application of an optimization-based multigrid method to state-constrained aerodynamic shape optimization problems. This work extends his previous paper [Haz08a] applying the same strategy to aerodynamic shape optimization problems without additional state constraints. One different feature of this approach is, instead of using the correction term \( \nabla f_H(I^H_h x_h) - I^H_h \nabla f_h(x_h) \) for the coarse grid subproblem as in MG/OPT, he chose \( I^H_h \nabla f_h(x_h) \) as a correction term for the coarse grid problem. And also, he added a correction term \( I^H_h \nabla c_h(x_h) \) to the additional coarse grid state constraint where \( \nabla c_h \) is the reduced gradient of the additional state constraint on the find grid. By choosing the underlying relaxation method for multigrid as simultaneous pseudo-timestepping, his algorithm resulted in a drastic improvement over the traditional gradient method with respect to the computation cost. The application of his optimization-based multigrid method reduced the number of iterations by more than 65% over that of single grid computations.

In [GST10], Gratton, Sartenaer and Toint introduced a recursive multiscale trust-region (RMTR) method for the solution of unconstrained nonlinear and possibly nonconvex discretized optimization problems. The main idea is, given an optimization problem, instead of directly solving an approximate quadratic model associated with a trust region radius as many practical trust-region algorithms do, this algorithm, within a fixed trust-region radius, provides a descent direction for the current quadratic model by utilizing multigrid-based optimization framework to solve an alternative linear model recursively which is cheaper than the corresponding quadratic model. They also proved global convergence of RMTR
to first-order critical points, that is convergence from arbitrary starting points to limit points satisfying the first-order optimality conditions. Comparing to MG/OPT, the difference of this approach in the coarse grid correction is, instead of using the correction term \( \nabla f_H(I^H_h x_h) - I^H_h \nabla f_h(x_h) \) for coarse grid subproblem as in MG/OPT, this approach chose \( g_H - I^H_h \nabla f_h(x_h) \) as a correction term where \( g_H \) is the gradient for the linear approximate model in the coarse grid and \( f_h \) is the original objective function. For the numerical results, they experimented on both simple quadratic examples and nonconvex examples using Truncated-Conjugate-Gradients as a smoothing technique, and choosing prolongation to be the linear interpolation operator and the restriction to be its transpose normalized to ensure that \( \| I^H_h \| = 1 \). As a result, RMTR converged much faster in terms of iteration numbers comparing to a classical trust-region method. Later, in order to extend RMTR to solve bound constrained optimization problems, Gratton, Mouffe, et al. [GMTWM07] modified RMTR to use the infinity norm to define the shape of the trust region instead using the Euclidean norm as in [GST10], which is well adapted to the handling of bounds, and also to the use of successive coordinate minimization as a smoothing technique.

In [VB08], Vallejos and Borzi discussed two different multigrid optimization schemes, the multigrid for optimization (MG/OPT) and the collective smoothing multigrid (CSMG), to solve elliptic linear and bilinear optimal control problems. The numerical results showed that both methods provide mesh independent and parameter independent convergence although CSMG is faster than MG/OPT. On the other hand, MG/OPT did not require any adaptation to the problem which is necessary for the application of CSMG.

In [WG07a], a line search multigrid method is presented based on MG/OPT for solving discretized versions of convex infinite dimensional optimization problems. Contrary to the necessary conditions to guarantee convergence of MG/OPT which is applying the smoothing step at each grid level, this new approach does not require it.
2.3 Centroidal Voronoi Tessellation

All the approaches related to MG/OPT we discussed so far have only been applied to classes of problems with regular geometric structure, and that can be reduced to unconstrained problems. However, to better understand the insight and potential of MG/OPT, the question is how it will perform on problems with a more complicated geometric hierarchy and even with constraints. Intent to delve deep on this consideration, we plan to study a class of vector quantization problems based on centroidal Voronoi tessellation (CVT).

A Voronoi diagram is defined as a map from the set of \( N \)-dimensional vectors in the domain \( \Omega \subset \mathbb{R}^N \) into a finite set of vectors \( \{z_i\}_{i=1}^K \) called generators. Given a distance function \( d(z,w) \), the Voronoi set \( V_i \) is the set of all elements belonging to \( \Omega \) that are closer to \( z_i \) than to any of the other elements \( z_j \) [Aur90, OBS92]:

\[
V_i = \{ w \in \Omega \mid d(w,z_i) < d(w,z_j), \ j = 1, \ldots, K, i \neq j \}.
\]

For a given density function \( \rho \) defined on \( \Omega \), we may define the centroids, or mass centers, of regions \( \{V_i\}_{i=1}^K \) by

\[
z^*_i = \frac{\int_{V_i} y \rho(y) \, dy}{\int_{V_i} \rho(y) \, dy}.
\]

A 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, \( z_i = z^*_i \) for all \( i \).

Given a set of points \( \{z_i\}_{i=1}^K \) and a tessellation \( \{V_i\}_{i=1}^K \) of the domain, we may define the energy function for the corresponding \( \{z_i\}_{i=1}^K \) as

\[
G(\{z_i\}_{i=1}^K) = \sum_{i=1}^K \int_{V_i} \rho(y) |y - z_i|^2 \, dy.
\]

The minimizer of \( G \) forms a CVT which illustrates the optimization property of the
CVT [DFG99]. This function appears in many engineering applications. For instance, it provides optimal least-squares vector quantizer design in electrical engineering problems.

The CVT concept also applies in many other areas such as astronomy, biology, image and data analysis, resource optimization, sensor networks, geometric design, and numerical partial differential equations [CMKB04, DFG99, DG02, DGJ02, DGJ03, DW02, DW06, HHD03, JDG02, MT04, WCL04, VC04]. In [DFG99], extensive reviews of the modern mathematical theory and diverse applications of CVTs are provided. The most widely used method for computing CVTs is the algorithm developed by Lloyd in the 1960s [Llo82]. Lloyd’s algorithm represents a fixed-point type 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 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 [Aur90, Dwy91, For87]. It was shown that Lloyd’s algorithm decreases the energy functional $G(\{z_i\}_{i=1}^{K})$ at every iteration, which gives a strong indication of its practical convergence. Despite its simplicity, proving convergence of Lloyd’s algorithm is not a trivial task. As shown in [DFG99], by measuring the preconditioned Hessian matrix for Lloyd’s map, the authors stated that Lloyd’s method has a linear convergence rate for strongly log-concave densities, $r \approx 1 - \frac{C}{K^2}$ where $C$ is a constant number and $K$ is the number of generators. On one hand, this analysis can provide a chance for us to explore more natural properties of CVT. On another hand, it can provide an alternative approach to measure the convergence ability of certain algorithm. Some recent work [DEJ06, EJR08] has substantiated earlier claims about global convergence of Lloyd’s algorithm, although single-point convergence for higher dimensions 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 [DGJ02, GN98, LBG80, SG86]. A particular extension using parallel and probabilistic sampling was given in [JDG02] 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 [DE06, LL10] and GPU extensions [RLW+11, VSCG08].

The idea of multigrid and multilevel implementation of Lloyd’s algorithm has been recently introduced by Yavneh et al. [KYS05]. They used a FAS implementation of the Lloyd-Max scheme method (Multigrid Lloyd-Max) based on minimizing the residual between generators and centroids. Although successful in the 1-dimensional setting, this approach was not generalizable to higher dimensions [KY06]. An alternative approach was introduced in [DE07], where multigrid method was used as an “outer” scheme, with Lloyd’s algorithm playing the role of a relaxation at each level (Multilevel-Lloyd). The method was rigorously shown to be uniformly convergent for all smooth perturbations of a constant density in [DE08] and was extended to the 2-dimensional setting in [Eme10], 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.

In [DEN12], we adapted the MG/OPT framework to the CVT context. In the 1D case, by choosing the appropriate transfer operators, MG/OPT outperforms other existing approaches with both uniform density and linear/nonlinear densities in the sense of significantly lower convergence factors which are independent of the problem size. This is exactly what we expected as typical multigrid efficiency. In particular, the advantage of MG/OPT
comparing to the Multilevel-Lloyd method comes in a form of significant reduction of com-
putational costs, while lower convergence factors and generalizability to higher dimensions
favorably distinguishes it from the Multigrid Lloyd-Max method.

In the 2D case, we chose the uniform density on a triangular domain, the number of
generators as the consecutive subset of the following sequence $[1, 10, 55, 253, 1181, \ldots]$, and
the initial guess is provided by slightly perturbing the local minimizer. The combination of
these settings possesses a certain beneficial feature that the coarse CVT nodes are already
a subset of finer CVT nodes, the rest of the finer nodes are basically every mid point of two
coarse generators which are Delaunay neighboring to each other. As a result, it provides us
a clear pattern to choose transform operators. Therefore, MG/OPT shows its superiority
over 1-level methods. Moreover, it maintains the typical multigrid efficiency similar to the
behavior observed in the 1-dimensional case.

It is obvious that this 2D implementation has major limitations. Due to the nature of
CVT, the stationary configuration is extremely sensitive to the choices of those fundamental
components such as number of generators, shape of domain and density. Hence, the exten-
sion to random configurations is a severe challenge, in the sense that trivial modifications
of those ingredients can result in a very irregular pattern of the corresponding hierarchies
and the transform operators will be extremely hard to formulate. We will show in Chapter
4 that at the end, we are able to overcome this limitation to some extent.
Chapter 3: Theoretical Analysis of CVT

3.1 Introduction

As the most traditional and popular method to construct CVTs, the Lloyd algorithm sparked enormous research efforts in recent years and has been studied in many contexts for different applications [DFG99]. Despite its great success in applications and a large number of studies over the last few decades, only limited theoretical results on the Lloyd algorithm have been obtained and many fundamental issues remain open concerning its convergence.

3.2 1D case

In this section, we will explore deeper the properties of Hessian matrix for 1D CVT problem, as well as the Jacobian of the Lloyd map.

Given a set of points \( \{x_i\}_{i=1}^K \) and a tessellation \( \{V_i\}_{i=1}^K \) of the domain, we may define the energy function for the corresponding \( \{x_i\}_{i=1}^K \) as

\[
\mathcal{G}(\{x_i\}_{i=1}^K) = \sum_{i=1}^{K} \int_{V_i} \rho(y)|y - x_i|^2 dy.
\]

The minimizer of \( \mathcal{G} \) forms a CVT which illustrates the optimization property of the CVT [DFG99]. Use \( x_i = \{x_i^{(k)}\} \) to denote the current generator where the \( k \)th component of \( x_i \) is \( x_i^{(k)} \), \( J_i \) to denote the indices of the Delaunay neighboring [A.15] generators to \( x_i \), \( W_{ij} \) to denote the common edge of Voronoi cells \( V_i \) and \( V_j \), \( f \) to denote the given distance function. According to Asami’s note [Asa91], the Hessian is derived as follows:
• If \( j \neq i \) and \( j \notin J_i \), then,

\[
\frac{\partial^2 G}{\partial x_i^{(k)} \partial x_j^{(\lambda)}} = 0.
\]

• If \( j \in J_i \), then,

\[
\frac{\partial^2 G}{\partial x_i^{(k)} \partial x_j^{(\lambda)}} = \int_{W_{ij}} \frac{2}{\|x_j - x_i\|} (x_i^{(k)} - x^{(k)})(x_j^{(\lambda)} - x^{(\lambda)}) f'(|x - x_i|^2) \rho(x) dx.
\]

• If \( j = i \), then,

\[
\frac{\partial^2 G}{\partial x_i^{(k)} \partial x_j^{(\lambda)}} = \int_{V_i} [2\delta(k, \lambda) f'(|x - x_i|^2) + 4(x_i^{(k)} - x^{(k)})(x_i^{(\lambda)} - x^{(\lambda)}) f''(|x - x_i|^2)] \rho(x) dx
\]

\[
- \sum_{j \in J_i} \int_{W_{ij}} \frac{2}{\|x_j - x_i\|} (x_i^{(k)} - x^{(k)})(x_i^{(\lambda)} - x^{(\lambda)}) f'(|x - x_i|^2) \rho(x) dx.
\]
Therefore, for the Hessian of the 1D CVT, we can obtain the following result:

\[
\begin{align*}
\frac{\partial^2 G}{\partial x_i \partial x_{i-1}} &= -\rho \left( \frac{x_{i-1} + x_i}{2} \right) \left( \frac{x_i - x_{i-1}}{2} \right), \quad i = 2 \ldots K - 1 \\
\frac{\partial^2 G}{\partial x_i \partial x_{i+1}} &= \rho \left( \frac{x_{i+1} + x_i}{2} \right) \left( \frac{x_i - x_{i+1}}{2} \right), \quad i = 2 \ldots K - 1 \\
\frac{\partial^2 G}{\partial x_i^2} &= \int_{x_{i-1} + x_i}^{x_{i+1} + x_i} 2\rho(y)\,dy + \rho \left( \frac{x_{i+1} + x_i}{2} \right) \left( \frac{x_i - x_{i+1}}{2} \right) \\
&\quad - \rho \left( \frac{x_{i-1} + x_i}{2} \right) \left( \frac{x_i - x_{i-1}}{2} \right), \quad i = 2 \ldots K - 1 \\
\frac{\partial^2 G}{\partial x_1^2} &= \int_{x_0}^{x_1} 2\rho(y)\,dy + \rho \left( \frac{x_1 + x_0}{2} \right) \left( \frac{x_1 - x_0}{2} \right) \\
\frac{\partial^2 G}{\partial x_K^2} &= \int_{x_{K-1} + x_K}^{x_K} 2\rho(y)\,dy - \rho \left( \frac{x_{K-1} + x_K}{2} \right) \left( \frac{x_K - x_{K-1}}{2} \right) \\
\end{align*}
\]

For the uniform density:

\[
\begin{align*}
\frac{\partial^2 G}{\partial x_i \partial x_{i-1}} &= -\frac{x_i - x_{i-1}}{2}, \quad i = 2 \ldots K - 1 \\
\frac{\partial^2 G}{\partial x_i \partial x_{i+1}} &= \frac{x_i - x_{i+1}}{2}, \quad i = 2 \ldots K - 1 \\
\frac{\partial^2 G}{\partial x_i^2} &= \frac{x_{i+1} - x_{i-1}}{2}, \quad i = 2 \ldots K - 1 \\
\frac{\partial^2 G}{\partial x_1^2} &= \frac{3x_1 + x_2}{2} \\
\frac{\partial^2 G}{\partial x_K^2} &= 2 - \frac{3x_K + x_{K-1}}{2}
\end{align*}
\]
Thus, at the minimum point:

\[
\nabla^2 G = \frac{1}{K} \begin{bmatrix}
\frac{3}{2} & -\frac{1}{2} & 0 & \ldots & 0 \\
-\frac{1}{2} & 1 & -\frac{1}{2} & \ldots & 0 \\
0 & -\frac{1}{2} & 1 & -\frac{1}{2} & \ldots \\
\ddots & \ddots & \ddots & \ddots & \ddots \\
0 & 0 & \ldots & -\frac{1}{2} & \frac{3}{2}
\end{bmatrix}
\]

**Theorem 3.2.1.** For the 1D CVT with the uniform density, the Hessian is positive definite.

**Proof.** First, \(\nabla^2 G\) is Hermitian. Given the inner tridiag\((-\frac{1}{2}, 1, -\frac{1}{2})\), \(\nabla^2 G\) is diagonally dominant. Furthermore, due to the fact that the two boundary rows are strictly diagonally dominant, \(\nabla^2 G\) is positive definite [HJ90]. □

### 3.2.1 Jacobian of the Lloyd map

Suggested by the above analysis of Hessian for 1D CVT, we can similarly analyze the performance of the Lloyd map in the 1D case. Denote the Jacobian of the Lloyd map \(T\) as \(dT\), \(M_i = \int_{V_i} \rho(x) dx\) and \(M = \text{diag}(M_1, M_2, \ldots, M_K)\). We have \(\nabla^2 G = 2M(I - dT)\) [DFG99].

For any smooth density, at the stationary points, we have \(\frac{\partial T_i}{\partial z_i} = \frac{\partial T_i}{\partial z_{i-1}} + \frac{\partial T_i}{\partial z_{i+1}}\), so

\[
dT = \frac{1}{4} M^{-1} A \quad \text{where } A \text{ is}
\]

\[
\begin{bmatrix}
(x_2 - x_1)\rho\left(\frac{x_1 + x_2}{2}\right) & (x_2 - x_1)\rho\left(\frac{x_1 + x_2}{2}\right) & 0 \\
(x_2 - x_1)\rho\left(\frac{x_1 + x_2}{2}\right) & (x_2 - x_1)\rho\left(\frac{x_1 + x_2}{2}\right) + (x_3 - x_2)\rho\left(\frac{x_3 + x_2}{2}\right) & (x_3 - x_2)\rho\left(\frac{x_3 + x_2}{2}\right) \\
\ddots & \ddots & \ddots & \ddots \\
0 & (x_K - x_{K-1})\rho\left(\frac{x_{K-1} + x_K}{2}\right) & (x_K - x_{K-1})\rho\left(\frac{x_{K-1} + x_K}{2}\right)
\end{bmatrix}
\]
Therefore, for the uniform density, $d\mathbf{T} = \begin{bmatrix} \frac{1}{4} & \frac{1}{4} & 0 \\ \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\ \vdots \\ 0 & \frac{1}{4} & \frac{1}{4} \end{bmatrix} = \frac{1}{4}A$

**Theorem 3.2.2.** In 1D, with the uniform density, the eigenvalues of $d\mathbf{T}$ are $\lambda(d\mathbf{T}) = \sin^2 \left( \frac{(2k+1)\pi}{2K+1} \right)$, $k = 0, \ldots, K - 1$ where $K$ is the number of generators.

**Proof.** Since $A$ is a symmetric, tridiagonal matrix, its eigenvalues are the zeros of $Q_K(x)$, where $Q_K(x)$ is a polynomial defined recursively by $Q_0(x) = 1$ and $Q_1(x) = 1 - x$,

$$Q_k(x) = (2 - x)Q_{k-1}(x) - Q_{k-2}(x), \quad k = 2, \ldots, K - 1,$$

$$Q_K(x) = (1 - x)Q_{K-1}(x) - Q_{K-2}(x).$$

If $U_k(x)$ is the second type of Chebyshev polynomial, then

$$Q_k(x) = U_k \left( 1 - \frac{x}{2} \right) - U_{k-1} \left( 1 - \frac{x}{2} \right), \quad k = 0, 1, \ldots, K - 1.$$ 

Thus, it is necessary to solve

$$Q_K(x) = (1 - x)(U_{K-1} \left( 1 - \frac{x}{2} \right) - U_{K-2} \left( 1 - \frac{x}{2} \right)) - (U_{K-2} \left( 1 - \frac{x}{2} \right) - U_{K-3} \left( 1 - \frac{x}{2} \right)).$$

$$= 0.$$
Since $U_{K+1}(x) = 2xU_K(x) - U_{K-1}(x)$, then

$$Q_K(x) = U_{K-1} \left(1 - \frac{x}{2}\right) - U_{K-2} \left(1 - \frac{x}{2}\right) - xU_{K-1} \left(1 - \frac{x}{2}\right)$$

$$+ xU_{K-2} \left(1 - \frac{x}{2}\right) - U_{K-2} \left(1 - \frac{x}{2}\right) + U_{K-3} \left(1 - \frac{x}{2}\right)$$

$$= U_{K-1} \left(1 - \frac{x}{2}\right) - U_{K-2} \left(1 - \frac{x}{2}\right) - \frac{1}{2} U_{K-1} \left(1 - \frac{x}{2}\right) - \frac{1}{2} U_K \left(1 - \frac{x}{2}\right)$$

$$+ \frac{1}{2} U_{K-1} \left(1 - \frac{x}{2}\right) + \frac{1}{2} U_{K-3} \left(1 - \frac{x}{2}\right) - U_{K-2} \left(1 - \frac{x}{2}\right) + U_{K-3} \left(1 - \frac{x}{2}\right)$$

$$= -\frac{1}{2} \left(U_K \left(1 - \frac{x}{2}\right) - U_{K-1} \left(1 - \frac{x}{2}\right)\right)$$

$$+ \left(U_{K-1} \left(1 - \frac{x}{2}\right) - U_{K-2} \left(1 - \frac{x}{2}\right)\right)$$

$$- \frac{3}{2} \left(U_{K-2} \left(1 - \frac{x}{2}\right) - U_{K-3} \left(1 - \frac{x}{2}\right)\right)$$

$$= 0. (*)$$

In order to find the zeros of equation $(*)$, it is equivalent to solve $U_K \left(1 - \frac{x}{2}\right) = U_{K-1} \left(1 - \frac{x}{2}\right)$, set $y = 1 - \frac{x}{2}$

Since the polynomials of the second kind satisfy $U_K(\cos(\theta)) = \frac{\sin((K+1)\theta)}{\sin \theta}$, we have

$$\sin((K+1) \arccos y) = \sin(K \arccos y)$$

$$(K+1) \arccos y = (2k + 1)\pi - K \arccos y$$

$$x = 2 - 2 \cos \left(\frac{(2k + 1)\pi}{2K+1}\right) = \lambda(A).$$
Finally, we obtain

\[
\lambda(dT) = \frac{1}{4} \lambda(A) = \frac{1}{2} - \frac{1}{2} \cos \left( \frac{(2k + 1)\pi}{2K + 1} \right) \\
= \sin^2 \left( \frac{(2k + 1)\pi}{2K + 1} \right), \ k = 0, \ldots, K - 1
\]

Corollary 3.2.3. For the uniform density on the unit interval, the Lloyd algorithm is globally convergent with a geometric convergence rate approximated by

\[
\|\lambda(dT)\| \approx 1 - \frac{\pi^2}{4K^2}
\]

Proof. Since \( \frac{(2k + 1)\pi}{2K + 1} \in \left[ \frac{\pi}{2K + 1}, \frac{(2K - 1)\pi}{2K + 1} \right] \subset [0, \frac{\pi}{2}] \),

\[
\|\lambda(dT)\| = \sin^2 \left( \frac{1}{2} - \frac{1}{2K + 1} \right) \pi = \cos^2 \left( \frac{\pi}{2K + 1} \right) = 1 - \sin^2 \left( \frac{\pi}{2K + 1} \right) \approx 1 - \frac{\pi^2}{4K^2}
\]

Comparing to the same result obtained in [DFG99] for an estimate of the convergence speed for 1D Lloyd map with uniform density. Our approach gives more rigorous result by providing exact eigenvalues of the corresponding Jacobian matrix. This approach seems to be extensible to nonuniform cases. However, the main difficulty arises from the fact that for nonuniform densities, the diagonal entries and the off-diagonal entries are no longer constant which gives more complications to find the solutions of the corresponding Chebyshev polynomials. Therefore, we leave this direction to our future work.

Next, we will do a similar analysis for the 2D case.
3.3 2D case

3.3.1 Uniform density

For the 2D CVT, given \( x_i^{(k)} \) as kth component of \( x_i \), the Hessian \( \nabla^2 G \) is as follows:

- If \( j \neq i \) and \( j \notin J_i \), then \( \frac{\partial^2 G}{\partial x_i^{(k)} \partial x_j^{(l)}} = 0 \).

- If \( j \in J_i \), then \( \frac{\partial^2 G}{\partial x_i^{(1)} \partial x_j^{(1)}} = \int_{W_{ij}} \frac{2}{\|x_j - x_i\|} (x_i^{(1)} - x_j^{(1)})(x_i^{(1)} - x_j^{(1)}) \rho(x) dx \).

- If \( j \in J_i \), then \( \frac{\partial^2 G}{\partial x_i^{(1)} \partial x_j^{(2)}} = \int_{W_{ij}} \frac{2}{\|x_j - x_i\|} (x_i^{(1)} - x_j^{(1)})(x_i^{(2)} - x_j^{(2)}) \rho(x) dx \).

- If \( j = i \), then \( \frac{\partial^2 G}{\partial x_i^{(1)} \partial x_i^{(1)}} = \int_{V_i} 2\rho(x) dx - \sum_{j \in J_i} \int_{W_{ij}} \frac{2}{\|x_j - x_i\|} (x_i^{(1)} - x_j^{(1)})^2 \rho(x) dx \).

- If \( j = i \), then \( \frac{\partial^2 G}{\partial x_i^{(1)} \partial x_i^{(2)}} = -\sum_{j \in J_i} \int_{W_{ij}} \frac{2}{\|x_j - x_i\|} (x_i^{(1)} - x_j^{(1)})(x_i^{(2)} - x_j^{(2)}) \rho(x) dx \).

We now quote some elementary conclusions at the stationary points with the uniform density.

According to Gersho’s conjecture: Let \( \rho \) be uniform density defined on a smooth bounded domain \( \Omega \). On a 2D plane at the stationary points, for large \( K \), we can obtain the following facts as well as demonstrated in Figure 3.1:

- Each Voronoi cell is a regular hexagon.

- Denote the Cartesian coordinates of \( x_i \) by \([x, y]\). Then we have the system to represent each element related to \( x_i \).

- \( |W_1| = |W_2| = |W_3| = |W_4| = |W_5| = |W_6| \).
Figure 3.1: representation of regular hexagon in Cartesian coordinate system, $r$ is the radius of the circumscribed circle of the hexagon enclosed by $[x_{j1}, \ldots, x_{j6}]$, $r_1$ is the radius of the circumscribed circle of the hexagon enclosed by the neighboring generators to $x_i$ labeled as $^*$.

**Lemma 3.3.1.** According to Figure 3.1, we have

\[
\int_{W_1} \frac{2}{\|x_j - x_i\|} (x^{(1)}_i - x^{(1)})(x^{(2)}_j - x^{(2)}) dx = \int_{W_4} \frac{2}{\|x_j - x_i\|} (x^{(1)}_i - x^{(1)})(x^{(2)}_j - x^{(2)}) dx,
\]

\[
\int_{W_3} \frac{2}{\|x_j - x_i\|} (x^{(1)}_i - x^{(1)})(x^{(2)}_j - x^{(2)}) dx = \int_{W_5} \frac{2}{\|x_j - x_i\|} (x^{(1)}_i - x^{(1)})(x^{(2)}_j - x^{(2)}) dx,
\]

\[
\int_{W_2} \frac{2}{\|x_j - x_i\|} (x^{(1)}_i - x^{(1)})(x^{(2)}_j - x^{(2)}) dx = \int_{W_6} \frac{2}{\|x_j - x_i\|} (x^{(1)}_i - x^{(1)})(x^{(2)}_j - x^{(2)}) dx.
\]

Furthermore,

\[
\int_{W_2} \frac{2}{\|x_j - x_i\|} (x^{(1)}_i - x^{(1)})(x^{(2)}_j - x^{(2)}) dx = 0.
\]
Proof. First, we show that

$$\int_{W_2} \frac{2}{\|x_j - x_i\|} (x_i^{(1)} - x^{(1)}_1)(x_i^{(2)} - x^{(2)}_2) dx = \int_{W_5} \frac{2}{\|x_j - x_i\|} (x_i^{(1)} - x^{(1)}_1)(x_i^{(2)} - x^{(2)}_2) dx = 0.$$ 

By parametrization,

$$\int_{W_2} \frac{2}{\|x_j - x_i\|} (x_i^{(1)} - x^{(1)}_1)(x_i^{(2)} - x^{(2)}_2) dx$$

$$= \frac{2}{\sqrt{3}} \int_0^1 (x_i^{(1)} - x^{(1)}_{j1} - (x_{j2}^{(1)} - x^{(1)}_{j1})t)(x_i^{(2)} - x^{(2)}_{j1} - (x_{j2}^{(2)} - x^{(2)}_{j1})t) dt$$

$$= \frac{2}{\sqrt{3}} \left( (x_i^{(1)} - x^{(1)}_{j1})(x_i^{(2)} - x^{(2)}_{j1}) - \frac{(x_i^{(1)} - x^{(1)}_{j1})(x_i^{(2)} - x^{(2)}_{j1})}{2} \right. - \frac{(x_i^{(2)} - x^{(2)}_{j1})(x_i^{(1)} - x^{(1)}_{j1})}{2} + \frac{(x_i^{(2)} - x^{(2)}_{j1})(x_i^{(1)} - x^{(1)}_{j1})}{3} \right) \tag{3.1a}$$

Notice from Figure 3.1, given the coordinate of \( x_i \) as \((x, y)\), then \( x_{j1} = (x + r \sin \left( \frac{\pi}{3} \right), y + r \cos \left( \frac{\pi}{3} \right)) \), \( x_{j2} = (x + r \sin \left( \frac{\pi}{3} \right), y - r \cos \left( \frac{\pi}{3} \right)) \). Therefore, substituting in (3.1a), we will have

$$\int_{W_2} \frac{2}{\|x_j - x_i\|} (x_i^{(1)} - x^{(1)}_1)(x_i^{(2)} - x^{(2)}_2) dx = 0.$$ 

Similarly, we can prove the rest of the equations. \(\square\)

**Lemma 3.3.2.** With the uniform density on a 2D plane at the stationary points, for large \( K \), we have \( \frac{\partial^2 G}{\partial \phi^{(1)}_i \partial x^{(2)}_i} = 0. \)
Proof. According to Lemma 3.3.1, we have

\[
\frac{\partial^2 G}{\partial x_i^{(1)} \partial x_i^{(2)}} = - \sum_{j \in J} \int_{W_{ij}} \frac{2}{\|x_j - x_i\|} (x_i^{(1)} - x^{(1)})(x_i^{(2)} - x^{(2)}) dx
\]

\[
= -2 \left( \int_{W_2} \frac{2}{\|x_j - x_i\|} (x_i^{(1)} - x^{(1)})(x_i^{(2)} - x^{(2)}) dx 
+ \int_{W_3} \frac{2}{\|x_j - x_i\|} (x_i^{(1)} - x^{(1)})(x_i^{(2)} - x^{(2)}) dx \right).
\]

Since \[
\int_{W_2} \frac{2}{\|x_j - x_i\|} (x_i^{(1)} - x^{(1)})(x_i^{(2)} - x^{(2)}) dx = - \int_{W_3} \frac{2}{\|x_j - x_i\|} (x_i^{(1)} - x^{(1)})(x_i^{(2)} - x^{(2)}) dx,
\]

\[
\frac{\partial^2 G}{\partial x_i^{(1)} \partial x_i^{(2)}} = 0.
\]

\[\square\]

Theorem 3.3.3. In the 2D CVT with uniform density, the Hessian \(\nabla^2 G\) is positive definite.

Proof. According to Lemma 3.3.2, denote two boundary points of edge \(W\) as \(x_-\) and \(x_+\).
following a clockwise direction. Follow the notations given in Figure 3.1, we have

\[
\frac{\partial^2 G}{\partial x_i(1) \partial x_i(1)} - \frac{\partial^2 G}{\partial x_i(1) \partial x_i(2)} - \sum_{j \in J_i} \left| \frac{\partial^2 G}{\partial x_i(1) \partial x_j(1)} \right| - \sum_{j \in J_i} \left| \frac{\partial^2 G}{\partial x_i(1) \partial x_j(2)} \right|
\]

\[
= 2 \int_{V_i} dx - \sum_{j \in J_i} \left( \int_{W_{ij}} \frac{2}{r_1} (x_i(1) - x(1))^2 dx + \int_{W_{ij}} \frac{2}{r_1} (x_i(1) - x(1))(x_j(1) - x(1)) dx \right)
\]

\[
- 4 \left| \int_{W_{ij}} \frac{2}{r_1} (x_i(1) - x(1))(x_j(2) - x(2)) dx \right|
\]

\[
= 2M_i - 2A_i \sum_{t \in W_{it}} \left( \frac{1}{6}(x_{jt} - x_{j(t+1)})^2 \right) - \frac{5}{3} r^2
\]

\[
= 2 \left( \frac{2\sqrt{3}}{2} r^2 - \frac{1}{\sqrt{3}} (x_+ - x_-)^2 \right) - \frac{5}{3} r^2
\]

\[
\geq 2 \left( \frac{2\sqrt{3}}{2} r^2 - \frac{1}{\sqrt{3}} (x_+ - x_-)^2 - \frac{1}{\sqrt{3}} (y_+ - y_-)^2 \right) - \frac{5}{3} r^2
\]

\[
= 3\sqrt{3} r^2 - \frac{2}{\sqrt{3}} r^2 - \frac{5}{3} r^2
\]

\[
= \frac{7\sqrt{3} - 5}{3} r^2
\]

\[
> 0
\]

Therefore, \( \nabla^2 G \) is strictly diagonally dominant. Since \( \nabla^2 G \) is also Hermitian, we can conclude that \( \nabla^2 G \) is positive definite. \( \square \)

**Corollary 3.3.4.** For the uniform density on a 2D plane, the Lloyd algorithm is locally convergent.

**Proof.** Recall that \( \nabla^2 G = 2M(I - dT) \). Since \( \nabla^2 G \) is positive definite, \( M(I - dT) \) is also positive definite. So the eigenvalues of

\[
M^{1/2}(I - dT)M^{-1/2} = I - M^{1/2}dT M^{-1/2}
\]

are all positive, which implies that \( \lambda(dT) < 1 \), a necessary condition for \( T \) to be a local contraction. \( \square \)
Chapter 4: Computational Experiments on CVTs

4.1 Introduction

As our analysis on Jacobian of Lloyd map shows from the last chapter, the convergence rate of the Lloyd map will be greatly slowed down as the number of generators is increased. Therefore, it is crucial to have fast and memory-efficient algorithms for computing the CVTs. In this sense, we are now ready to apply MG/OPT to construct CVT.

Before we show the practical implementation of this application, we should first point out several difficulties of this topic. Clearly, the overall performance of MG/OPT for CVT problems is atypical of that of other applications of MG/OPT such as PDE-constrained optimization problems. The PDE-constrained problems share a common property that their numerical solutions are based on regularly discretized grids, and this characteristic separates CVT from them since there is no such concepts as discretization and grids in the CVT structures. As a result, this difference gives rise to several challenges when we apply MG/OPT to construct CVTs:

First and most importantly, for CVT problems, in order to show the benefit of MG/OPT, it is necessary to maintain a fixed relation between generators during the whole process, in the sense that, if two generators are Delaunay neighboring to each other at the current iteration, they can not move too far away from each other at the next iteration, otherwise, the error will remain oscillatory along the relaxation which is not ideal in the multigrid context. This observation is demonstrated in the following 2D CVT experiments. The configuration is chosen as an equilateral triangle domain and uniform density with 253 generators. In each following figure, the error of the solution is illustrated separately as $x$-component and $y$-component. Figure 4.1 shows the initial error of solutions for CVT, the
initial guess is randomly picked. During the process of OPT, there are global movements for several generators; as a result the relative relations between generators are continuously changed. Accordingly, the error is not smoothed until convergence as shown in Figure 4.2. On the contrary, if we choose the initial guess as slight perturbation of the exact solution where the initial error is shown in Figure 4.3, the main advantage of this choice is that every generator in the process of OPT will only move locally around its previous location. In this case, a well smoothed error is obtained as shown in Figure 4.4.

Figure 4.1: initial error with a random initial guess
Figure 4.2: final error from OPT with a random initial guess

Figure 4.3: initial error with a special initial guess slightly perturbed from exact solution
It is easy to check that this phenomenon is caused by the nature of TN and it happens to every dimension of CVT. In the 1D case, the preservation of the order of the generators is easy to force simply by sorting the variables in ascending order. However, it is not trivial in 2D and higher dimensional cases.

Second, due to the irregular underlying geometry of higher dimensional CVTs with random configurations (random density, random boundary and random number of generators), there is no clear way to choose the fundamental components for MG/OPT such as the downdate operator and the update operator. Especially for the higher dimensional CVTs, unlike 1D case and problems with discretizations of PDEs, the coarse models are generally not guaranteed to be subsets of the fine model. Therefore, in the following sections, we will illustrate specific remedies for these challenges with different scenarios.
4.2 Application of MG/OPT to 1D CVT

In this section, we will show how to apply MG/OPT for 1D CVT. To make sure we are dealing with the same underlying problem on different levels, we scaled the objective function value by $K^2$ where $K$ is the current number of generators, so that the optimal energy value would be the 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 the 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 MG/OPT for solving the CVT problem, we also need to specify the appropriate choices of the operators and algorithm parameters. It turns out 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 $v_h$ on the fine level, the downdate operator $I_H^h$ samples $v_h$ 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 $\nabla G$ 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 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 $I_h^H = 2(FW)^T$ [TOS01, p. 61], 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 \( I^h_H = (H^H)^T \), a property that we rely on in our implementation.

For a \( V_{1,1} \) cycle implementation, one pre-smoothing and one post-smoothing relaxation are 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. Both OPT and MG/OPT were terminated due to saturation of the energy gradient \(|\mathcal{G}(z) - \mathcal{G}(z^*)| \leq 10^{-8}\), where \( z^* \) is the solution to the CVT problem. The estimation of the exact solution \( z^* \) was obtained by running OPT with a tight convergence tolerance.

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 \( \{z_i\}_{i=1}^K \) assures that the objective function is not affected by this modification.

1-dimensional examples

All 1D examples have been computed on the \([0, 1]\) domain, the objective function \( \mathcal{G} \) 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 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 [DEJ06]), 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. We will use this test quite often for other examples as well.

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 4.5a). Next we use the density $\rho(y) = 6y^2e^{-2y^3}$. The results are in Figure 4.5b. 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:

$$C = \left( \frac{|G(z^{k+1}) - G(z^*)|}{|G(z^1) - G(z^*)|} \right)^\frac{1}{k+1},$$

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

![Figure 4.5: 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](image)

In Figure 4.6 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 number of cycles, convergence factor and elapsed time are given in Table 4.1. Both MG/OPT and OPT algorithms have been stopped when the residual reaches a tolerance threshold of $10^{-8}$.

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

<table>
<thead>
<tr>
<th>k</th>
<th>MG/OPT</th>
<th>OPT</th>
<th>number of cycles</th>
<th>time(sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>0.0234</td>
<td>0.1803</td>
<td>4</td>
<td>18</td>
</tr>
<tr>
<td>32</td>
<td>0.0440</td>
<td>0.2094</td>
<td>5</td>
<td>20</td>
</tr>
<tr>
<td>64</td>
<td>0.0952</td>
<td>0.2897</td>
<td>6</td>
<td>26</td>
</tr>
<tr>
<td>128</td>
<td>0.0604</td>
<td>0.4839</td>
<td>5</td>
<td>43</td>
</tr>
<tr>
<td>256</td>
<td>0.1174</td>
<td>0.6474</td>
<td>6</td>
<td>69</td>
</tr>
<tr>
<td>512</td>
<td>0.0932</td>
<td>0.8129</td>
<td>6</td>
<td>150</td>
</tr>
<tr>
<td>1024</td>
<td>0.0796</td>
<td>0.9076</td>
<td>6</td>
<td>330</td>
</tr>
<tr>
<td>2048</td>
<td>0.0728</td>
<td>0.9403</td>
<td>6</td>
<td>500</td>
</tr>
<tr>
<td>4096</td>
<td>0.0757</td>
<td>0.9679</td>
<td>6</td>
<td>970</td>
</tr>
</tbody>
</table>

Figure 4.6: 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
Since the CVT algorithms in general show sensitivity to the choice of the initial configuration, we ran several tests with an intentionally "bad" choices of the initial guess, with all generators clustered near the origin in the [0, 1] region. As Figure 4.7 demonstrates, no significant difference in performance has been noted.

Figure 4.7: 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.

Figure 4.8 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 $\nabla G$ (which is related to the difference between the positions of generators in the approximate 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 $G$, 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 based 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 4.8, 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 4.8: Performance of MG/OPT with different stopping criteria ($\rho(y) = 1$). Blue-Dot: $|\mathcal{G}(z) - \mathcal{G}(z^*)| \leq 10^{-8}$; Red-Circle: $|\mathcal{G}(z) - \mathcal{G}(z^*)| \leq 10^{-11}$; Green-Square: $|\nabla \mathcal{G}|_{\infty} \leq 10^{-6}$.

Figure 4.9 demonstrates the performance of MG/OPT when applied to problems with different density functions. In particular we consider $\rho_1(y) = 1 + 0.1y$ and $\rho_2(y) = 6y^2e^{-2y^3}$.

In the left plot, we can see that the iteration count for $\rho_1$ stays constant with the increase of the problem size, and increases only slightly for $\rho_2$. For both densities the convergence factor exhibits insensitivity to the problem size, although the convergence factor is slightly larger for $\rho_2$, while remaining bounded from above by 0.2.

Figure 4.10 compares the performance of MG/OPT to that of the multilevel optimization based method introduced in [DE08]. Although the latter was also based on the idea of formulating an optimization problem based on the CVT energy $\mathcal{G}$, 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. 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 4.2 compares the performance of MG/OPT with that of the OPT algorithm and the Multigrid Lloyd-Max (MLM) algorithm introduced in [KYS03], 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 [KYS03] has been chosen for fair comparison. The column MG[KYS03] shows the convergence factors reported in [KYS03], where the
Table 4.2: Comparing performance of MG/OPT with 1-level optimization (OPT) and the Multilevel Lloyd-Max (MLM) method of [KYS03] for a nonlinear density: $y = 6x^2e^{-2x^3}$.

<table>
<thead>
<tr>
<th>$k$</th>
<th>MG/OPT$^1$</th>
<th>MG/OPT$^2$</th>
<th>MLM[KYS03]</th>
<th>OPT</th>
<th>MG/OPT$^1$</th>
<th>OPT</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>0.0350</td>
<td>0.0023</td>
<td>0.1725</td>
<td>0.8060</td>
<td>4</td>
<td>108</td>
</tr>
<tr>
<td>32</td>
<td>0.0314</td>
<td>0.0177</td>
<td>0.1782</td>
<td>0.9024</td>
<td>5</td>
<td>228</td>
</tr>
<tr>
<td>64</td>
<td>0.0673</td>
<td>0.0551</td>
<td>0.1847</td>
<td>0.9381</td>
<td>5</td>
<td>354</td>
</tr>
<tr>
<td>128</td>
<td>0.1120</td>
<td>0.0673</td>
<td>0.1962</td>
<td>0.9736</td>
<td>7</td>
<td>846</td>
</tr>
<tr>
<td>256</td>
<td>0.0980</td>
<td>0.0983</td>
<td>———</td>
<td>0.9877</td>
<td>7</td>
<td>1818</td>
</tr>
<tr>
<td>512</td>
<td>0.1230</td>
<td>0.1927</td>
<td>———</td>
<td>0.9943</td>
<td>7</td>
<td>4276</td>
</tr>
<tr>
<td>1024</td>
<td>0.1105</td>
<td>0.1504</td>
<td>———</td>
<td>0.9971</td>
<td>7</td>
<td>8388</td>
</tr>
<tr>
<td>2048</td>
<td>0.1253</td>
<td>0.2890</td>
<td>———</td>
<td>0.9983</td>
<td>7</td>
<td>14297</td>
</tr>
</tbody>
</table>

convergence factors were computed using

$$C = \frac{|G(z^{k+1}) - \tilde{G}(z^*)|}{|G(z^k) - \tilde{G}(z^*)|}$$  \hfill (4.2)

The column MG/OPT$^1$ lists the convergence factors for our algorithm MG/OPT computed according to (4.2). MG/OPT$^2$ uses the geometric mean convergence factor (4.1); MLM[KYS03] is the result from [KYS03] with the same convergence factor formula as MG/OPT$^1$; ‘——’ refers to data not provided in [KYS03].
4.2.1 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 [NN91]. 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 4.3.

<table>
<thead>
<tr>
<th>Operations</th>
<th>per coordinate</th>
<th>on the fine level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Downdate(variable)</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Downdate(residual)</td>
<td>3</td>
<td>$3K/2$</td>
</tr>
<tr>
<td>Search Direction</td>
<td>1</td>
<td>$K/2$</td>
</tr>
<tr>
<td>Update</td>
<td>3</td>
<td>$3K/2$</td>
</tr>
<tr>
<td>Line search</td>
<td>1</td>
<td>$K$</td>
</tr>
</tbody>
</table>

Table 4.3: The complexity of various parts of MG/OPT measured in floating point operations.

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 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.

4.3 Apply MG/OPT to 2D CVT with special configuration

In this case, we chose different transform operators with 1D CVT since the geometric CVT structures are different. The number of generators we focus on is a subset of the triangular number $[1, 10, 55, 253, 1181, \ldots]$ denoted as $\hat{T}$. Specifically, if choose the number of generator for coarse level as $\hat{T}_k$ in $\hat{T}$, the number of generators for finer level as $\hat{T}_{k+1}$, then the coarse CVT nodes are already a subset of finer CVT nodes, the rest of the finer nodes are basically every mid point of two coarse generators which are Delaunay neighboring to each
other. The corresponding hierarchy is illustrated as Figure 4.11.

Figure 4.11: hierarchy of CVT with number of generators as a subset of triangular number: $[1, 10]$, $\lambda$ is the weight of the fine-to-coarse node, this pattern can be extended to larger set

In our 2D experiments, the update operator is motivated by choosing the initial configurations which are close to exact solution

$$
[I_H^h v_H]^i = \begin{cases} 
4v_H^i & \text{if } i \in C; \\
2v_H^{j_1} + 2v_H^{j_2} & \text{otherwise}.
\end{cases}
$$

where $C$ is the subset of index of fine generators which corresponds to the coarse generators, $j_1$ & $j_2$ are the index of two Delaunay neighboring coarse generators corresponding to $ith$ fine generators.
Similarly as in the 1D CVT, transfer of the solution from a finer to a coarser grid is given by an injection which is defined as follows:

\[
[I_h^H v_h]^i = v_h^i, \; i \in C
\]

where \( C \) is defined as above.

The corresponding downdate operator \( \hat{I}_h^H \) for the gradient is given by:

\[
\hat{I}_h^H = \frac{1}{4} (I_h^H)^T
\]

**2D examples**

Below are some preliminary results that have been obtained in the 2-dimensional case for the case of a triangular domain.

In Figure 4.12, we give convergence results for MG/OPT and its competitors when the initial guess is taken to be relatively close to the local minimizer \(^1\). It is clear that MG/OPT maintains its superiority over OPT and Lloyd 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 4.4 provides numerical data for the performance of all three of these methods. Again, there is a clear advantage in using MG/OPT in this case. It has also been noted that the tendency of moving points outside of the domain typical for the regular 1-level OPT method is significantly reduced in the 2-dimensional MG/OPT implementation.

\(^1\)The result listed in the table is based on a large perturbation of the solution
Figure 4.12: Comparison of performance of MG/OPT, OPT and Lloyd for the case of a triangular 2-d region with $\rho = 1$. Red-Circle: OPT, Blue-Square: MG/OPT.

<table>
<thead>
<tr>
<th>$K$</th>
<th>MG/OPT</th>
<th>OPT</th>
<th>Lloyd</th>
<th>MG/OPT</th>
<th>OPT</th>
<th>Lloyd</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>0.0127</td>
<td>0.0871</td>
<td>0.8323</td>
<td>2</td>
<td>6</td>
<td>54</td>
</tr>
<tr>
<td>55</td>
<td>0.0231</td>
<td>0.1378</td>
<td>0.9554</td>
<td>4</td>
<td>10</td>
<td>244</td>
</tr>
<tr>
<td>253</td>
<td>0.0121</td>
<td>0.1957</td>
<td>0.9891</td>
<td>4</td>
<td>14</td>
<td>1143</td>
</tr>
<tr>
<td>1081</td>
<td>0.0092</td>
<td>0.4055</td>
<td>0.9973</td>
<td>4</td>
<td>28</td>
<td>5265</td>
</tr>
</tbody>
</table>

Table 4.4: Comparing performance of 2-dimensional MG/OPT with 1-level optimization (OPT) and Lloyd method for constant density in triangular domain.

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 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.

In next section, we will extend multigrid idea to improve initial configuration for 2D CVT with more general number of generators. Consequently, we are able to achieve multigrid efficiency by using MG/OPT to this case.

4.3.1 Application of MG/OPT to 2D CVT with number of generators as triangular numbers

In the last section, we illustrated how to apply MG/OPT to 1D CVT with a random configuration as well as 2D CVT with a special number of generators. In this section, we
will continue to explore the potential of MG/OPT for 2D CVT with a broader choice of the number of generators. Still with an equilateral triangle domain, we will focus on the **triangular numbers** $[A.22]$ as number of generators for the experimental purpose shown as Figure 4.13.

![Figure 4.13: demonstration of triangular numbers forming equilateral triangles [Wik11]](image)

If we choose the number of generators at each level as a triangular number, a hierarchy of CVT at two successive level can be obtained which is illustrated as Figure 4.14. Basically, the CVT node at the coarse level is closely approximated by the center of three Delaunay neighboring nodes from the fine level CVT.

This observation suggests that coarse level solutions can be used to compute an improved initial guess for the fine level 2D CVT construction with a triangular number of generators. Therefore, we consider a special multilevel-like nested iteration to do this job. Here is the description of this method labeled as “REFINE”:

**Algorithm of REFINE:**

Given a random initial estimate of the solution $u^0_H$ on the coarsest level. Select a non-negative integer $k$. Set

$$u_h \leftarrow \text{REFINE}(f_H, u^0_H)$$

where the function REFINE is defined as follows:
Figure 4.14: Hierarchy of CVT with 3 generators at coarse level, 6 generators at fine level, and $\lambda$ is the weight of coarse node to Delaunay neighboring fine node

- **Coarsest-level solve:** exactly solve the optimization problem:

\[ u_H \leftarrow \text{OPT}(f_H, u^0_H). \]

- Update solution to the finer level: $u^0_h = I^h_H u_H$

  - If on the finest level,

    \[ u_h \leftarrow \text{OPT}(f_h, u^0_h). \]

  - Otherwise,

    \[ u_h \leftarrow \text{OPT}(f_h, u^0_h, k). \]

In our implementation, we choose $k = 2$, and the update operator to transfer solutions
from coarser level to finer level as follows:

\[
[I_h^H v_H]^i = \frac{1}{3} v_{H}^{j_1} + \frac{1}{3} v_{H}^{j_2} + \frac{1}{3} v_{H}^{j_3}
\]

where \(j_1, j_2, j_3\) are the indexes of three Delaunay neighboring coarse generators corresponding to \(i\)th fine generator.

Table 4.5 provides numerical data for the performance of REFINE compared to OPT. It clearly shows that using REFINE in this case gives a much lower convergence factor than OPT, as well as lower time consumed for both methods to converge. Furthermore, REFINE shows a tendency to give a constant convergence factor, while the convergence factors from OPT grow much faster and approach 1. In Figure 4.15, another advantage of using REFINE is shown: that a better optimal solution can be achieved as the number of generators are increased.

![Figure 4.15: Optimal function value obtained by OPT minus optimal function value obtained by REFINE: \(f_{opt}^* - f_{REFINE}^*\)](image)
Table 4.5: Comparing performance of 2-dimensional REFINE with 1-level optimization (OPT) for constant density in triangular domain.

Next, inspired by the performance of REFINE, we will employ the same idea of constructing transform operators to the application of MG/OPT.

We first apply REFINE to randomly generated initial configuration and use the solution updated from the second finest level as the initial guess for MG/OPT. Then, transfer of the solution from the finer to coarser level is motivated as follows:

\[
[I_h^H v_h]^i = \frac{1}{3}v_{j_1}^h + \frac{1}{3}v_{j_2}^h + \frac{1}{3}v_{j_3}^h
\]

where \(j_1, j_2, j_3\) are the indexes of three Delaunay neighboring fine generators corresponding to \(i\)th coarse generator. Accordingly, the downdate operator \(\hat{I}_h^H\) for the gradient is given by

\[
\hat{I}_h^H = I_h^H
\]

the update operator is chosen as

\[
I_H^h = c(I_h^H)^T
\]  \hspace{1cm} (4.3)

where \(c\) is a constant number.
At this point, in order to determine an optimal value for $c$ in equation (4.3), we would like to introduce a very useful tool to detect if the search direction provided by recursion in MG/OPT is close to optimal. Denote the error on the fine level at cycle $j$ as $e_h^* = u_h^* - u_h^j$, the interpolated coarse level search direction $e_h = I_H^h e_H$. The basic idea is to compare $e_h^*$ with $e_h$. If they match each other, we can say that the search direction provided by recursion is close to optimal, therefore it is acceptable. As shown in Figure 4.16, it is clear that $e_h$ is matching very well with $e_h^*$ from both shape and size scale.

Figure 4.16: MG/OPT search direction versus error: dashed line: interpolated coarse level search direction $e_h$; solid line: error on fine level $e_h^*$. 

Back to our problem, if we choose $c = 2$ in equation (4.3), the search direction comparison is shown in Figure 4.17. It is obvious that there is a scaling issue between step to solution and search direction. Instead, if we choose $c = \frac{3}{2}$, the comparison is shown in Figure 4.18 where the step to solution is perfectly matching search direction. As a result,
provided that $c = \frac{3}{2}$, we apply MG/OPT to different number of generators on the finest level, the performance of MG/OPT is demonstrated in Figure 4.19. This result well exposes the multigrid efficiency at this situation with constant number of cycles, and a low and flat convergence factor, while the number of generators is increased.

Figure 4.17: wrong scale between step to solution and search direction
Figure 4.18: the search direction matches the step to solution perfectly
Figure 4.19: performance of MG/OPT with triangle number of generators
4.4 Summary for MG/OPT-based CVT solutions

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. The main advantage of the new method is its superior convergence speed when compared to other existing approaches. The fact that the convergence factors are on average much lower than those of other computing methods makes this formulation attractive when it comes to solving large CVT problems.

The next question is how to apply this new scheme to real life problems, focusing on the 2D case due to its practicality. Currently, we have only applied MG/OPT to the 2D case with a limited configuration, in the sense that the implementation requires an equilateral triangle domain with a special number of generators. However, through the following simple analysis, we demonstrate that this new technique is applicable more broadly.

Let us consider a simple application of 2D CVTs with a rectangle domain $R_s$ which is common in CVT applications. In order to employ the well designed implementation introduced in the previous sections to this situation and still obtain the multigrid efficiency, the basic idea is to first cover $R_s$ by a reasonable number of equilateral triangles, and then analyze the portion of region remained uncovered. For this uncovered part, we will use single-level OPT to do the optimization. Since our choice TN for OPT is also a powerful tool to solve large-scaled nonlinear optimization problems, we expect the overall performance to still be good.
The first covering strategy is illustrated as Figure 4.20. Assume the width of the given rectangle $R_s$ is $a$ and the length is $\frac{2\sqrt{3}}{3}a$, then the uncovered area is $A_u = 0.19a^2$ while the total area is $A = 1.15a^2$. Therefore, the coverage percentage is 84%.
The second scheme is to decompose $R_s$ to the following subtriangles $A_1, A_2, A_3, A_4, I$ as shown in Figure 4.21, where $A_1, A_2, A_3, A_4$ are equilateral triangles and $I$ is an isosceles triangle. Through a similar calculation as the first scheme, we can get a better covering percentage as 95%. Even better, if we further break $I$ into smaller triangles as $I_1, A_{11}, I_1$ in Figure 4.22, the total covering percentage of $R_s$ by equilateral triangles can be 89%. Therefore, if the whole given domain is large, then it is possible to keep doing this process until the final covering percentage is close to 1.

![Figure 4.22: further decomposition of $I$ into $I_1, A_{11}$ and $I_1$](image)

Now, let us analyze a specific case to illustrate this idea. Still given $R_s$ as the domain for a 2D CVT problem with the uniform density, we apply the second scheme to decompose the whole area into the following subtriangles $A_1, A_2, A_3, A_4$ and $I$ as shown in Figure 4.21, where $A_1, A_2, A_3, A_4$ are equilateral triangles and $I$ is an isosceles triangle. Suppose we need to construct a CVT with 9319 generators on $R_s$. If we allocate generators proportional to the area of $A_1, A_2, A_3, A_4, I$ and keep the number of generators distributed to each equilateral triangle as a triangular number, the number of generators that should be assigned to each triangle is calculated and shown in table 4.6. Then, for $A_1, A_2, A_3, A_4$, we apply MG/OPT to construct the local CVTs. For $I$, we apply OPT (TN) to construct the corresponding
local CVTs. Next, we patch up these local solutions as an initial guess for OPT to find the final solution on $R_s$, and compare with the performance of OPT using random initial guess. Label the step to provide a special initial guess to the whole domain as “PMG”. In Figure 4.23, the red line represents the performance of OPT with special initial guess obtained by PMG, and the blue line represents the performance of OPT with random initial guess. It clearly shows that PMG is able to provide a much better initial guess so that the convergence speed is much faster than with the random initial guess. More specific, the convergence factor of OPT using the initial guess provided by PMG is 0.1369 while the convergence factor of OPT with random initial guess is 0.8580.

Figure 4.23: Compare the performance of OPT two different initial guess. The red line represents the performance of OPT using the special initial guess obtained by PMG, and the blue line represents the performance of OPT using random initial guess.
So far, we have focused on the uniform density in 2D CVT. Applying MG/OPT to nonuniform densities is also attractive and practical in CVT applications. However, this direction is still under exploration. One challenge is how to choose the fixed boundary points following the given density since the distribution with nonuniform density is difficult to predict in advance. Therefore, this will be one of the main focuses in our future work.

In conclusion, MG/OPT can be applied to general configuration with a combination of OPT for the regions uncovered by equilateral triangles, and still can maintain a dramatically improved convergence speed. Therefore, MG/OPT has a practical utility in the real-life CVT applications.

<table>
<thead>
<tr>
<th></th>
<th>K</th>
<th>MG/OPT</th>
<th>OPT</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_1$</td>
<td>1081</td>
<td>0.0092</td>
<td>0.76</td>
</tr>
<tr>
<td>$A_2$</td>
<td>276</td>
<td>0.0100</td>
<td>0.45</td>
</tr>
<tr>
<td>$A_3$</td>
<td>91</td>
<td>0.0096</td>
<td>0.49</td>
</tr>
<tr>
<td>$A_4$</td>
<td>36</td>
<td>0.0060</td>
<td>0.28</td>
</tr>
<tr>
<td>$I$</td>
<td>26</td>
<td>–</td>
<td>0.23</td>
</tr>
<tr>
<td>$R_s$</td>
<td>9319</td>
<td>0.1369</td>
<td>0.85</td>
</tr>
</tbody>
</table>

Table 4.6: performance of a combination of MG/OPT and OPT v.s. OPT for constant density in a rectangular domain
Chapter 5: MG/OPT for Constrained Optimization Problems

5.1 Constrained version of MG/OPT

As we discussed before, MG/OPT has also been successfully applied to a class of PDE-constrained optimization problems that can be transformed to unconstrained problems. However, the MG/OPT framework is general, in the sense that it does not specify the type of underlying optimization algorithm, providing a great flexibility in how it is implemented. In particular, the MG/OPT constrained framework stated in [Nas10] shows a potential to be applied to a broad range of optimization models. In this section, we will direct our efforts to show specific implementations for certain types of PDE-constrained optimization problems by employing some underlying constrained optimization algorithms.

Suppose we need to solve the constrained optimization problems of the form:

\[
\min_{u_h} \quad f(u_h) \\
\text{s.t.} \quad a_h(u_h) \leq 0.
\]

The Lagrangian function is

\[
L_h(u_h, \lambda_h) = f_h(u_h) + a_h(u_h)^T \lambda_h
\]

where \(\lambda_h\) are Lagrange multipliers for the constraints.

Define OPT as a function of the form

\[
(u^+, \lambda^+) \leftarrow \text{OPT}(f, v, a, s_\alpha, \bar{u}, \bar{\lambda}, k)
\]
which applies $k$ iterations of a convergent optimization algorithm to a shifted or “surrogate” problem:

$$\min_u \quad f_s(u) \equiv f(u) - v^T u$$
$$\text{s.t.} \quad a_s(u) \equiv a(u) - s_a \leq 0$$

with initial guess $(\bar{u}, \bar{\lambda})$ to obtain $(u^+, \lambda^+)$ We also assume that the OPT is based on an augmented-Lagrangian merit function

$$M_h(x_u) \equiv f_h(u_h) + \alpha^+(u_h)^T \bar{\lambda}_h + \frac{\rho}{2} \alpha^+(u_h)^T \alpha^+(u_h)$$

where $\alpha^+(u_h) = \max(\alpha_h(u_h), 0)$ represents the violation of the constraints, and $\rho$ is a penalty parameter.

Here then is the algorithm: Given an initial estimate of the solution $(u^0_h, \lambda^0_h)$ on the fine level, set $v_h = 0$ and $s_{a,h} = 0$, provide available update and downdate operators: $I^H_h$ and $I^H_h$ for the variables $u_h$, and a downdate operator $J^H_h$ for the constraints $a_h$. Select non-negative integers $k_1$ and $k_2$ satisfying $k_1 + k_2 > 0$. Then for $j = 0, 1, \ldots$, set

$$(u^{j+1}_h, \lambda^{j+1}_h) \leftarrow \text{MG/OPT}(f_h, v_h, a_h, s_{a,h}, u^j_h, \lambda^j_h)$$

where the function MG/OPT is defined as follows:

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

  $$(u^{j+1}_h, \lambda^{j+1}_h) \leftarrow \text{OPT}(f_h, v_h, a_h, s_{a,h}, u^j_h, \lambda^j_h).$$

  Otherwise,

- **Pre-smoothing:**

  $$(\bar{u}_h, \bar{\lambda}_h) \leftarrow \text{OPT}(f_h, v_h, a_h, s_{a,h}, u^j_h, \lambda^j_h, k_1)$$
• **Coarse-grid correction:**

  – Compute

  \[
  \bar{u}_H = I_h^H \bar{u}_h \\
  \bar{\lambda}_H = J_h^H \bar{\lambda}_h \\
  \check{v}_H = I_h^H v_h + \nabla \mathcal{L}_H(\bar{u}_H, \bar{\lambda}_H) - I_h^H \nabla \mathcal{L}_h(\bar{u}_h, \bar{\lambda}_h) \\
  \check{s}_{a,H} = J_h^H s_{a,h} + a_H(\bar{u}_H) - J_h^H a_h(\bar{u}_h)
  \]

  – Apply MG/OPT recursively to the surrogate model:

  \[
  (u_H^+, \lambda_H^+) \leftarrow \text{MG/OPT}(f_H, \check{v}_H, a_H, \check{s}_{a,H}, \bar{u}_H, \bar{\lambda}_H)
  \]

  – Compute the search directions \( e_H = u_H^+ - \bar{u}_H \) and \( e_h = I_h^H e_H \).

  – Use a line search to determine \( u_h^+ = \bar{u}_h + \alpha e_h \) satisfying \( M_h(u_h^+) \leq M_h(\bar{u}_h) \).

  – Compute the new multipliers \( \lambda_h^+ \)

• **Post-smoothing:**

  \[
  (u_h^{j+1}, \lambda_h^{j+1}) \leftarrow \text{OPT}(f_h, v_h, a_h, s_{a,h}, u_h^+, \lambda_h^+, k_2)
  \]

### 5.2 1D Laplacian problem

Our first model problem is the 1D Laplacian problem with bound constraints:

\[
\begin{align*}
\min_u & \quad f(u) = \frac{1}{2} u^T A u - b^T u \\
\text{s.t.} & \quad l \leq u
\end{align*}
\] (5.1)
where $A$ is the negative matrix of 3 point finite difference Laplacian. If $h$ denotes the grid size, then

$$A = -\frac{1}{h^2} \begin{pmatrix} -2 & 1 & 0 & \ldots \\ 1 & -2 & 1 & \ldots \\ & \ddots & \ddots & \ddots \\ 0 & \ldots & 1 & -2 \end{pmatrix}.$$ 

The first attempt is to apply the same version of bound constraints to each level. By doing so, the search direction from recursion will have a chance to force the variables out of bound on the fine level, this observation leads to the fail of MG/OPT as shown in Figure 5.1 since MG/OPT performs similar as OPT. To be more specific, if we choose $I_h^H$ as the full weighting matrix and $I_h^h = 2I_h^{HT}$, and suppose the solutions $\bar{u}_{h,2i} = l$, $i = 1, \ldots, N_H$ are binding after pre-smoothing, and the other components satisfy $u_{h,j} > l$. Then

$$\bar{u}_{H,i} = \frac{1}{4} \bar{u}_{h,2i-1} + \frac{1}{2} \bar{u}_{h,2i} + \frac{1}{4} \bar{u}_{h,2i+1} > \bar{u}_{h,2i}.$$ 

Therefore, if the optimal solution from the recursion satisfies $u_{H,i} = l$, the updated variables on the fine level after the recursion will satisfy

$$u_{h,2i}^+ = \bar{u}_{h,2i} + u_{H,i}^+ - \bar{u}_{H,i} < \bar{u}_{h,2i} = l$$ 

which leads to the infeasibility of $u_h$. 
Thus, our goal is to formulate modified constraints on the coarser grid so that the updated variables after recursion will not go out of bound. Therefore, if we take $I_h^H$ and $J_h^H$ as full weighting matrix and $I_H^h = 2I_h^H$, we have

$$u^+_h = \bar{u}_h + I_H^h e_H.$$  \[
\text{(bound1)}
\]

Let’s now consider even and odd nodes separately:

For even nodes, we choose

$$l_{H,i} = l_{h,2i} + \frac{1}{4} \bar{u}_{h,2i-1} - \frac{1}{2} \bar{u}_{h,2i} + \frac{1}{4} \bar{u}_{h,2i+1}. \quad \text{(bound1)}$$
Then

\[ \bar{u}_{h,2i} + I_H^H(u_{H,i}^+ - \bar{u}_{H,i}) \]

\[ \geq \bar{u}_{h,2i} + I_H^H(l_{H,i} - \bar{u}_{H,i}) \]

\[ \geq \bar{u}_{h,2i} + I_H^H(l_{h,2i} + \frac{1}{4} \bar{u}_{h,2i-1} - \frac{1}{2} \bar{u}_{h,2i} + \frac{1}{4} \bar{u}_{h,2i+1} - \bar{u}_{H,i}) \]

\[ = \bar{u}_{h,2i} + l_{h,2i} + \frac{1}{4} \bar{u}_{h,2i-1} - \frac{1}{2} \bar{u}_{h,2i} + \frac{1}{4} \bar{u}_{h,2i+1} - (\frac{1}{4} \bar{u}_{h,2i-1} + \frac{1}{2} \bar{u}_{h,2i} + \frac{1}{4} \bar{u}_{h,2i+1}) \]

\[ = l_{h,2i} \]

For odd nodes, we choose

\[ l_{H,i-1} = l_{h,2i-1} - \bar{u}_{h,2i-1} + \bar{u}_{H,i-1} \quad \text{(bound2)} \]

and

\[ l_{H,i} = l_{h,2i-1} - \bar{u}_{h,2i-1} + \bar{u}_{H,i} \quad \text{(bound3)} \]

. Then

\[ \bar{u}_{h,2i-1} + \frac{1}{2}(u_{H,i-1}^+ - \bar{u}_{H,i-1}) + \frac{1}{2}(u_{H,i}^+ - \bar{u}_{H,i}) \]

\[ = \frac{1}{2}(\bar{u}_{h,2i-1} + u_{H,i-1}^+ - \bar{u}_{H,i-1}) + \frac{1}{2}(\bar{u}_{h,2i-1} + u_{H,i}^+ - \bar{u}_{H,i}) \]

\[ \geq \frac{1}{2}(l_{h,2i-1} + l_{h,2i-1}) \]

\[ = l_{h,2i-1} \]
Summarizing these three lower bounds (bound1), (bound2) and (bound3) to node $i$, then

\[
    l_{H,i} = \max \left( l_{h,2i} + \frac{1}{4}\bar{u}_{h,2i-1} - \frac{1}{2}\bar{u}_{h,2i} + \frac{1}{4}\bar{u}_{h,2i+1}, \\
    l_{h,2i+1} - \bar{u}_{h,2i+1} + \bar{u}_{H,i}, \\
    l_{h,2i-1} - \bar{u}_{h,2i-1} + \bar{u}_{H,i} \right)
\]

will guarantee the feasibility of variables at each level. For upper bounds and higher dimensional problems, similar calculations can be applied.

In our implementation, we choose OPT as TN with active set method embedded to handle bound constraints.

Figure 5.2: 1D Laplacian problem with lower bound
Figure 5.3: 1-D Laplacian problem with upper and lower bounds

Figure 5.4: Advection problem with upper bound

In Figure 5.2, we demonstrate the performance of MG/OPT compared to OPT for the
1D Laplacian problem with lower bounds. The grid size is 1151. With the same number of equivalent fine level gradient evaluations, MG/OPT shows a dramatic improvement on residual of objective function value than OPT. Similarly as shown in Figure 5.3, both upper and lower bounds are applied to the same problem, and the benefit of MG/OPT over OPT is clearly shown.

5.3 1D advection problem

Our next model problem is the initial value problem (IVP) for the linear advection equation stated at [LN00],

$$u_t + cu_x = 0 \tag{5.2}$$

$$u(x,0) = a(x)$$

The design variable is the initial condition $a(x)$. The objective is to minimize

$$F(a) = \frac{1}{2} \int_0^T \int \left[ \alpha(u(x,t) - \phi(x,t))^2 + \beta(u_x(x,t) - \phi_x(x,t))^2 \right] dx \, dt$$

where $\alpha, \beta$ are non-negative weights, and $\phi$ is a prescribed target. The numerical solution to (5.2) is stated in [LN00] as well as the choice for $\phi$. In the objective function, $\alpha = \beta = 1$.

Similar to problem (5.1), we take $I^H_h$ and $J^H_h$ as the full weighting matrix and $I^H_H = 2I^H_h^T$, apply the same strategy as well to handle the bound constraints on different levels, then the performance of MG/OPT compared to OPT is demonstrated in Figure 5.4. Clearly, MG/OPT outperforms OPT again. The reason MG/OPT does not reach the same accuracy as OPT is that the next function value reached by MG/OPT is smaller than the “exact” solution obtained by OPT.
5.4 2D linear control problem

Next, we will extend this framework to a 2D model problem: a Linear Control problem. The problem is stated as follows:

$$\min \|y - y\|^2 + \frac{\mu}{2} \|u\|^2 = f(u)$$

s.t. \(y = -A^{-1}(u + f)\)

(5.3)

where \(A\) is a five-point finite-difference Laplacian [A.23].

The first experiment we tried is to apply the same idea to handle the coarse grid bound constraints as in problem (5.1) so that no variables will go infeasible, but this attempt failed as shown in Figure 5.5, since

- First, on the coarse grid, the corresponding constraint will be too limited so that variables are barely moving, then the recursion is not able to give a descent direction at the finer level.

- Second, this strategy is not general and will cause a lot of complications in the higher-dimension cases.
First, let us take the standard choices for $I^H$ and $J^H$ as full weighting matrix and $I^h = 4(I^H)^T$, the search direction versus the error comparison is shown in Figure 5.6. Notice that the search direction is approximately 10 times tinier than the step to solution error, this result implies that the scaling of original problem is having an issue. Accordingly, it results in the failure of MG/OPT compared to OPT as shown in Figure 5.7, where the reduction of residual obtained by MG/OPT stagnates quickly before convergence. Motived by this fact, we rescale $I^H$ by $\frac{1}{2}$, accordingly, rescale $I^h$ by 2 based on the original setting. Therefore, $I^h = 16(I^H)^T$. With this change, we get good performance from MG/OPT as
shown in Figure 5.8.

Figure 5.6: MG/OPT search direction versus error for problem (5.3)

Figure 5.7: Equivalent gradient evaluations comparison between MG/OPT and OPT for problem (5.3)
Figure 5.8: Equivalent gradient evaluations comparison between MG/OPT and OPT for problem (5.3) after rescaling the transform operators

5.5 Summary of constrained version of MG/OPT

The extension to the constrained setting shows the considerable flexibility of MG/OPT. The computational experiments suggest that with proper adjustments of the fundamental components of MG/OPT, the overall convergence can be guaranteed and the coarse-level shifted problems can produce descent directions for the finer level models. Therefore, MG/OPT is capable of dramatically accelerating the performance of the underlying classical optimization algorithm.
Chapter 6: Conclusion and future work

In this dissertation, we achieved a deeper understanding of the nature of CVT. We looked at CVT via the energy function minimization and applied MG/OPT to this function. We also explored the theoretical properties of CVT by analyzing the 1D and 2D Hessian matrices.

We prove the Hessian matrices are positive definite for both dimensions with uniform density. In the same manner, we prove the convergence of the Lloyd method for the 2D case with the uniform density. Furthermore, for the 1D CVT with uniform density, we provide a rigorous analysis for the convergence speed obtained from the Lloyd method. With these results, more access is provided for us to analyze the convergence performance for certain methods to construct CVTs.

In the numerical experiments, we extend the multilevel optimization (MG/OPT) framework to construct CVTs. In the 1D case, we are able to successfully apply MG/OPT to general configurations with random initial guesses and nontrivial densities, by choosing the appropriate components such as transform operators so that MG/OPT beats other existing approaches. Besides, MG/OPT shows a significantly lower convergence factor which is independent of the problem size. This is exactly what we expected as typical multigrid efficiency. In the 2D case, MG/OPT shows its benefit for the equilateral triangle-based domain, in the sense that MG/OPT is also capable of achieving dramatically low and stable convergence factors with appropriate choices of algorithmic components. Although the multigrid efficiency is only exhibited for a triangular number of generators in this situation, we argue that our approach is still desirable in a general configuration. Therefore, the practical utility of MG/OPT is guaranteed to a great extent.

We also introduced a specific implementation of MG/OPT to solve certain types of PDE-constrained optimization problems in an explicit constrained setting. This outcome ensures
the flexibility of MG/OPT and opens a broader area for MG/OPT to show benefit in the manner of practical implementation. However, to achieve rapid convergence for MG/OPT comparable to traditional multigrid scheme, there is still a need for more sophisticated implementations which would provide an easier way to apply MG/OPT to specific problems, e.g., what is an appropriate underlying solver, how to wisely construct the coarse grid model and transfer operators. This gives us the motivation that the future work will also extend the potentiality of constrained version of MG/OPT to a broader class of constrained optimization problems such as nonlinear constraints.

Overall, the simplicity of the algorithmic design and the results of preliminary tests suggest that the method is generalizable to higher-dimension CVTs and other more complicated constrained optimization problems, 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 work. Future work also includes application of this technique to various scientific and engineering applications, including image analysis and grid generation.
Appendix A: Definitions and Background Information

A.1 Elliptic PDE

A second-order partial differential equation, i.e., one of the form

\[ Au_{xx} + 2Bu_{xy} + Cu_{yy} + Du_x + Eu_y + F = 0 \]

is called elliptic if the matrix

\[ Z \equiv \begin{bmatrix} A & B \\ B & C \end{bmatrix} \]

is positive definite.

A.2 Relaxation method

Classical iteration methods such as Gauss-Seidel-type and Jacobi-type iterations are often called relaxation methods (or smoothing methods or smoothers) if they are used for the purpose of error smoothing.
A.3Stencil notation

On the infinite grid $G_h$, let’s consider grid functions $w_h : G_h \to \mathbb{R}$. A general stencil $[s_{k_1k_2}]_h$

$$[s_{k_1k_2}]_h = \begin{bmatrix}
\vdots & \vdots & \vdots \\
\cdots & s_{-1,1} & s_{0,1} & s_{1,1} & \cdots \\
\cdots & s_{-1,0} & s_{0,0} & s_{1,0} & \cdots \\
\cdots & s_{-1,-1} & s_{0,-1} & s_{1,-1} & \cdots \\
\vdots & \vdots & \vdots & & \vdots
\end{bmatrix}$$

defines an operator on the set of grid functions by

$$[s_{k_1k_2}]_h w_h(x,y) = \sum_{(k_1,k_2)} s_{k_1k_2} w_h(x + k_1 h_x, y + k_2 h_y)$$

A.4Jacobi method

Given a square system of $n$ linear equations with unknown $x$:

$$Ax = b$$

(A.1)

where

$$A = \begin{bmatrix}
a_{11} & a_{12} & \cdots & a_{1n} \\
a_{21} & a_{22} & \cdots & a_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
a_{n1} & a_{n2} & \cdots & a_{nn}
\end{bmatrix},
\begin{bmatrix}
x_1 \\
x_2 \\
\vdots \\
x_n
\end{bmatrix},
\begin{bmatrix}
b_1 \\
b_2 \\
\vdots \\
b_n
\end{bmatrix}$$
A can be decomposed into a diagonal component $D$, and the remainder $R$:

\[
A = D + R \quad \text{where} \quad D = \begin{bmatrix}
a_{11} & 0 & \ldots & 0 \\
0 & a_{22} & \ldots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \ldots & a_{nn}
\end{bmatrix} \quad \text{and} \quad R = \begin{bmatrix}
0 & a_{12} & \ldots & a_{1n} \\
a_{21} & 0 & \ldots & a_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
a_{n1} & a_{n2} & \ldots & 0
\end{bmatrix}
\]

The solution is then obtained iteratively via

\[
x^{(k+1)} = D^{-1}(b - Rx^{(k)}). \]

A.5 Gauss-Seidel method

In numerical linear algebra, the Gauss-Seidel method, also known as the Liebmann method or the method of successive displacement, is an iterative method used to solve a linear system of equations. Consider the above problem (A.1), then $A$ can be decomposed into a lower triangular component $L$, and a strictly upper triangular component $U$:

\[
A = L + U, \quad \text{where} \quad L = \begin{bmatrix}
a_{11} & 0 & \ldots & 0 \\
a_{21} & a_{22} & \ldots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
a_{n1} & a_{n2} & \ldots & a_{nn}
\end{bmatrix} \quad \text{and} \quad U = \begin{bmatrix}
0 & a_{12} & \ldots & a_{1n} \\
0 & 0 & \ldots & a_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \ldots & 0
\end{bmatrix}
\]

The system of linear equations may be rewritten as:

\[
L x = b - U x
\]
The Gauss-Seidel method is an iterative technique that solves the left hand side of this expression for \( x \), using previous value for \( x \) on the right hand side. Analytically, this may be written as:

\[
x^{(k+1)} = L^{-1}(b - Ux^{(k)}).
\]

However, by taking advantage of the triangular form of \( L \), the elements of \( x^{(k+1)} \) can be computed sequentially using forward substitution:

\[
x^{(k+1)}_i = \frac{1}{a_{ii}} \left( b_i - \sum_{j<i} a_{ij}x^{(k+1)}_j - \sum_{j>i} a_{ij}x^{(k)}_j \right), \quad i = 1, 2, \ldots, n.
\]

### A.6 red-black Gauss-Seidel method

Imagine coloring the grid with alternate red and black grid points, starting at the lower right with red. We’ll assume here that \( n \) is even, or equivalently that the number of grid points on a side is odd. Denoting red by \( r \) and black by \( b \) a 5x5 grid would look like:

\[
\begin{array}{ccccc}
r & b & r & b & r \\
 b & r & b & r & b \\
r & b & r & b & r \\
b & r & b & r & b \\
r & b & r & b & r \\
\end{array}
\]

The red-black Gauss-Seidel (abbreviated RB G-S) is the order defined as follows:

First loop over the Red points, starting at the bottom left, going across the rows, then going up to the next row and so on. Then when ALL red points are finished, do the black ones, again starting at the bottom left. Boundary points are skipped as usual.

This particular ordering is very interesting, it is completely parallel WITHIN the red points. And within the black points.
A.7 Newton’s method

In numerical analysis, Newton’s method is a method for finding successively better approximations to the roots of a real-valued functions.

\[ x : f(x) = 0 \]

The Newton’s method in one variable is implemented as follows:

Given a function \( f \) defined over the reals \( x \), and its derivative \( f' \), we begin with a first guess \( x_0 \) for a root of the function \( f \), then repeat the iteration as

\[
x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}
\]

until a sufficiently accurate value is reached.

A.8 Linear Interpolation in stencil notation

\[
I^h_H \equiv \frac{1}{4} \begin{bmatrix}
1 \\
2 \\
1
\end{bmatrix}
\]
A.9  **Bilinear Interpolation**

The bilinear interpolation operator is a very frequently used interpolation method to map 2\(h\)-grid functions into \(h\)-grid functions, which is given by:

\[
I_{2h}^h v_{2h}(x, y) = \begin{cases} 
    v_{2h}(x, y) \\
    \frac{1}{2}(v_{2h}(x, y + h) + v_{2h}(x, y - h)) \\
    \frac{1}{2}(v_{2h}(x + h, y) + v_{2h}(x - h, y)) \\
    \frac{1}{4}(v_{2h}(x + h, y + h) + v_{2h}(x + h, y - h) + v_{2h}(x - h, y + h) + v_{2h}(x - h, y - h))
\end{cases}
\]

A.10  **Full-weighting Restriction in stencil notation**

\[
I_h^H \equiv \frac{1}{16} \begin{bmatrix}
    1 & 2 & 1 \\
    2 & 4 & 2 \\
    1 & 2 & 1
\end{bmatrix}
\]

A.11  **Nested Iteration**

A general idea of nested iteration is to provide an initial approximation on a fine grid by the computation and interpolation of approximations on coarser grid.

A.12  **Convergence factor**

Convergence factor of an iteration matrix \(T\) is defined as \(\max\{|\lambda|, \lambda \in \sigma(T), \lambda \neq 1\}\) where \(\sigma(T)\) is the spectrum of \(T\).

A.13  **Interior point Method**

The interior point methods are a certain class of algorithms to solve linear and nonlinear convex optimization problems. The method consists of a self-concordant barrier function
used to encode the convex set and reaches an optimal solution by traversing the interior of
the feasible region.

**A.14 Linear Conjugate gradient method**

The linear conjugate gradient (CG) method is an algorithm for the numerical solution of
particular systems of linear equations, namely those whose matrix is symmetric and positive-
definite. The conjugate gradient method is an iterative method, so it can be applied to
sparse systems that are too large to be handled by direct methods such as the Cholesky
decomposition. The simplest algorithm of CG for solving the system $Ax = b$ is as follows:

- Set $x_0 = 0, r_0 = b = b - Ax_0, p_{-1} = 0, \beta_0 = 0$, and specify the convergence tolerance
  $\epsilon$.
- For $i = 0, 1, \ldots$,
  
  1. If $\|r_i\| \leq \epsilon$, stop.
  2. If $i > 0$, set $\beta_i = r_i^T r_i / r_{i-1}^T r_{i-1}$
  3. Set $p_i = r_i + \beta_i p_{i-1}$
  4. Set $\alpha_i = r_i^T r_i / p_i^T A p_i$
  5. Set $x_{i+1} = x_i + \alpha_i p_i$
  6. Set $r_{i+1} = r_i - \alpha_i A p_i$

**A.15 Delaunay Neighboring**

In mathematics and computational geometry, a Delaunay triangulation for a set $P$ of points
in a plane is a triangulation $DT(P)$ such that no point in $P$ is inside the circumcircle of
any triangle in $DT(P)$. Accordingly, we call two points in $P$ are Delaunay neighboring if
they share a common edge in $DT(P)$.
A.16  **Truncated Newton method**

The truncated Newton methods are a compromise on Newton’s method which are used to solve the problem minimize \( f(x) \), and they compute a search direction by finding an approximate solution to the Newton equations \( \nabla^2 f(x_k)p \approx -\nabla f(x_k) \) using some iterative method, usually the conjugate gradient method.

A.17  **Maximal Independent set algorithm**

In graph theory, a maximal independent set is an independent set that is not a subset of any other independent set. That is, it is a set \( S \) such that every edge of the graph has at least one endpoint not in \( S \) and every vertex not in \( S \) has at least one neighbor in \( S \). The maximal Independent set algorithm is to find the maximal Independent set by using breadth first search.

A.18  **Barycentric Interpolation**

Barycentric interpolation is one kind of interpolation operator. Denote \( r_{i,j} \) as the entry to the interpolation operator \( I^h_H \) where \( i \) is the index for the fine nodes \( v_h \) and \( j \) is the index for the coarse nodes \( v_H \). The procedure to construct this \( I^h_H \) is to define the Delaunay triangulation based on coarse grid nodes. Then for each fine grid point \( v_{hi} \) identify the unique triangle it belongs to in this triangulation and set \( r_{i,j} \) as its barycentric coordinate with respect to each coarse node \( v_{Hj} \) which determines the corresponding triangulation.

A.19  **Schur complement method**

In numerical analysis, the Schur complement method, named after Issai Schur, is the basic and the earliest version of non-overlapping domain decomposition method. This method is proceeded as follows: a finite element problem is split into non-overlapping subdomains,
and the unknowns in the interiors of the subdomains are eliminated. The remaining Schur complement system on the unknowns associated with subdomain interfaces is solved by the conjugate gradient method.

**A.20 Richardson iteration**

To solve a system of linear equations \( Ax = b \), the Richardson iteration is \( x^{k+1} = x^k + \omega(b - Ax^k) \) where \( \omega \) is a scalar parameter that has to be chosen such that the sequence \( x^k \) converges.

**A.21 Karush-Kuhn-Tucker (KKT) conditions**

In mathematics, the Karush-Kuhn-Tucker (KKT) conditions (also known as the Karush-Kuhn-Tucker conditions) are first order necessary for a solution in nonlinear programming to be optimal, provided that some regularity conditions are satisfied. It is described as follows:

Let \( x^* \) be a clonal minimum point of \( f \) subject to the constraints \( g(x) \geq 0 \). Let the columns of \( Z(x^*) \) form a basis for the null space of the Jacobian of the active constraints at \( x^* \). If \( x^* \) is a regular point for the constraints, there exists a vector of Lagrange multipliers \( \lambda^* \) such that

- \( \nabla_x L(x^*, \lambda^*) = 0 \)
- \( \lambda^* \geq 0 \)
- \( \lambda^*_T g(x^*) = 0 \), and
- \( Z(x^*)^T \nabla_{xx}^2 L(x^*, \lambda^*) Z(x^*) \) is positive semidefinite
A.22 **Triangular number**

A triangular number or triangle number counts the objects that can form an equilateral triangle. The $n$th triangle number is the number of dots composing a triangle with $n$ dots on a side, and is equal to the sum of the $n$ natural numbers from 1 to $n$. The sequence of triangular numbers, starting at the 0th triangular number, is:

\[ 0, 1, 3, 6, 10, 15, 21, 28, 36, 45, 55, 66, 78, 91, 105, 120 \ldots \]

A.23 **Discrete Laplace operator: five point stencil**

In numerical analysis, given a square grid in one or two dimensions, the five-point stencil of a point in the grid is made up of the point itself together with its four "neighbors". It is used to write finite difference approximations to derivatives at grid points.
Bibliography
Bibliography


Curriculum Vitae

Zichao Di received her Bachelor of Sciences from Dalian University of Technology, Dalian, China in 2008. She was employed as graduate research assistant from spring 2010 to fall 2012, graduate teaching assistant at spring 2009 and spring 2013.