

DISSERTATION

RADIAL BASIS FUNCTIONS FOR COLOR CONVERSION

Submitted by

Yue Qiao

Department of Mathematics

In partial fulfillment of the requirements

for the degree of Doctor of Philosophy

Colorado State University

Fort Collins, Colorado

Spring 2008

UMI Number: 3400387

All rights reserved

INFORMATION TO ALL USERS

The quality of this reproduction is dependent upon the quality of the copy submitted.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if material had to be removed, a note will indicate the deletion.



UMI 3400387

Copyright 2010 by ProQuest LLC.

All rights reserved. This edition of the work is protected against unauthorized copying under Title 17, United States Code.



ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

COLORADO STATE UNIVERSITY

April 10, 2008

WE HEREBY RECOMMEND THAT THE DISSERTATION PREPARED UNDER OUR SUPERVISION BY YUE QIAO ENTITLED RADIAL BASIS FUNCTIONS FOR COLOR CONVERSION BE ACCEPTED AS FULFILLING IN PART REQUIREMENTS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY.

Committee on Graduate Work



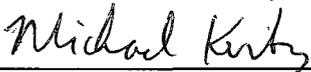
Dr. Charles Anderson



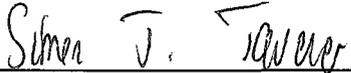
Dr. Gerhard Dangelmayr



Dr. Nenad Rijavec



Advisor



Department Head

ABSTRACT OF DISSERTATION

RADIAL BASIS FUNCTIONS FOR COLOR CONVERSION

The most difficult and challenging task in color printing is to reduce costs while maintaining superior quality. This dissertation proposes significant enhancements to printer color conversion techniques including accurate nonlinear models that incorporate perceptual color difference metrics, lossless gray component replacement (GCR) transformations, optimized toner saving algorithms and numerical/perceptual based gamut mapping methods.

Radial Basis Functions (RBFs) combined with the L_p norm approximation with emphasis on L_1 , L_2 , and L_∞ were developed for color conversion. The exchange algorithm was employed in the L_∞ and L_1 approximations with RBFs that satisfy the Haar condition. Both the Barrodale and Phillips (BP) algorithm for solving the dual problem and the Bartels and Conn's (BC) algorithm for solving the primal were extended to multidimensional color conversion.

A new approach for lossless GCR was achieved by finding one dimensional color manifolds in the CMYK color space using multidimensional optimization techniques. We proposed objective functions for toner savings, cost savings, etc., with no quality degradation.

The color conversion with the toner/ink limitation problem was solved via both L_1 and L_∞ approximation algorithms in the neutral and saturated color regions respectively. The L_1 algorithm was a modified Barrodale and Roberts (BR) primal algorithm with an added constraint, while the L_∞ algorithm was developed based on the BP dual algorithm which extended the three-stage algorithm to a four-stage algorithm.

A novel gamut mapping algorithm was developed based on the numerical model guided by a perceptual color difference model. The direction of the gamut mapping is not fixed as in other methods. The algorithm transformed a connected out-of-gamut colors to connected colors around the boundary of the device gamut. The out-of-gamut colors in a small neighborhood vary continuously and smoothly.

Our results indicated that the color conversion quality was significantly improved. The lossless GCR algorithm is accurate and efficient. Both the BP and BC algorithms for solving the toner/ink

limitation are able to convert colors from CIELab to CMY with any given toner/ink limitation. We foresee this research will have significant impact on the color reproduction industry.

Yue Qiao
Department of Mathematics
Colorado State University
Fort Collins, Colorado 80523
Spring 2008

ACKNOWLEDGEMENTS

I would like to thank my advisor, Dr. Michael Kirby, for his brilliant thesis topic suggestion and guidance. I wish to thank my committee members: Dr. Charles Anderson, Dr. Gerhard Dangelmayr and Dr. Nenad Rijavec for being interested in my thesis topic and for reviewing the thesis. Special thanks go to two of my professors, Dr. Kelly McArthur and Dr. Edwin Chong, who made their course work interesting and relevant, and always made time and effort to answer all my questions.

I would like to thank my company Infoprint Solutions, previously IBM Printing Systems Division, for the approval and support which enable my Ph.D study. Thanks also go to my colleges for there understanding and encouragement. Particularly, I would like to thank my mentors Joan Mitchell and Karen Fukuma, who always believe in me and never doubted that I would finish. In addition, they have provided role models and inspirations that I hope that someday I can pass on.

Words fail me to express my greatest gratitude to my family, my husband, Jeff, and my daughter, Poetica. I dedicate this work to them for their love, patience and support.

I would like to dedicate this thesis to my husband, Jeffrey, and my daughter, Poetica, for their love, support and patience.

TABLE OF CONTENTS

| | |
|--|-----------|
| 1 Problem Definition | 1 |
| 1.1 Introduction to Color Science | 2 |
| 1.1.1 Perceptual Color Spaces | 3 |
| 1.1.2 Perceptual Color Difference Tolerances | 5 |
| 1.2 Introduction to Color Management Systems | 7 |
| 1.2.1 What Does Color Management Do? | 7 |
| 1.2.2 Additive Color Systems and Subtractive Color Systems | 8 |
| 1.3 Printer Color Conversions | 9 |
| 1.3.1 The Nature of Printer Color Conversions | 10 |
| 1.3.2 Source of Errors | 11 |
| 1.3.3 Approximation Technique | 13 |
| 1.3.4 Gray Component Replacement (GCR) in Color Printing | 16 |
| 1.3.5 Toner Savings in Color Printing | 17 |
| 1.3.6 Gamut Mapping Algorithms | 17 |
| 1.3.7 Printer Color Conversion Quality Assessment | 21 |
| 1.4 Goals of the Research | 22 |
| 2 Fitting Nonlinear Scattered Data | 24 |
| 2.1 RBFs Overview | 25 |
| 2.2 Training Set Selection | 27 |
| 2.2.1 Ideal Data Description | 27 |
| 2.2.2 Data Used in This Thesis | 28 |
| 2.3 Cluster/Center Selection | 29 |
| 2.3.1 Clustering Algorithm | 31 |
| 2.3.2 Subset Selection Algorithms | 32 |
| 2.4 Function Optimization | 34 |
| 2.4.1 Parameter Optimization | 36 |

| | | |
|----------|---|-----------|
| 2.4.2 | Weight Determination vs. Norm Selection | 37 |
| 2.4.3 | Interior Points vs. Boundary Points | 38 |
| 2.4.4 | Cross Validation | 40 |
| 2.5 | Color Conversions Using the RBF Data Fitting Algorithm in the L_2 Norm | 42 |
| 2.6 | Summary | 44 |
| 3 | Overdetermined Linear Systems in the L_∞ Sense | 45 |
| 3.1 | Approximation Problem | 46 |
| 3.2 | Formulation of Linear Programming Problem | 47 |
| 3.2.1 | Primal Problem Formulation | 47 |
| 3.2.2 | Dual Problem Formulation | 48 |
| 3.3 | The Exchange Algorithm and Theorems | 48 |
| 3.4 | Barrodale and Phillips' Algorithm for Solving Dual Problem | 51 |
| 3.4.1 | The Condensed Tableau in the BP Dual Algorithm | 51 |
| 3.4.2 | The Three-Stage BP Dual Algorithm | 51 |
| 3.5 | Bartels and Conn's Algorithm for Solving Primal Problem | 53 |
| 3.5.1 | Conversion from a Constrained Problem to an Unconstrained Optimization Problem | 54 |
| 3.5.2 | Search Direction and Step Size | 55 |
| 3.6 | The BC Primal Algorithm in Color Conversion | 56 |
| 3.7 | The BP Dual Algorithm in Multidimensional Color Conversion | 58 |
| 3.8 | Experiments and Results | 61 |
| 3.9 | Summary | 62 |
| 4 | Color Conversion Using the L_1 Norm Approximation | 64 |
| 4.1 | Neutral Color Conversion Problem | 64 |
| 4.2 | Overdetermined Linear Systems in the L_1 Sense | 65 |
| 4.3 | The Theory of the Best L_1 Approximation | 66 |
| 4.4 | Barrodale and Roberts' Algorithm (BR algorithm) for Discrete L_1 Linear Approximation | 68 |
| 4.5 | BR Algorithm for Solving the Multidimensional Neutral Color Conversion Problem . . . | 69 |
| 4.6 | Experiments and Results | 72 |
| 4.7 | Summary | 73 |

| | | |
|----------|--|------------|
| 5 | One Dimensional Color Manifolds in the CMYK Color Space | 74 |
| 5.1 | Computing an Optimal CMYK Color | 75 |
| 5.2 | Initializing the Algorithm | 76 |
| 5.3 | Optimized Gray Component Replacement Algorithm | 77 |
| 5.4 | Finding a Level Set for a Given Color | 79 |
| 5.5 | The Optimal Toner/ink Saving Solution | 80 |
| 5.6 | Experiments and Results | 80 |
| 5.7 | Summary | 82 |
| 6 | Color Conversion with Ink Limitations | 83 |
| 6.1 | The Ink Limitation Problem Using the L_1 Approximation | 83 |
| 6.1.1 | The Primal Problem Formulation in the L_1 Sense | 84 |
| 6.1.2 | The Modified BR L_1 Approximation with the Toner/ink Limitation | 85 |
| 6.2 | Solving the Ink Limitation Problem Using the L_∞ Approximation | 87 |
| 6.2.1 | The Problem Formulation in the L_∞ Sense | 87 |
| 6.2.2 | The Modified BP L_∞ Approximation Algorithm for the Toner/ink Limitation Problem | 89 |
| 6.3 | Experiment and Results | 92 |
| 6.4 | Summary | 92 |
| 7 | Gamut Mapping | 93 |
| 7.1 | Gamut Mapping Algorithm Based on a Numerical and Perceptual Model | 93 |
| 7.2 | Experiments and Results of Gamut Mapping Algorithm Comparison | 95 |
| 7.3 | Summary | 97 |
| | Appendix I: Color Science Terminology | 97 |
| | Appendix II: Matlab Code | 99 |
| | References | 117 |

LIST OF FIGURES

| | | |
|------|--|----|
| 1.1 | Color reproduction problems. | 2 |
| 1.2 | What color science studies. | 3 |
| 1.3 | Illustration of CIELab color space. | 4 |
| 1.4 | Illustration of CIELCh color space. | 4 |
| 1.5 | The non-uniformity of CIELab chroma [5]. | 6 |
| 1.6 | Illustration of non-uniformity of CIELab in hue angle [5]. | 6 |
| 1.7 | Illustration of a color management system. | 8 |
| 1.8 | Illustration of the five tetrahedra in a cube [31]. | 14 |
| 1.9 | Gamut comparison of a monitor (represented by the wire frame) and a printer (represented by the solid shape). | 18 |
| 1.10 | Colors of Google image outside a printer device gamut. | 19 |
| 1.11 | Gamut mapping directions [35]. | 20 |
| 2.1 | A hue angle histogram for a $9 \times 9 \times 9 \times 9$ regular-spaced CMYK data. | 28 |
| 2.2 | Number of cluster centers vs. maximum errors. | 30 |
| 2.3 | Number of cluster centers vs. mean errors. | 30 |
| 2.4 | The partition of the CIELab color space in the ab^* plane for different norm approximations. | 39 |
| 2.5 | K values in the K-fold cross validation versus mean errors. | 41 |
| 2.6 | K values in the K-fold cross validation versus variance of mean error. | 41 |
| 2.7 | K values in the K-fold cross validation versus maximum errors. | 42 |
| 3.1 | Illustration of the main idea of the Exchange Algorithm. | 50 |
| 3.2 | The maximum error and its variance as function of cross validation K value. | 62 |
| 4.1 | The mean error and its variance as a function of cross validation K value. | 72 |
| 5.1 | The level set of CMYK values for $CMY = \{0.55, 0.4, 0.55\}$ | 81 |
| 5.2 | The level set of CMYK values for $CMY = \{0.7, 1.0, 1.0\}$ | 81 |

| | | |
|-----|---|----|
| 7.1 | Illustration of the current gamut mapping algorithm for the color data at hue angle 300° . . . | 96 |
| 7.2 | Gamut mapping comparison for blue-purple colors. | 96 |

LIST OF TABLES

| | | |
|-----|--|----|
| 3.1 | Condensed initial simplex tableau. | 52 |
| 3.2 | Condensed initial simplex tableau for the CIELab to CMY conversion. | 60 |
| 4.1 | Condensed initial simplex tableau for the CIELab to CMY conversion in the L_1 sense. | 71 |
| 6.1 | Condensed initial simplex tableau for the CIELab to CMY conversion with toner/ink limitation in the L_1 sense. | 86 |
| 6.2 | Condensed initial simplex tableau for the CIELab to CMY conversion with the Toner/ink Limitation. | 91 |

Chapter 1

PROBLEM DEFINITION

In recent years, the technology of color reproduction methods has undergone a radical transformation from an analog to a digital base. There has also been a dramatic change in the consumer market, where digital devices including digital cameras, color scanners, color monitors, color printers, etc have become significantly less expensive and widely available in homes and offices. While people are excited about the opportunity of reproducing color photographs via digital editing in their own homes, they also have started to realize that obtaining accurate, consistent and predictable color reproduction with different color devices and software is very challenging. For example, when an image is captured by a digital camera, displayed on a monitor and printed on a color printer, “what you see is what you get” usually does not hold true. This is shown in Figure 1. The image displayed on the monitor is often darker and less detailed than the original scene. In addition to the quality degradation in the monitor image, colors in the printed image are less accurate and less vivid.

Similar to biological species in nature, digital devices, operating systems, applications and device drivers all interpret and produce color differently. However, the human visual system (HVS) is highly discriminating when discerning quality in color reproductions. Therefore, there is a strong demand for the technology to ensure consistent and predictable color across peripheral devices and across operating-system platforms. This technology is called the Color Management System (CMS)

The most challenging task in the CMS process is to characterize each color device in the color reproduction system. There are many issues involved in building the characterization model, such as the color space, color data approximation, gamut mapping, color difference tolerance, etc.[1]. This dissertation proposes significant enhancements to color conversion techniques including accurate nonlinear models that incorporate perceptual color difference metrics, lossless gray component replacement (GCR) transformations, optimized toner saving algorithms and numerical/perceptual based gamut mapping methods. We foresee this research will have significant impact on the color reproduction industry.

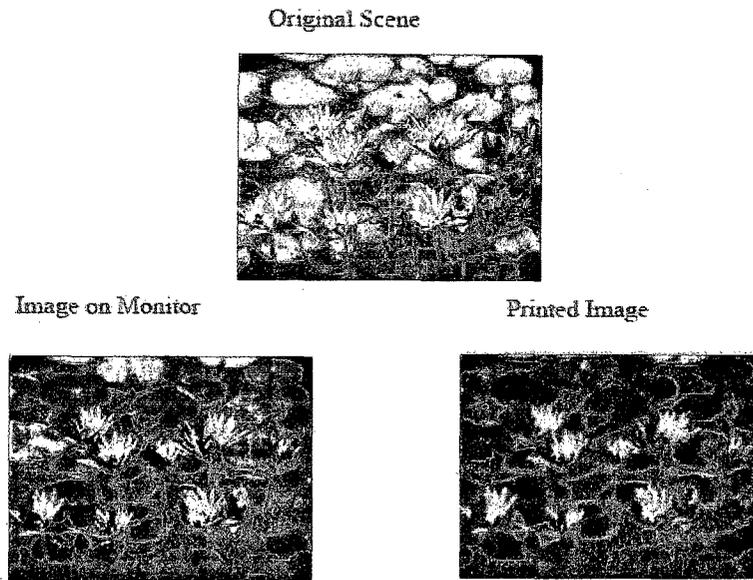


Figure 1.1: Color reproduction problems.

1.1 Introduction to Color Science

Color science studies the interactions of light with an object, including its background and surround, and the observer. This is illustrated in Figure 1.2. The light source is characterized by its relative spectral power distribution; the object is measured by spectral reflectance; the HVS has three receptors characterized by three spectral sensitivity functions. The Commission International De l'Eclairage (CIE) has standardized these functions to represent the standard observers, for example the CIE 1931 2 degree standard observer. The output of the product of light source, object, and CIE observer thus has three values (X, Y, Z) , which are called tristimulus values. When the tristimulus values of two colors are equal, i.e., $(X_1, Y_1, Z_1) = (X_2, Y_2, Z_2)$, these colors are said to match. Based on tristimulus values, CIE developed perceptual color spaces such as CIE XYZ and CIE Lab. The glossary of color science terminology is found in Appendix: "Color Science Terminology".

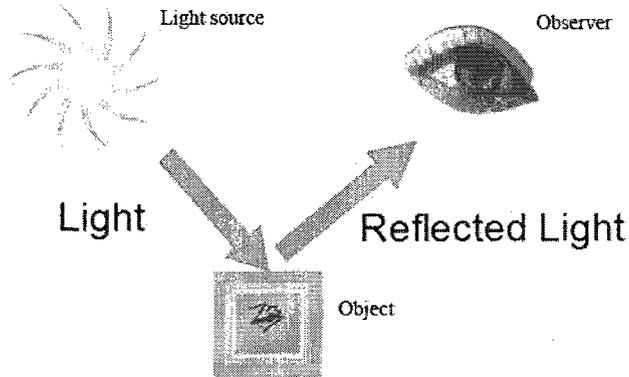


Figure 1.2: What color science studies.

1.1.1 Perceptual Color Spaces

The perceptual color spaces, both CIEXYZ and CIELab, are visualized as three dimensional color spaces, where every color that human can see is uniquely located. The location of any color in the space is determined by its color coordinates. In the CIEXYZ color space, the components are X, Y, and Z; Y is luminance, X and Z do not correlate to color appearances. In the CIELab color space: the components are L^* , a^* , and b^* ; L^* is lightness of the color ($L^* = 0$ yields black and $L^* = 100$ indicates white), a^* is the value between red and green (negative values indicate green while positive values indicate red) and b^* is the value between yellow and blue (negative values indicate blue and positive values indicate yellow). The CIELab color space can also be described using cylindrical coordinates: the L^*C^*h color space where C_{ab}^* is the chroma coordinate, and h is the hue coordinate. Below are the mathematical expressions to convert a^* and b^* into C_{ab}^* and h:

$$C_{ab}^* = \sqrt{a^{*2} + b^{*2}}$$

$$h = \tan\left(\frac{b^*}{a^*}\right)$$

Figure 1.3 shows the CIELab color space, while Figure 1.4 shows the CIELCh color space. Note that the asterisk (*) are normally left out of the color space name.

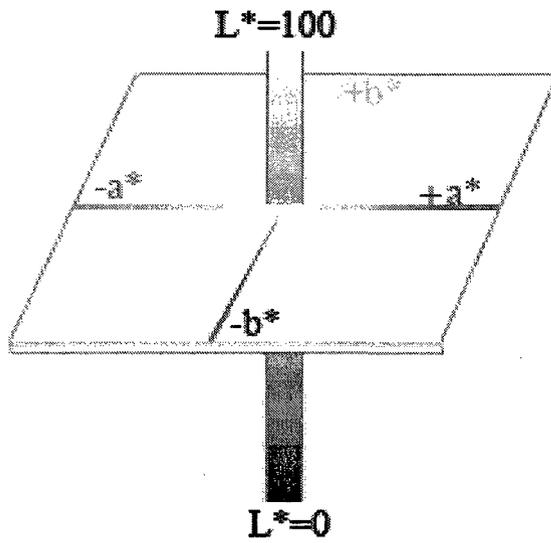


Figure 1.3: Illustration of CIELab color space.

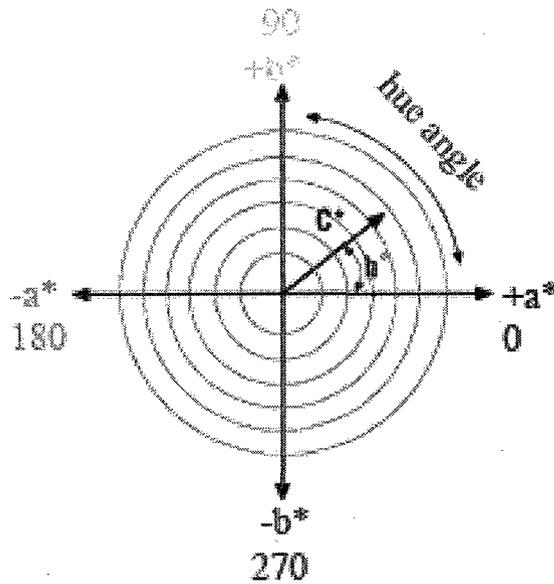


Figure 1.4: Illustration of CIELCh color space.

The relationship between CIE XYZ and CIE Lab is described using the following equations:

$$\begin{aligned} L^* &= 116f\left(\frac{Y}{Y_n}\right) - 16, \\ a^* &= 500\left(f\left(\frac{X}{X_n}\right) - f\left(\frac{Y}{Y_n}\right)\right), \\ b^* &= 200\left(f\left(\frac{Y}{Y_n}\right) - f\left(\frac{Z}{Z_n}\right)\right), \end{aligned}$$

where X_n, Y_n, Z_n are the tristimulus values of a reference white. It is also referred as the white point.

$$f(x) = \begin{cases} x^{1/3}, & x > 0.008856 \\ 7.787x + \frac{16}{116}, & x \leq 0.008856 \end{cases}$$

1.1.2 Perceptual Color Difference Tolerances

The color difference in either CIE XYZ or CIE Lab is calculated using the Euclidean distance. Though the CIE XYZ and CIE Lab color space are perceptual color spaces, they are not perceptually *uniform* color spaces as the Euclidean distance in the space does not correspond to perceptual distance. For example, all the colors that have the same Euclidean distance with respect to an anchor color are on a circle. However, these circles are transformed to ellipsoids when the human perceptual distances are measured. The major axis of the ellipsoid represents the higher tolerance of the color difference, and the minor axis of the ellipsoid represents the lower tolerance for the color difference. The CIE Lab color space is a perceptually more uniform color space than the CIE XYZ color space. Thus, the CIE Lab color space is the color space used most often for color-difference evaluations.

Studies have shown that in the CIE Lab color space, the magnitude of the perceptual color difference depends upon the color location and the changing direction in chroma and hue [4]. CIE Lab is believed to increasingly overstate the magnitudes of perceived chroma differences. For example, the human visual system (HVS) is very sensitive to the change of chroma in the neutral color area, and is insensitive to the change of the chroma in a highly saturated color area. Figure 1.5¹ shows the size of the ellipsoid increases with the increase of the chroma value [5]. A few experimental hue supra-threshold data sets were generated independently by different research institutes. They all indicated that the CIE Lab color space is also non-uniform regarding hue angle, i.e., the thresholds of visual tolerances are a function of hue angle. These data sets were plotted in Figure 1.6². It shows that human perception is very sensitive to the hue change around hue angle 60 (Orangeish red color), and 300 (blueish purple color) [5].

¹Reprinted with permission of John Wiley & sons, inc.

²Reprinted with permission of John Wiley & sons, inc.

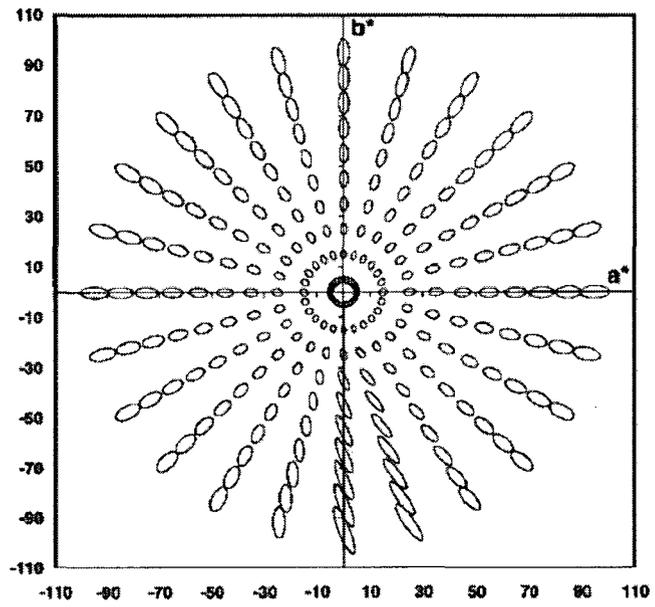


Figure 1.5: The non-uniformity of CIE Lab chroma [5].

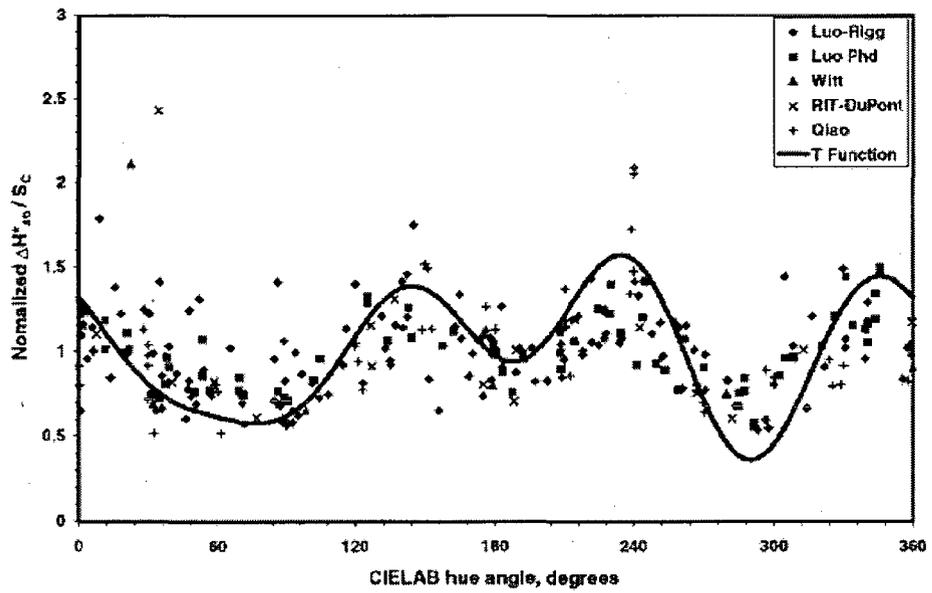


Figure 1.6: Illustration of non-uniformity of CIE Lab in hue angle [5].

If we examine the non-uniformity of the CIELab color space in chroma and hue from another perspective, it means that CIELab colors have different characteristics at different locations. We believe the change of these characteristics is continuous. Understanding the local and global characteristics of colors is very important in color space modeling, and will be discussed further in later chapters.

Color difference models are developed to correlate the perceptual distance with the Euclidean distance in the CIELab color space based on vast amounts of color difference experimental data [4, 6, 7]. The experiments provided the visual tolerance thresholds which define the visually approved matches based on the tolerance of observer's acceptability for pairs of colors. There are several CIE perceptual-based color difference models [8, 9, 10, 11]. The most recent one is CIEDE2000, which calculates the perceptual difference by weighted lightness difference, chroma difference, hue difference, and the interaction between the chroma and the hue difference. However, the CIEDE2000 was developed for small color differences (industrial size), and it does not work well for large color differences [5]. The empirical weighted color differences of lightness, chroma, and hue are often used for this case, with higher weighting factors for hue and lightness differences, and a lower weighting factor for chroma differences. The color difference calculated using the Euclidean distance in the CIELab is called ΔE , also referred as ΔE , or dE_{ab} . dE_{00} represents the color difference calculated using the CIEDE2000.

1.2 Introduction to Color Management Systems

A typical imaging system consists of a source device (e.g., scanner, digital camera, etc), a display device (e.g., monitor) and an destination device (e.g., printer). Each of these devices produces and interprets colors differently. This creates a significant problem in terms of color accuracy and consistency in color reproduction. For example, a customer's banking statement contains several object contents that originated from different devices and software: a personal photo taken by a digital camera, a car advertisement scanned from a picture, the background graphics created by a graphic package, etc. It is very hard to predict what these colors look like when it is displayed on different monitor or printed from different printers. It's not hard to believe that colors will be inconsistent and inaccurate.

1.2.1 What Does Color Management Do?

The goals of all color management systems are to:

- Establish consistent and predictable colors throughout all parts of the color reproduction chain.
- Establish standard operating procedures for color reproduction to reduce color deviation caused by variation in techniques.

A color management system comprises three major components:

- A device-independent color space either CIEXYZ or CIELab is used as the reference space communicating color information across different devices and systems [2]. The CIELab color space is the reference color space most often used for printers.
- Device profiles which provide color conversions between the device-dependent color space and the reference color space. The reference color space is also called the profile connection space (PCS).
- A Color Matching Module (CMM) that interprets colors using the color conversion model contained in a profile.

Figure 1.7 illustrates a color reproduction system: a scanner, a display, and a printer. All have their own profiles converting from/to a PCS.

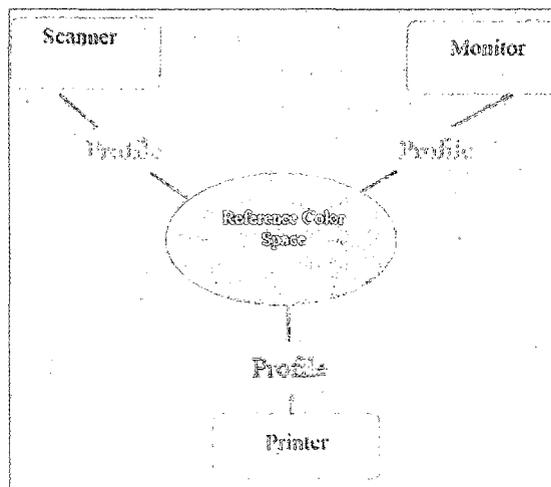


Figure 1.7: Illustration of a color management system.

1.2.2 Additive Color Systems and Subtractive Color Systems

In general, there are two types of systems among color devices: additive systems and subtractive systems. In additive systems, such as video monitors and televisions, colors are transmitted lights. Red, green and blue lights are referred as the additive primary colors. When used in various degrees of intensity and variation, they create all other colors of light; when superimposed equally, they create gray. Color printers are subtractive systems. The primary colors are cyan, magenta, yellow, and black. They are used together to effectively create a multitude of other colors based on the subtractive color theory.

The color conversions for an additive system are normally between the RGB color space and CIEXYZ color space. The conversions are relatively easy and straightforward since for most devices, the gamma corrected RGB primaries are linearly related to XYZ values in the CIEXYZ space. An example of such conversion is the conversion for the sRGB colors described as follows:

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} 0.4124 & 0.3576 & 0.1805 \\ 0.2126 & 0.7152 & 0.0722 \\ 0.0193 & 0.1192 & 0.9505 \end{pmatrix} \begin{pmatrix} g(R_{srgb}) \\ g(G_{srgb}) \\ g(B_{srgb}) \end{pmatrix}$$

where

$$g(K) = \begin{cases} \left(\frac{K+\alpha}{1+\alpha}\right)^\gamma, & K > 0.4045 \\ \frac{K}{12.92}, & \text{otherwise} \end{cases}$$

and

$$\gamma = 2.2$$

$$\alpha = 0.055$$

Among all the device color conversions, the printer color conversion is most challenging and complicated. Issues regarding building printer color conversion profiles are described in detail in the next section.

1.3 Printer Color Conversions

Color printers can be toner-based or ink-based. Most of the printers are only able to produce a limited numbers of gray levels, e.g., a binary printer can only produce two levels, i.e., 0 and 1; 2-bit printers produce level of 0, 1, 2, and 3. In order to produce continuous tone imagery that contains an infinite range of colors or greys, a reprographic technique called halftoning needs to be applied to create the illusion of continuous tone images through the use of dot arrangements and dots of varying size [12]. The combination of the printer halftone design and the specific toner/ink selection determine the number of colors that a printer is physically able to produce which is called its gamut. The gamut is measured in the CIE Lab color space.

It's easy to see that the printer color conversions depend on the printer technology, the toner/ink selections, and the halftone design. It is performed between a device-dependent color space (CMYK) and a device-independent color space (CIE Lab). Not required but most common, the format of the color conversion is an equally spaced lookup table (LUT). The values in the LUT are either 1-byte or 2-byte integers.

The first step in generating a printer color conversion is to print many CMYK color patches whose values span the CMYK color space. Each component value of CMYK is normally 1-byte integer. The

printed patches are measured using a spectral photometer which measures the spectral reflectance of each patch under a standard illuminant. The software in the spectral photometer calculates the tristimulus values, and then converts these values to the CIELab values. The CIELab values are floating point numbers. Followed by the implementation of a data modeling algorithm and a gamut mapping algorithm, the color conversions LUTs are generated using this set of data. All the CMYK and the CIELab values are normalized to the range of 0 – 1 for these calculations.

Both CMYK to CIELab and CIELab to CMYK conversions need to be generated for a color printer. The CMYK to CIELab color conversion is used when an image generated in this CMYK color space is to display on a monitor or to print on a different color printer. In other words, this conversion is used when the printer CMYK color space is used as a source color space. The CIELab to CMYK color conversion is used when the CIELab color space is the source color space or this CMYK color space is used as a destination color space.

In this section, we discuss the facts, issues, and current methodologies in the printer color conversions including:

- The characteristics of printer color conversions.
- The analysis of color conversion source errors.
- Interpolation algorithms.
- The Gray Component Replacement algorithm.
- The toner saver algorithm.
- The gamut mapping algorithm.

1.3.1 The Nature of Printer Color Conversions

For most color printers, the relationship between CMYK color space and the CIELab color space is highly nonlinear due to the interactions of cyan, magenta, yellow, and black planes in the printer subtractive system. No trivial functions can describe the behavior. Another complication to the printer color conversion is due to the halftone design. Most of halftone techniques are capable of creating color gray levels continuously. However, an abrupt change of two adjacent levels is a weakness in many halftone designs. Therefore, the color conversion between CMYK and CIELab is continuous but may not be differentiable.

The CMYK to CIELab color conversion is neither injective nor surjective:

- Every color in the CMYK color space can be mapped to a color in the CIELab color space. However, different CMYK values can be mapped to the same CIELab value. So the CMYK to CIELab color conversion is not injective.
- The range of CIELab color space is much bigger than the range of CMYK color space, i.e., for many CIELab values, there do not exist CMYK values such that $f(C, M, Y, K) \rightarrow (L, a, b)$. Thus the CMYK to CIELab color conversion is not surjective.

The CIELab to CMYK conversion is from \mathbb{R}^3 to \mathbb{R}^4 . There exists one degree of freedom in this conversion. One CIELab value could be mapped to several CMYK values. On the other hand, because of the smaller gamut of the color printer, not every CIELab value can be mapped to a CMYK value. Thus, there does not exist a function for the CIELab to CMYK color conversion. The CIELab to CMYK conversion is normally achieved by converting the CIELab values to CMY values first. Then a special rule is created for adding the black toner to each CMY value.

- The CIELab to CMY color conversion within a printer gamut is both injective and surjective.
 - Every CIELab value inside the printer color gamut can be mapped to a unique color in the CMY color space. So the CIELab to CMY color conversion is injective within color gamut.
 - For every CMY value, there exists a CIELab value such that $g(L, a, b) \rightarrow (C, M, Y)$. Thus the CIELab to CMY color conversion is surjective.
- The out-of gamut CIELab values can not be mapped to a CMY value. Therefore, there does not exist a function for the CIELab to CMY conversion outside the printer gamut.

Thus the color conversion between CMYK color space and CIELab color space can be only achieved by approximation.

1.3.2 Source of Errors

Understanding the sources of error is critical for the design of an approximation model. In addition, it's important to understand how these errors interfere with human perceptions for the design of the printer color conversion model.

There are four significant sources of error in the color conversion process. These errors are normally measured with the Euclidean distance in the CIELab color space, ΔE :

- Measurement errors from the instrument. These errors are normally smaller than $1 \Delta E$.

- Errors due to the machine reliability and repeatability. For a well-calibrated printer, these errors could be smaller than 2 units of ΔE .
- Errors due to paper roughness and non-uniform fibers and fillers formation. These errors are typically smaller than 1 unit of ΔE .
- Errors from the model of the color conversion. For inside gamut color conversion, the average interpolation error is around 3-4 units of ΔE , and the maximum error can be anywhere from 10 to 20 units of ΔE .

Because human perceptual tolerance varies from location to location in the CIELab color space, care must be taken in the approximation algorithms in different color areas.

- For colors in the neutral areas, the visual color difference tolerance is low (around 1 ΔE). The interpolation errors in these regions thus need to be small. If the accumulated errors from the first three categories are added to the modeling data, the effect of the outliers in the modeling set has to be minimized.
- For the medium colors, with the increase of the chroma values in these areas, the perceptual tolerance increases. Though the outliers still exist, they play a much less important role in the color conversion. It's appropriate to design an approximation model to minimize the least squares error.
- For colors in the highly saturated color regions, although the visual color difference tolerance is much higher, the interpolation errors are also high. The effect of errors from the first three sources do not contribute to much of output color differences. The goal of the approximation algorithm thus is to minimize the maximum error in the interpolation.

In general, color conversion approximation algorithms aim at minimizing least square errors [16, 22, 23, 24]. Due to the sources of error described above, we are motivated to consider the L_p error criteria, in particular the L_1 and L_∞ error for a robust neutral area color conversion and the reduction of maximum error in the saturated color regions. The color conversion techniques we developed with the considerations of printer sources of error and human perceptual tolerance are described in Chapters 2-4.

1.3.3 Approximation Technique

The core component in the printer color conversion is the design of the approximation algorithm. Over the last twenty years, there has been much research in this area. Today it is still an active area of research.

The techniques are divided into two categories: data interpolation and data fitting [24]. Most of the research in color conversion has been concentrated on the interpolation algorithms, while very little research has been done on the data fitting techniques. The major difference of these two approaches is if the approximation function passes through or passes by all the data points. The common belief is that, although the color data could have a little noise from the printer and the measurements, the data is considered to be accurate enough. Therefore, the interpolation techniques are desirable.

In this subsection, we first discuss the popular interpolation techniques used in the printer color conversion, and the pros and cons of each algorithm. We then briefly discuss the advantages of the data fitting algorithm for the color conversion problem.

The interpolation algorithms may be classified as local and global approaches. Because of the difficulties in finding the nonlinear model of the printer color conversion, the most popular interpolation techniques are local interpolation. There are two types of local interpolation methods based on the geometry structure: triangulation and Voronoi tessellation [24]. The local interpolation techniques include linear interpolation, quadratic spline, cubic spline, polynomials, etc. The following is the example of triangular linear interpolation, also called tetrahedral linear interpolation.

Triangulation based interpolation divides the CMY color space evenly into fine regions, and further triangulates these regions. Each of the color components cyan, magenta, yellow in the CMY color space is divided into N segments. The color combination of the i th node point of cyan, magenta and yellow $P_i(c, m, y)$ is printed and measured with the CIELab value $V_i(L, a, b)$, where $0 \leq i < N^3$; and V_i is measured CIELab value of P_i . Tetrahedral linear interpolation is then applied to a cube which is divided into 5 or 6 tetrahedra [31]. Figure 1.8³ illustrates the division of a cube into five tetrahedra.

We have the following equations:

$$P = P_0 + \sum_{j=1}^3 u_j (P_j - P_0)$$
$$V = V_0 + \sum_{j=1}^3 u_j (V_j - V_0)$$

³Reprinted with permission of SPIE and author Dr. Henry Kang.

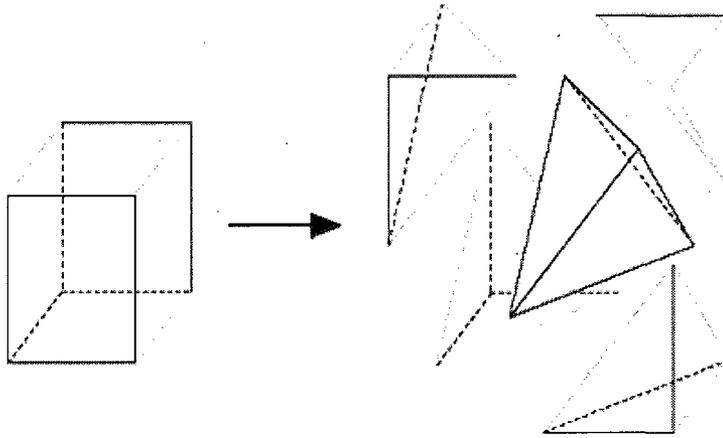


Figure 1.8: Illustration of the five tetrahedra in a cube [31].

where $u_j = V_j/V$ and V is the volume of the tetrahedron defined by the four points at which the function value is known, and V_j is the volume of the tetrahedron defined by the points if P is substituted for the point P_i . To solve for P for a given V , one solves the second equation for the u_j and substitutes the values into the first equation [32].

The converted CMY values are transformed to CMYK values using gray component replacement (GCR), which is discussed in detail in the following subsection. The drawbacks of this approach are

- One has to generate a large data set that evenly spans the gamut of the device.
- Data which are not on the grid of evenly spaced data are wasted.
- The local linear assumption is only an approximate model of the data. It is not accurate in the regions which are highly nonlinear.
- The linear approximation model is C^0 continuous, and is not C^1 continuous. It may create a non-smooth change at the boundary between adjacent tetrahedra.

The quadratic spline, cubic spline, and polynomials are the improvements over the linear interpolation method [16, 25, 26]. These approximation functions are not only C^0 continuous but also C^1 continuous. Even though all three methods give much smoother results, they may not give accurate results for some colors due to the the quality of the underlying triangulation [24]. The predetermined triangulations may not suit the best of the underlying nonlinear phenomena of the printer subtractive

system. In addition, the color gamut of some color printers is not convex. Some tetrahedra may be partially outside the printer gamut. Thus the interpolation within these regions gives meaningless results [24].

Another local interpolation algorithm is the natural neighbor interpolation method, which is based on the Voronoi region [29].

Definition 1.3.1 [46] *A Voronoi region V_i associated with the center c_i is the set of points for which c_i is the nearest center vector, i.e.,*

$$V_i = \{x \in \mathbb{R}^n : i = \arg \min_{j \in I} \|x - c_j\|\},$$

where I is the set of center indices.⁴

This interpolation method is a local region-based weighting method. It is believed that the Voronoi region fits the underlying printer characteristics better than the triangulation.

The examples of global interpolation methods include the inverse distance weighting method and the RBF method [16, 17, 18, 19, 24, 27, 28]. For the inverse distance weighting method, all the data points are used to interpolate a value. It assumes that the interpolated values are affected more by nearby points and less by the more distant points [24]. The assumption is too simple and not accurate for colors in many regions. Thus, this method in general gives poor color conversion results.

There have been relatively few attempts to apply the RBF method to the conversion problem. Artusi and Wilkie were the first to implement Radial Basis Function Networks (RBFNs) for the printer color conversion problem [16]. They attempted to demonstrate that the RBFNs were superior to polynomial and triangular-based interpolations. They used a small data set and a very basic RBFN algorithm to solve the color conversion problems. Their experimental results indicated that the RBFN method performed equally well or better than the polynomial based network method and the tetrahedral interpolation method for all their testing data sets. Isaac Amidror conducted a survey on commonly used scattered data interpolation algorithms for color conversion [24]. He reached a similar conclusion as Artusi and Wilkie, that the RBF based interpolation performed better than all algorithms that we reviewed in this subsection. We remark that all previous applications of RBFs to the color conversion problem employ the standard least square error criterion and do not employ perceptual color models. A detailed discussion of the RBFs algorithms is given in Chapters 2-4.

⁴Sometimes this is referred to as a first order Voronoi region since only one center is used in its definition.

We propose to establish that the RBF data fitting algorithm equipped with an L_p error-criterion works very well for the color conversion problem. Even though color data are considered to be accurate enough for building the color conversion model, we provide evidence that the perceptual color difference tolerance should be taken into account. As discussed in the previous subsection, the perceptual color difference tolerances vary over different CIELab color regions. Taking this into account, the data fitting technique has the advantage of minimizing color differences in different areas with respect to human perceptions and give much smoother results. We propose to use a RBFs data fitting technique that incorporates the perceptual color difference tolerances to solve the color conversion problem.

1.3.4 Gray Component Replacement (GCR) in Color Printing

Although black toner/ink is one of the primaries in a color printer, the black toner/ink actually serves as a “secondary” toner in color printing. Because the mix of cyan, magenta, and yellow toner/ink can only produce muddy brown instead of black, the black toner/ink is added in to make a rich black, and to make color richer and darker. Black toner/ink also costs less than cyan, magenta, and yellow toner/ink, a fact that will occupy our attention in Chapter 5.

Printer CIELab to CMYK color conversion is a map from \mathbb{R}^3 to \mathbb{R}^4 . First, CIELab is mapped to CMY color space using an interpolation technique, then black toner/ink denoted by K, is added in. The amount of cyan, magenta, yellow toner/ink may be adjusted by applying the under color removal (UCR) and the under color addition (UCA) corrections. The typical way of applying UCR is to remove equal amounts of yellow, magenta, and cyan and replace them with the same amount of black. GCR is the result of UCR followed by UCA. However, colors resulting from UCR are dark, less saturated, and hue shifted. UCA is applied to improve the weakness of UCR. Small portions of the three colors of cyan, magenta, and yellow are added back to these areas. The UCA process is empirical and labor intensive due to the mapping from a lower dimensional space to a higher dimensional space, i.e., \mathbb{R}^3 to \mathbb{R}^4 , and the lack of a high fidelity model describing the nonlinear behavior of each toner. It is normally achieved by printing many patches and choosing the best results. In practice, the UCR and UCA correction schemes introduce errors in the color conversion process. For example, one set of data we investigated indicated that the maximum color difference due to the GCR process only was above 14 ΔE in the CIELab color space. The quality of color conversions will be dramatically improved if techniques are developed for the GCR process with theoretically zero color difference.

Because the CIELab to CMYK conversion is a map from \mathbb{R}^3 to \mathbb{R}^4 , there should exist a set of CMYK values corresponding to the same CIELab value. Let g denote a mapping from the CIELab values to the CMYK values, then $\exists\{(C, M, Y, K)^{(i)}, i = 1, 2, \dots, k\}$, s.t., $g^{-1}(C, M, Y, K)^{(i)} = (L, a, b)$,

discrete only in practice, $i = 1, 2, \dots, k$. In practice, we show these level sets of CMYK values can be found through an optimization process if good approximation models are developed for the CMYK to CIELab conversion and the CIELab to CMY conversion. The detailed discussion on lossless gray component replacement is presented in Chapter 5.

1.3.5 Toner Savings in Color Printing

In the highly competitive printing industry, technology to help manage and reduce the cost of printing is one of the key ingredients for success. Hence, reducing toner coverage, while obtaining good printing quality is an ongoing effort for color scientists and engineers. Traditionally, toner savings are achieved by UCR. In four-color (or more) printing, UCR is the process of eliminating amounts of yellow, magenta, and cyan that would have added to a dark neutral (black) and replacing them with black toner/ink during the color separation process. However, as mentioned before, the black ink by itself in a shadow may not be dark enough, and it tends to produce dull-looking images. The UCA is applied to add some amount of cyan, magenta, and yellow back. Thus, toner saving is achieved at the cost of sacrificing the quality of color conversion.

Another application relates to the limitation of the amount of toner/ink coverage on a page to avoid excessive bleed-through. This application is particularly needed for inkjet printing. If the color conversion is performed from CIELab to printer CMYK color space, there often exists some constraints for the amount of cyan, magenta, yellow, and black toner/ink. For example:

$$C + M + Y + K \leq L$$

where L is the total amount of the toner/ink.

This constraint is most likely to affect the colors in highly saturated or dark neutral regions. The typical way of solving the toner/ink coverage problem is to linearly scale or clip the toner/ink percentage to meet the toner limit. The toner/ink coverage problem could be solved via an optimization problem with constraints if a good approximation technique were available and an algorithm proposed. Color conversion with the toner/ink limitation problem is discussed in Chapter 6.

1.3.6 Gamut Mapping Algorithms

Gamut mismatch is a common problem in color reproduction systems since each device has different gamuts, in practice the number of colors in each gamut is generally different. In general input devices

such as digital cameras, scanners, etc., generate more colors than the output devices such as color printers. An example of gamut comparison for an input and an output device are shown in Figure 1.9⁵.

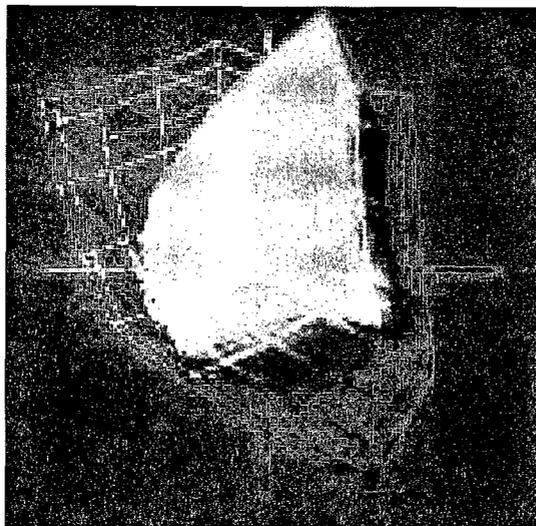


Figure 1.9: Gamut comparison of a monitor (represented by the wire frame) and a printer (represented by the solid shape).

As the result of the gamut mismatch, there are many colors in an image which originated from a RGB system that can not be physically produced by the printer. Figure 1.10 shows that the colors in the Google image⁶ that are outside a specific printer gamut. The outside-gamut colors are marked as white in the lower Google image.

When the input color space is bigger than the gamut of the output color device, gamut-mapping algorithms are applied. The gamut mapping process transforms a point in the source gamut to a realizable color inside the gamut of the output device, also is called destination gamut.

The form of this transformation can dramatically impact the quality of the reproduced images, especially print images. For high-end digital printers, the gamut is relatively small. The maximum error of color conversions normally comes from mapping an outside-gamut color to a inside-gamut color. As such, care needs to be used in the design and implementation of gamut mapping transformations.

⁵This image was downloaded from the Munsell Color Science Lab web site. Reprinted with permission of Musell Color Science Lab.

⁶Google is the trademark of Google.

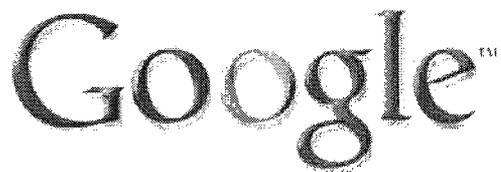
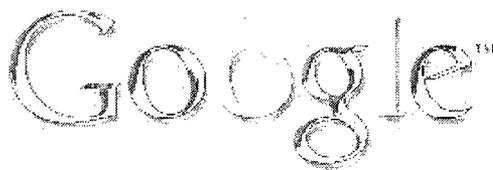
The image shows the word "Google" in its characteristic multi-colored font. The letters are rendered with a high level of detail, showing individual pixels and some color fringing or halos around the edges, which is characteristic of a low-resolution or high-contrast scan of a printed image.This is another instance of the Google logo, similar to the one above, but with slightly different color fringing and pixelation artifacts, illustrating the variability in color reproduction outside a printer's gamut.

Figure 1.10: Colors of Google image outside a printer device gamut.

The first step in the gamut mapping process is to identify a device color gamut boundary. The calculated boundaries are normally connected lines, because they are obtained by either local linear interpolation or by convex hull methods. As we discussed in the previous subsection, none of these methods gives sufficiently accurate results. Evidence strongly suggests that nonlinear approximation techniques are needed to accurately identify the gamut boundaries.

The current gamut-mapping algorithms [33] map all out-of-gamut points directly to the destination gamut. The most typical gamut mappings are clipping and scaling algorithms.

- Clipping algorithms clip out-of-gamut points to the destination gamut boundary.
- Scaling algorithms scale the input color gamut to output color gamut, i.e., some out-of-gamut points are mapped to inside of the destination gamut, some out-of-gamut points are mapped to the boundary of the destination gamut.

The direction of the mapping is the choice of lines along which the mapping is applied. For example, the mapping direction can be a line changing in chroma only at constant lightness and hue. Selecting mapping directions is an active area of research [34]. The mapping directions are decided based on visual experiments. The examples of directions are described as following [35], and illustrated in Figure 1.11⁷.

⁷Reprinted with permission of IS&T: The Society for Imaging Science and Technology sole copyright owners of The Journal of Imaging Science and Technology

In summary:

- The mapping directions may be along the lines of constant perceptual attribute predictors (e.g., constant lightness and hue, constant saturation and hue). The mapping directions of constant lightness and constant saturation are shown in Figure 1.11.
- The mapping directions may be along the lines towards single center-of-gravity (e.g., center is the compression towards $L^*=50$ on lightness axis). The mapping direction of Center is also shown in Figure 1.11.
- The mapping directions may be along the lines towards variable center-of-gravity (e.g., cusp is the compression towards lightness of cusp on lightness axis). See the mapping direction of Cusp shown in Figure 1.11.
- The mapping directions may be along the lines towards the nearest color in the reproduction gamut (e.g. as is the case with minimum ΔE clipping).

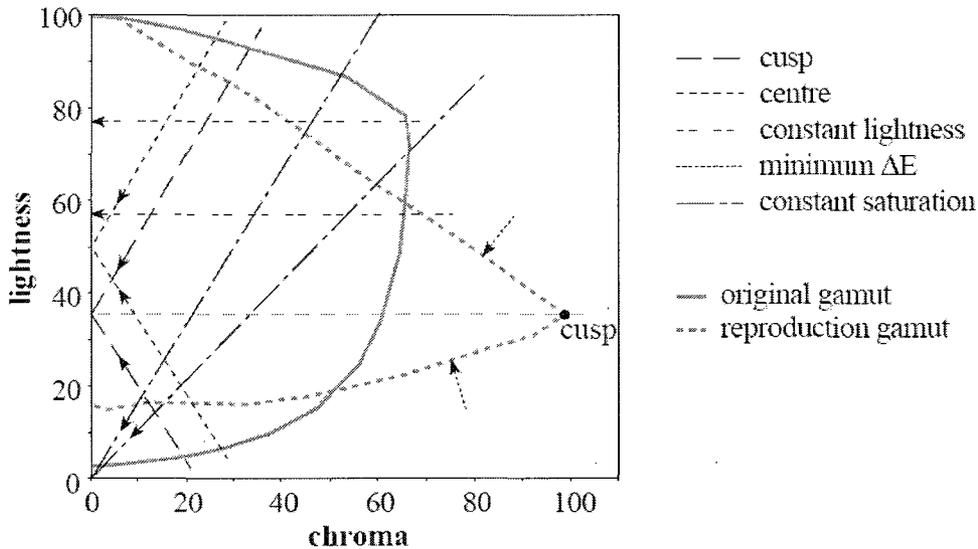


Figure 1.11: Gamut mapping directions [35].

The large variability in past color gamut mapping studies suggests that there is no universal gamut mapping algorithm that works for every device and image. Ideal gamut mapping depends on image content, preservation of perceived hue throughout color space, and the extent of the gamut mismatch in various regions of color space. Image dependent and regional-dependent gamut mappings are preferred. However, image dependent gamut mapping algorithms suffer a performance penalty [44].

It is very clear that current gamut mapping algorithms are more art than science. Relationships among color data developed by a numerical model are not being considered in the gamut mapping process. We believe that the underlying relationship among color data needs to be more accurately retained or transformed rationally after gamut mapping. Thus the best gamut mapping algorithm should be based on the numerical model with guidance from the perceptual model. A detailed discussion on the gamut mapping algorithm based on both numerical and perceptual model is given in Chapter 7.

1.3.7 Printer Color Conversion Quality Assessment

There are two evaluation methods for the color conversion quality: the numerical method and the psychophysics method.

The numerical method consists of quantitatively comparing the color difference between the original color and the printed color. The original colors are the CIELab colors. These colors are converted with the printer color conversion to the CMYK colors. The CMYK colors are then sent to the printer to print, and measured by spectral photometers. The test data are selected in three major areas: inside the printer gamut, near the boundary of the gamut, and outside the printer gamut. The numerical method tests the approximation errors inside color gamut and gamut mapping results out-side the color gamut. The mean error and the maximum error are calculated using either dE_{ab} or dE_{00} for the color conversion inside the printer gamut. The maximum conversion error is often weighted more than the mean error in the evaluation because the significant color difference is more objectionable than the average color difference.⁸

The psychophysics experiments correlate the human perceptions with physical stimuli. A set of test images including pictorial images, vector graphics, and text samples are printed from the printer. The colors in the images include highlight colors, low-light colors, saturated colors, and out-of-gamut colors. Observers look at these images and make perceptual judgements based on the selected psychophysics method. The most used method is paired comparison [15]. Some 10-20 observers are selected to participate in the experiment. Observers are then, for example, asked to compare an image pair converted by different color conversion methods. The original photos are normally presented as the reference. Typical questions are “Which one do you like better?”, “Which one is more similar to the original image?”, etc. The observers’ data are then analyzed and converted to the perceptual values using the Thurstone’s law of comparative judgment [13, 14].

⁸This approach to quality assessment is suggestive of the impact of the use of a uniform approximation criteria in gamut mapping.

The results from both numerical tests and psychophysical experiments provide the comprehensive evaluation for the printer color conversions.

1.4 Goals of the Research

It is believed that the key problem of the current color conversion techniques is the lack of high fidelity nonlinear C^1 smooth approximation functions to describe the nonlinear behavior of toner/ink interactions.

Current approaches for the empirical nonlinear modeling are essentially based on minimizing the mean-square error. While this is satisfactory, indeed appropriate, for many applications, based on the discussions above, we propose that this is not optimal for the printer color conversion problem. Thus, the primary goal of this research is to implement a nonlinear function approximator for scattered data that is better suited to the sensitivity of the HVS. Given our application, we are particularly interested in developing efficient algorithms for uniform approximation as well as the problem of mixed norms, i.e., where we employ different error norms in different color regions.

The main contributions of this dissertation include:

- Novel implementation of Radial Basis Functions (RBFs) using the exchange algorithm for L_∞ approximation in the over-determined least squares setting.
- Extension of the algorithm for L_p norms with an emphasis on L_1 .
- Extension of both Barrodale and Phillips (BP) algorithm for solving dual problem and Bartels and Conn's (BC) algorithm for solving primal Problem to account for printer (toner) related side constraints in saturated color regions.
- Extension of primal algorithms to account for printer (toner) related side constraints in neutral color area.
- The introduction of "mixed" RBFs that satisfy the Haar condition, i.e., approximations that are based on an array of RBFs such as Gaussian, thin plate splines, for a given data set.
- A new approach for lossless GCR process via the implementation of multidimensional optimization techniques, and to find a level set of CMYK values for each given color.
- A new approach for placement of basis functions dictated by the needs of the gamut mapping problem.

The overall goal of this work is an improved algorithm for color conversion that exploits nonlinear approximation methods that incorporate perceptual color differences. The practical objective is to improve color conversion accuracy, efficiency and to optimize the color conversion quality with consideration of ink limitations. We foresee that this work will have significant impact on the color printing industry.

Chapter 2

FITTING NONLINEAR SCATTERED DATA

As discussed in Chapter 1, the printer color conversion problem has been approached mostly by using interpolation techniques [16, 20, 21, 23, 24, 30, 31, 32]. Research results on color conversion indicate that the Voronoi-based algorithm performs better than the triangular-based interpolation algorithm [24]. Most related to our work are the investigations on color conversion via RBFs using interpolation [16, 24]. In [24], Amidror conducted a comprehensive survey on several popular scattered data interpolation techniques used in color conversion. The interpolation techniques described there include triangulation based methods, inverse distance weighed methods, radial basis function methods and natural neighbor methods. One data fitting algorithm was also compared with these interpolation algorithms. He concluded that the radial basis functions (RBFs), in particular Hardy's original multi-quadratic interpolation method, were among the most promising methods in terms of fitting ability and visual smoothness. Also, the least squares data fitting algorithm gave much smoother results than the interpolation algorithm [17, 24]. Artusi also applied the RBFs interpolation technique to the printer color conversion [16]. A small data set was used for training. The method was compared with the polynomial interpolation method. His results indicate that the RBFs algorithm outperforms the polynomial based algorithm.

In this research, we are interested in solving the printer color conversion problems with RBFs using data fitting techniques adapted to L_∞ and L_1 error criteria. In addition, we employ a perceptual color model to guide our implementation of the RBFs. We chose this approach based on the following considerations:

- Scattered data versus data on a grid.
- Interpolation versus over-determined systems.
- Noise presented in the neutral color regions.
- Smoothness of the color conversion model.

All these considerations are related to the perceptual color difference tolerance which are now described.

Theoretically, a color printer is capable of producing up to 4 bytes of colors (1 byte of cyan, magenta, yellow, and black). A subset of these CMYK colors and their associated CIE Lab values are selected for generating the color conversion model. Because of the underlying complicated characteristics of a color printer, a large CMYK data set and their associated CIE Lab values (often above 2000 data points) are selected to generate the color conversion model. The data selection does not only depend on the halftone design which sometimes gives a nonlinear response, but also depends on the perceptual color difference tolerance as the perceptual color difference tolerances vary over different CIE Lab color regions. More data must be selected in these perceptual sensitive regions. Therefore, when these “clouds” of data are selected, the Voronoi-based algorithm is more suitable than the triangular-based algorithm. The RBF approach is suitable for this problem as the clustering algorithms used in the implementation of RBFs can be Voronoi-based.

We also believe that different color conversion models should be generated for different regions based on the perceptual color difference tolerance. The continuity and smoothness of these color conversion models are critical to the overall quality of the color conversion. In addition, the human vision system (HVS) is more forgiving for the color difference presented in a pictorial image (the tolerance threshold could be above $10 \Delta E$); but is more very sensitive to the smoothness and contrast.

Although the noise from the printer, paper, and measurements is small, as discussed in Chapter 1, it is significant for some color areas, e.g., the neutral color region as the perceptual color difference tolerance is small. RBFs provide the flexibility of applying to both the interpolation and approximation data fitting problems. Further, when noise is present, it makes sense to solve over-determined systems. We will provide evidence that this approach is practical and superior for any device color conversion which have nonlinear behavior such as color printers.

In this chapter, we first give an overview of RBFs, and then discuss in detail the data selection algorithm, clustering/center selection algorithm, norm selections, function and parameter selections, and cross validation.

2.1 RBFs Overview

RBFs are popular for interpolating scattered data as the associated system of linear equations is guaranteed to be invertible under very mild conditions on the locations of the data points. Originally, RBFs were introduced as an approximating tool in Broomhead and Lowe [45].

The RBFs are useful for approximating an unknown function from the data in the form

$$f(x) = Ax + a_0 + \sum_{i=1}^N w_i \phi(\|x - c_i\|) \quad (2.1)$$

where generally, we assume $f(x)$ to be continuous and differentiable over a compact domain. $x \in \mathbb{R}^n$, and $f(x) \in \mathbb{R}$. The term Ax represents the linear component of the mapping and the offset a_0 allows for the mean of the data to be non-zero. The nonlinear portion of the map is represented by the superposition of the nonlinear function ϕ weighted by vectors w_i .

As we can see, the RBF-type network has a few attractive features [46]:

- The weight parameters may be determined using linear models (not the c_i though).
- The RBFs can be either local or global.
- The locations of the basis functions may be adapted using different clustering routines. For the color conversion problem, we believe that the Voronoi region, for example, is a good choice as the clustering routine because it can incorporate the color difference model to characterize each cluster with its unique characteristics.
- RBFs do not require that the data lie on any sort of regular grid.
- The function $f(x)$ is continuous and differentiable. These function properties are important for the color conversion problem as we discussed in Chapter 1.

We explore the RBF approach in the context of the printer color conversion problem. Traditionally, the weights w_i are determined by minimizing the mean-square error. However, due to the unique error characteristics in the color conversion problem, this research extends the algorithms for solving RBFs' weights using L_p norm with emphasis on L_∞ norm, L_2 norm, and L_1 norm.

Our approach includes:

- Choosing a unique training/testing data set for the color conversion problem.
- Clustering color data, determining the centers, and optimizing the number of centers.
- Optimizing RBFs and parameter selections.
- Optimizing the L_p norm selections, specifically, L_∞ norm, L_2 norm, and L_1 norm.
- Combining RBFs with linear programming for L_∞ norm and L_1 norm approximations, and employing the duality theorem.
- Employing the perceptual color model to improve the qualitative result.

2.2 Training Set Selection

The color conversion model is built based on the availability of a data set of input-output pairs, i.e., CMYK vs. CIELab. There are many ways to select the training data set. Some select the CMYK colors in regular grids and some select the CMYK colors based on the halftone curves. Almost all of them are determined solely by the modeling techniques. The CMYK data are printed on a color printer, and measured in the CIELab color space. These CIELab values are random within the printer gamut. These data are expressed in the following form:

For the CMYK to CIELab conversion, $f: \mathbb{R}^4 \rightarrow \mathbb{R}^3$

$$\{x^{(i)} = (C, M, Y, K)^{(i)} \in \mathbb{R}^4, i = 1, 2, \dots, m\}$$

$$\{y^{(i)} = (L, a, b)^{(i)} \in \mathbb{R}^3, i = 1, 2, \dots, m\}$$

where m is the number of color pairs in the data set.

For the CIELab to CMYK conversion: $g: \mathbb{R}^3 \rightarrow \mathbb{R}^4$

$$\{x^{(i)} = (L, a, b)^{(i)} \in \mathbb{R}^3, i = 1, 2, \dots, m\}$$

$$\{y^{(i)} = (C, M, Y, K)^{(i)} \in \mathbb{R}^4, i = 1, 2, \dots, m\}$$

Besides these selected color data, care needs to be taken for the color areas where human perceptions are sensitive to the color changes. For example, the neutral color regions, blue/purple color regions, etc. There is a great chance for poor color conversion in these areas if there are not enough data selected and used in the training process. For the printers we experimented on, we found that problems with these selection methods were due to low sampling density in some perceptual sensitive color regions. Figure 2.1 shows that the blue region colors, hue angles from 265-310 degree, are not populated enough when compared to other color regions in a $9 \times 9 \times 9 \times 9$ CMYK data set, while human perception is very sensitive to the hue angle changes in these regions.

2.2.1 Ideal Data Description

The printer color conversions include both CMYK to CIELab and the CIELab to CMYK color conversions. The CIELab to CMYK conversion is achieved by converting the CIELab values to the CMY values first, and then adding black to each CMY value via the GCR process. Therefore, the CMYK set should have enough CMY and K values. It's preferable that the grid points of cyan, magenta, yellow, and black components represent the curve change of each component created by the halftone process. This means that the grids point may not be equally spaced.

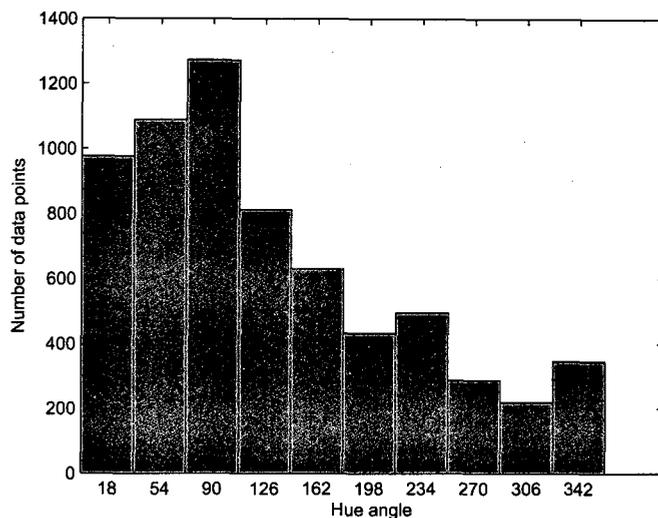


Figure 2.1: A hue angle histogram for a $9 \times 9 \times 9$ regular-spaced CMYK data.

To summarize, for the color conversion problem, a training set should contain scattered data such as:

- The CMY colors.
- The CMYK colors.
- Additional CMY colors in the blue region.
- CMY colors in the neutral color region.
- K ramp.

2.2.2 Data Used in This Thesis

Due to limited printer availability, the data available to this research was not selected by the criteria we described above. Two sets of data from two different printers were collected. Set I contains an equally-spaced $5 \times 5 \times 5 \times 5$ CMYK modeling data and an equally-spaced $9 \times 9 \times 9 \times$ CMYK testing data. We used this data set for our pilot experiment. We describe the experiment in the next subsection. Set II is used for the main experiments in this research. There are 1410 data points in this set include:

- 866 non-equally spaced CMY values.
- 20 non-equally spaced K values.

- 524 non-equally spaced CMYK values.
- 99 data points with hue angles from 265-310.
- 144 data points in the neutral color area.

These data are used for training and validation only. The testing data are collected independently after the models are generated. Therefore, all 1410 data points are used in the CMYK to CIELab conversion, and all 866 CMY data points were used in the CIELab to CMY conversion.

2.3 Cluster/Center Selection

The purpose of data clustering is to identify the locations where basis functions should be placed in the model. In general, RBF algorithms focus on placing functions such that the mean square error is minimized. Due to the uniqueness of perceptual color characteristics, we believe the location of the basis function should be placed in the local regions which have similar properties in terms of human visual tolerances. As discussed before, the CMYK values are device-specific values, and do not directly relate to human perception. So, specifically in our construction of the mapping $g : \mathbb{R}^3 \rightarrow \mathbb{R}^4$ and $f : \mathbb{R}^4 \rightarrow \mathbb{R}^3$, although traditionally, clustering is performed on the domain of the data, we recommend always performing clustering on the CIELab data so the characteristics of the color data can be best described by each cluster. The visual tolerances of color data can be determined using a color difference model. In this research, the most recent color difference model CIEDE2000 is used as the distance metric for each color relative to the center color because the distances are designed to be small.

Before conducting clustering algorithms on the main data (Set II), we conducted a preliminary experiment on a small set of data (Set I) to understand how the number of clusters affects the color conversion accuracy, in other words, we would like to check if there exists an upper bound limit for the number of clusters, i.e., after which the conversion accuracy decreases with the increase of the cluster number.

There are 625 data points in Set I. In this preliminary example, the least squares approximation was applied. The number of centers selected are $\{N = 10, 20, 30, \dots, 400\}$. The LBG algorithm was used to find the centers and clusters [48]. Gaussian, multiquadratic, cubic, inverse multiquadratic, and thin spline functions were also used. The results of our preliminary experiment indicated that RBFs approximation errors on the test data are a function of the number of cluster centers independent of what function was used [51]. The results are shown in Figures 2.2 and 2.3 using the multiquadratic function as the example. In Figure 2.2, the maximum error in the testing data decreases as the number of centers

increase until about 90 – 100. At that point, the error flattens out and actually begins to increase about 250 – 300 centers. A similar behavior is seen in the Figure 2.3 showing the average approximation error for the radial basis function model. For a range of cluster numbers, the color conversion performs equally well.

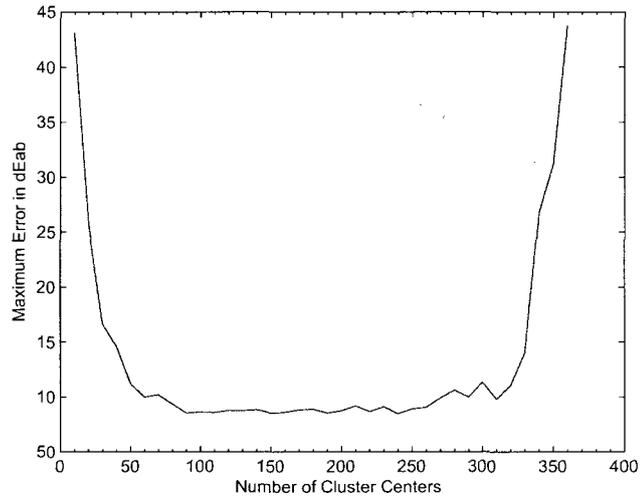


Figure 2.2: Number of cluster centers vs. maximum errors.

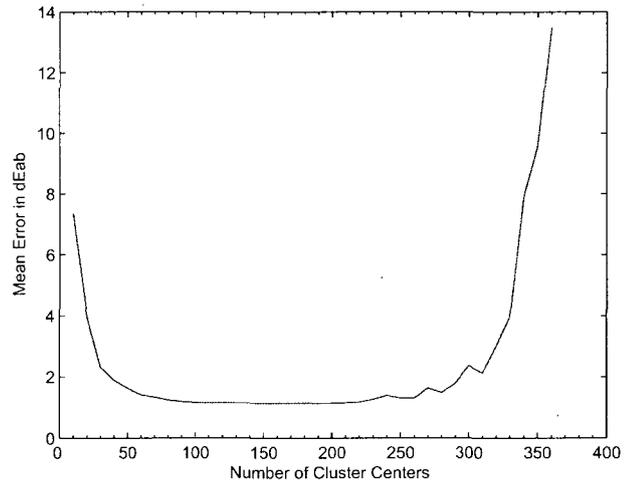


Figure 2.3: Number of cluster centers vs. mean errors.

The determination of the appropriate number of radial basis functions for a given data set is clearly an important problem. In particular, one has to be careful to not retain too many basis functions or the error of the representation will actually degrade on the testing set, even though it may improve on the training set.

Based on the preliminary results, we designed our clustering algorithm by combining a Voronoi-based clustering algorithm with a subset selection algorithm. Because the CIELab data are random, a random center selection algorithm is used as the first step for clustering/center selections. The number of random centers was chosen larger than necessary. Clustering algorithms are then used to identify the clusters, and to refine the center of each cluster. There are several efficient clustering algorithms available including k -means, LBG, and topology-preserving mappings [48, 49]. For our case, they all performed equally well. The LBG clustering algorithm was arbitrarily selected. We remark that this phase of the algorithm is off-line, so we may seek topologically expensive solutions with little penalty. The number of centers is then optimized using subset selection techniques such as the popular orthogonal least squares (OLS) algorithm [52].

In this research, a subset selection algorithm was also developed based on the covariance matrix which out-performed the OLS algorithm. The LBG clustering algorithm and the subset selection algorithms of OLS and covariance matrix are discussed in following subsections.

2.3.1 Clustering Algorithm

The LBG algorithm we use for determining centers in this research is practically the same as that employed by [48, 47]. In general, the domain of the data is clustered into Voronoi regions (Definition 1.3.1) using a global competitive learning algorithm [48].

The algorithm proceeds as follows:

- Select an initial number N_c of centers randomly from the m data points in the training set.
- For each center, compute the Voronoi set, i.e., all points in the data closest in the perceptual color space to this center relative to all other centers.
- Update the centers as the mean of the points in the Voronoi set.
- Repeat until the CIELab values of these centers do not change anymore.

The initial number of centers N_c is determined as $M/4$. We implemented several clustering algorithms including k -means, LBG, and topology-preserving mappings. For this research, they produced very similar results.

2.3.2 Subset Selection Algorithms

The clustering algorithm described above can be effectively combined with a subset selection algorithm to eliminate unnecessary clusters. Two subset selection algorithms were implemented: the standard orthogonal least squares algorithm [52] and a novel covariance matrix algorithm.

Orthogonal Least Squares (OLS) Algorithm

The OLS method for center optimization was originally proposed in [52]. The OLS method serves to identify which of the N centers are most useful in the RBF model and indicates which centers may be deleted from the model and at what expense to accuracy expressed in color difference.

We present here the theory behind this refinement technique in a general setting [46]. The computation of the weight parameters in Equation (2.1) is an over-determined least squares problem. Thus, we seek a solution to the set of inconsistent equations

$$y = \Phi w$$

where each column of the matrix Φ is associated with a single center and f does not actually reside in the column space of Φ . The question then becomes which of the columns of Φ is most useful for solving the problem? In general, one associates a quality function to a center. In this case that means measuring the value of a column ϕ_i of Φ in solving the least squares problem. One measure is the cosine of angle between f and each column of Φ ,

$$v_i = \left(\frac{f^T \phi^{(i)}}{\|f\| \|\phi^{(i)}\|} \right)^2$$

In other words the best center is the one for which v_i is as large as possible, i.e., the angle is as small as possible. This solution is indexed by i^* meaning ϕ_{i^*} is the solution to the optimization problem.

Once the best column (and hence RBF center) of Φ has been determined, the next best center can be computed by projecting the remaining columns along ϕ_{i^*} . This approach may be iterated to obtain a reduced subset of centers that has acceptable modeling accuracy.

The algorithm proceeds as follows:

Let $k = 1, 2, \dots, N_c$

- Step 1: $k = 1$

$$i_1 = \arg \max \left(\frac{f^T \phi^{(i)}}{\|f\| \|\phi^{(i)}\|} \right)^2$$

$$q^{(1)} = \phi^{(i_1)}$$

- Step k:

$$\alpha_{jk}^{(i)} = \frac{q_j^T \phi_i}{(q_j^T q_j)}, \quad 1 \leq j < k$$

$$q_k^{(i)} = \phi_i - \sum_{j=1}^{k-1} \alpha_{jk}^{(i)} q_j$$

$$i_k = \arg \max \left(\frac{f^T q_k^{(i)}}{\|f\| \|q_k^{(i)}\|} \right)^2$$

where $i_k \leq i \leq N_c, i \neq i_1, \dots, i \neq i_{k-1}$

$$q_k = q_k^{i_k} = \phi_{i_k} - \sum_{j=1}^{k-1} \alpha_{jk} q_j$$

- The procedure is terminated at the M th step when

$$1 - \sum_{j=1}^M \left(\frac{f^T q_j}{\|f\| \|q_j\|} \right)^2 < \rho, \quad \text{where } 0 < \rho < 1$$

We note that the performance of OLS is specific to the function selection, and the norm selection in RBFs. In our application the primary goal of clustering data is to identify regions which have the similar perceptual properties. This may explain why OLS was out performed by the algorithm presented below.

Clustering Algorithm Based on Covariance Matrix

A cluster can be described by its mean vector and variance-covariance matrix. The mean vector is the center of the cluster, and the variance-covariance matrix consists of the variances of the color data points along the main diagonal and the covariances between each pair color components in the other matrix positions. For example, if a cluster is in the CMYK color space, the covariance of this cluster provides a measure of the strength of the correlation between any two color components. Let the center CMYK value of a cluster be $\bar{x} = (\bar{C}, \bar{M}, \bar{Y}, \bar{K})$, i.e., $\bar{x}_1 = \bar{C}$, $\bar{x}_2 = \bar{M}$, $\bar{x}_3 = \bar{Y}$, and $\bar{x}_4 = \bar{K}$, and let m_c denote the number of data points in a cluster. The formula for computing each element $COV_{jj'}$ in a 4×4 CMYK covariance matrix for a cluster is:

$$COV_{jj'} = \sum_{i=1}^{m_c} [(x_j^i - \bar{x}_j)(x_{j'}^i - \bar{x}_{j'})] / (m_c - 1) \quad (2.2)$$

where $j, j' = 1, \dots, 4$

When the rank of the covariance matrix is less than the number of the input color components, it means:

- The correlations are not all independent of each other.

- The number of the data points in the cluster is less than the number of color components.

A Gaussian function which can be expressed as the function of the covariance matrix can not be selected if there exists a non-full rank cluster.

Our clustering algorithm is described as following:

- Calculate the covariance matrix of each cluster.
- If the rank of the covariance matrix of a cluster is less than the number of input color components, the center of this cluster is deleted, and the data points in this cluster are moved to nearby clusters.
- Update the center of each cluster, and perform the LBG algorithm to refine each cluster.
- Repeat until every cluster has a full rank covariance matrix.

Results of Algorithm Comparison

Our experiments indicated that the clusters obtained from the covariance matrix method are sufficient for building the color conversion model. We performed both the OLS algorithm and the covariance algorithm using data Set II (1410 data points). The number of the clusters obtained from the OLS is around 160 depending on the function ϕ selection. For the color conversion from the CMYK to CIE Lab, the mean color conversion errors ranged from 1.1 to 1.3, and the maximum errors ranged from 3.2 to 3.7. The number of clusters obtained from the covariance matrix method is 188. Although the number of clusters obtained from the covariance method is larger than the number obtained from the OLS, the color conversion is more robust. The mean color conversion errors ranged from 0.7 to 0.9, and the maximum errors ranged from 2.7 to 3.0 depending on the function selection.

The number of clusters via the covariance matrix method may not be time efficient. However, building the color conversion model is an off-line process. So, the computational expense is not an over-riding concern.

2.4 Function Optimization

The radial basis functions themselves are generally selected so that they satisfy an invertibility condition on the square interpolation problem for the weights [64]. These functions have the property that expansions such as those given by Equation (2.1) represent continuous functions over compact domains. The location of these functions is stipulated by the vector centers $\{c_k\}$ generally scattered over the domain in a manner that reflects the distribution of the data. Both theoretical investigation and practical results suggested that the choice of the nonlinearity $\phi(\cdot)$ is not critical to the performance of

the RBF [52]. However, for our application, the color conversions from CMYK to CIELab, and CIELab to CMYK, we have found the multiquadric function

$$\phi(x) = \sqrt{r^2 + x^2}$$

to be the most accurate model for most of cases. Other functions that performed well include the Gaussian

$$\phi(x) = \exp\left(\frac{-x^2}{r^2}\right)$$

and the inverse multiquadratic function

$$\phi(x) = \frac{1}{\sqrt{r^2 + x^2}}$$

On the other hand, the Gaussian function

$$\phi(x) = \exp[-(x - \mu_j)^T \Sigma_j^{-1} (x - \mu_j)]$$

where μ_j is the mean of a cluster, and Σ_j^{-1} is the covariance matrix of the cluster, and global functions such as the thin plate spline

$$\phi(x) = x^2 \ln x$$

and the cubic

$$\phi(x) = x^3$$

did not perform well on data Set II. The maximum and the mean conversion errors were at least 20% higher than errors generated using the multiquadratic function.

We also mixed RBFs, for example, a multiquadratic and a Gaussian, a thin plate spline, and an inverse multiquadratic, etc,

$$\phi(x) = \alpha \sqrt{r^2 + x^2} + \beta \exp\left(-\frac{x^2}{r^2}\right),$$

$$\phi(x) = \alpha \frac{1}{\sqrt{r^2 + x^2}} + \beta x^2 \ln x$$

The accuracy with these models did not improve over single RBFs on color data Set II. So the combined functions were not being used in later research.

More functions have been suggested recently including BRBF's with compact support due to Wendlan [53, 54, 55]. These functions are polynomials.

2.4.1 Parameter Optimization

The radius r in Gaussian, multiquadratic, and inverse multiquadratic functions needs to be optimized. Our experiments indicated that the multidimensional Newton's method was an efficient optimization method for 2-norm least square settings. In this case, the objective function aims at minimizing the mean error. We describe this method using the color conversion from CMYK to CIELab: $\mathbb{R}^4 \rightarrow \mathbb{R}^3$, i.e.,

$$\begin{aligned} f_L(x_i, r_L) &= A_L x_i + a_{L0} + \sum_{j=1}^n w_{L_j} \phi_L(\|x_i - c_j\|, r_L) \\ f_a(x_i, r_a) &= A_a x_i + a_{a0} + \sum_{j=1}^n w_{a_j} \phi_a(\|x_i - c_j\|, r_a) \\ f_b(x_i, r_b) &= A_b x_i + a_{b0} + \sum_{j=1}^n w_{b_j} \phi_b(\|x_i - c_j\|, r_b) \end{aligned}$$

The problem formulation is:

$$\begin{aligned} E(r_L, r_a, r_b) &= \sum_{i=1}^m [(f_L(x_i, r_L) - y_1^i)^2 + (f_a(x_i, r_a) - y_2^i)^2 + (f_b(x_i, r_b) - y_3^i)^2] \\ E_{min} &= \min E(r_L, r_a, r_b) \end{aligned}$$

Then for each iteration, let

$$\begin{aligned} g^{(k)} &= \begin{pmatrix} \partial E / \partial r_L \\ \partial E / \partial r_a \\ \partial E / \partial r_b \end{pmatrix} \\ F^{(k)} &= \begin{pmatrix} \frac{\partial^2 E}{\partial r_L^2} & \frac{\partial^2 E}{\partial r_L \partial r_a} & \frac{\partial^2 E}{\partial r_L \partial r_b} \\ \frac{\partial^2 E}{\partial r_a \partial r_L} & \frac{\partial^2 E}{\partial r_a^2} & \frac{\partial^2 E}{\partial r_a \partial r_b} \\ \frac{\partial^2 E}{\partial r_b \partial r_L} & \frac{\partial^2 E}{\partial r_b \partial r_a} & \frac{\partial^2 E}{\partial r_b^2} \end{pmatrix} \end{aligned}$$

Then

$$r_{k+1} = r^{(k)} - F(r^{(k)})^{-1} g^{(k)}$$

For every $r^{(k+1)}$, the weighting factor w_j 's are updated, the new function $E(r)$ is obtained. The iteration stops when $E(r)$ does not improve anymore.

Other optimization algorithms can also be used. The radius r and weighting factor w_j 's can also be optimized together, however, because the size of Jacobian matrix F gets bigger, it's more computational expensive to optimize all parameters together.

2.4.2 Weight Determination vs. Norm Selection

The algorithms for RBF data fitting depend on the selection of norms. Traditionally, RBFs are solved employing a L_2 norm to measure the magnitude of the residuals. For our color conversions application, we believe the norm should be selected considering the visual tolerance of the colors in the CIELab color space, and the source of errors in the color conversion process.

We are interested in employing different norms for different regions in the CIELab color space. Particularly, we are interested in the following norms:

- L_1 norm:

$$|x|_1 = \sum_{i=1}^m |x_i|$$

- L_2 norm:

$$|x|_2 = \sqrt{\sum_{i=1}^m x_i^2}$$

- L_∞ norm:

$$|x|_\infty = \max_{1 \leq x \leq m} |x_i|$$

The relationship of errors via L_1 norm, L_2 norm, and L_∞ norm are stated in *Theorem 2.4.1*

Theorem 2.4.1 (Theorem 1.3 in[78])

For all e in $C[a, b]$ the inequalities

$$\|e\|_1 \leq (b-a)^{1/2} \|e\|_2 \leq (b-a) \|e\|_\infty$$

hold.

This theorem indicates that if we are able to find small error via L_∞ norm, then the errors via L_1 norm and L_2 norm are also small.

For our application, $[a, b] = [0, 1]$. The CIELab color space is heuristically chosen in three regions:

- Neutral region: $0 \leq C_{ab}^* \leq 7$.
- Medium color region: $7 < C_{ab}^* \leq 30$.
- Saturated color region: $C_{ab}^* > 30$.

where C_{ab}^* is the chroma of colors in CIELab color space.

For the neutral color region, the visual color difference tolerance is very small. Error from the measurements and machine can affect the accuracy of the color conversion. The L_1 norm is preferred since it is least sensitive to the outliers. For the medium color range, the color difference tolerance is getting bigger, and the errors from the measurements and machine play less of a role affecting the quality of the color conversions. However, we need to control the color conversion error to be less than the magnitude of 2, so the color difference is less visible. The L_2 norm is chosen to minimize the average error. For the saturated color regions, the errors from the machine is very small compared to the color conversion error mainly due to gamut mapping. It's unlikely for the HVS to detect any errors with magnitude less than 2. It's in our best interest to minimize the maximum error due to the color conversion, thus the L_∞ norm is preferred.

The approximation algorithms for solving the weighting factors in RBFs depend upon the norm selection. For the least squares method, the typical algorithms are normal equation, SVD, and QR factorization.

The data fitting problem using the L_∞ or the L_p approximations were studied by G.A. Watson ([86, 87]). However, very little research has been done on high dimensional nonlinear scattered data fitting with L_∞ norm or L_1 norm. Further, we are not aware of any research nor any use of these norms in the color conversion problem. These problems can be solved via linear programming. The methods and algorithms for solving color conversion problems using L_1 and L_∞ norms are discussed in later chapters.

2.4.3 Interior Points vs. Boundary Points

As discussed in the previous chapter, the continuity and smoothness are critical to the quality of the color conversion. When adjacent clusters or two adjacent regions have different mapping functions, care needs to be taken to maintain the smoothness of the color conversion. This may be achieved by including some of the data in the adjacent clusters in the current cluster [46]. In our application, we optimized the overlapping ranges specified as δ 's for each region. The δ 's values are the distance calculated in the CIELab color space. The data regions partitioned for the L_1 , L_2 , and L_∞ approximation are:

- L_1 approximation: $0 \leq C_{ab}^* < 7 + \delta_n$. Intended range: $0 \leq C_{ab}^* \leq 7$.
- L_2 approximation: $7 - \delta_{ml} < C_{ab}^* < 30 + \delta_{mu}$. Intended range: $7 \leq C_{ab}^* \leq 30$.
- L_∞ approximation: $C_{ab}^* > 30 - \delta_s$. Intended range: $C_{ab}^* \geq 30$.

where δ_n is the extended region for the L_1 norm, δ_{ml} and δ_{mu} are the upper and lower extended regions for the L_2 norm, and δ_s is the extended region for the L_∞ norm.

The partitions of the CIELab color space are illustrated in Figure 2.4 where the dotted lines “.” represent the overlapping ranges, and the solid lines represent the original boundary for the neutral, intermediate, and saturated color regions respectively. The lines in blue indicate the neutral region, magenta lines indicate the boundaries of the intermediate color region, and red lines indicate the saturated color region.

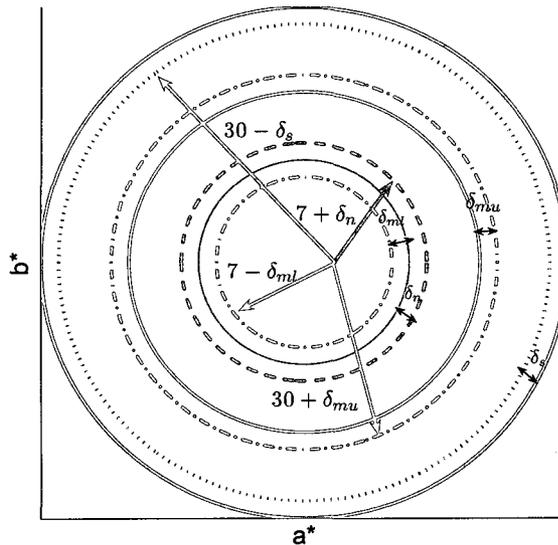


Figure 2.4: The partition of the CIELab color space in the ab^* plane for different norm approximations.

For the CMYK to CIELab conversion, the domain space is the CMYK color space. It is impossible to predict what the CIELab value is for any given CMYK color that is not in the data set II. Therefore the color space is not divided into regions. All 1410 data points are used. For the CIELab to CMY conversion, the CIELab color space is divided into the three regions that are described above using the 866 CMY data. The optimized δ 's values are found in the L_1 , L_2 , and L_∞ experiments in later chapters and sections.

2.4.4 Cross Validation

Validation techniques are used for model accuracy estimation and model selection. The most common techniques are cross validation and bootstrap [56]. Among the cross validation techniques, the K-fold cross validation and the leave-one-out cross validation are the most used methods. For our application, we use the validation technique mainly for model selection. Therefore, we need to balance the absolute accuracy and the bias of variances. The validation method we used for the color conversion application is the K-fold cross validation since it is widely used [57, 58, 59, 60, 61, 62, 63].

In general, several thousands data points are selected for building a color conversion model. We can afford a relatively large K number to ensure the model accuracy even though the data set is large. Different from other cross validation applications, the evaluation of color conversion models involves both numerical analysis and psychophysics analysis. For the numerical analysis, the test data set is collected spanning the input color space. For the perceptual analysis, a number of images are often selected in a visual experiment. At the present stage of the research, due to the printer availability, psychophysics experiments have yet to be conducted.

We conducted experiments to select the K value for the color conversion using L_1 , L_2 , and L_∞ approximations. For the CMYK to CIELab conversion, all 1410 data points in the data set II were used. And for the CIELab to CMY color conversion, the 866 CMY data were used. For the L_2 approximation, the mean error is used as the measure for the model selection. And for the L_1 and L_∞ approximations, the L_1 error and maximum error are calculated instead respectively. For all cases, the variance as a function of the K value is obtained to decide the K value. In this chapter, we concentrate on the K value determination for the least squares approximation. The determination of K values for the L_∞ and L_1 approximation are discussed in Chapter 3 and Chapter 4 respectively

The cross validation technique is described using CMYK to CIELab conversion as the example:

- The modeling set is divided into K-fold partition data sets.
- For each of K experiments, use $K - 1$ folds for training and the remaining one for testing.
- Apply K-fold cross validation to determine the best K value, and best model. We tested all RBF combinations, and optimized the radius r using Newton's method if radius r is a parameter specified in the function.

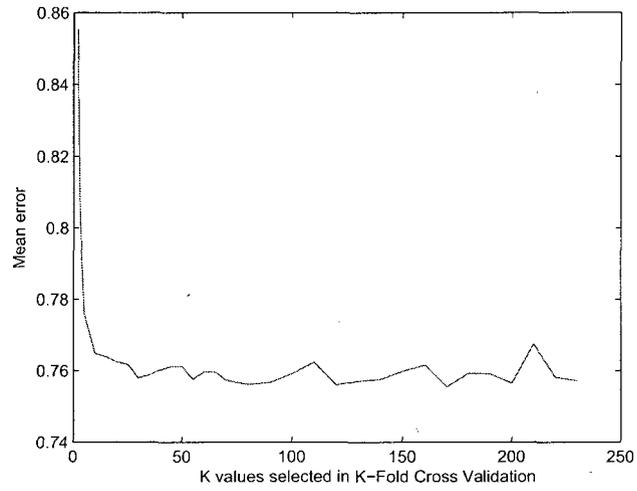


Figure 2.5: K values in the K-fold cross validation versus mean errors.

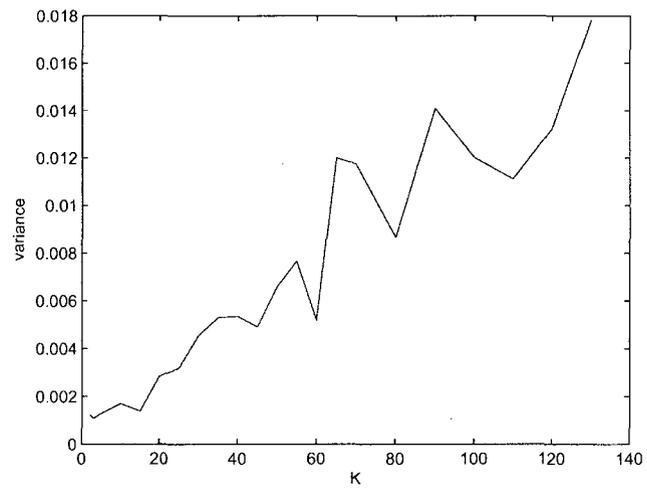


Figure 2.6: K values in the K-fold cross validation versus variance of mean error.

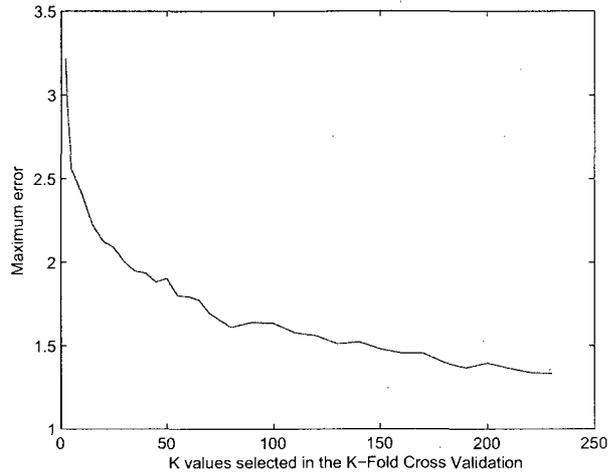


Figure 2.7: K values in the K-fold cross validation versus maximum errors.

Figures 2.5, 2.6, and 2.7 show the mean least squares error, the variance of mean error, and maximum error as functions of the K value for the L_2 approximation. The mean error decreases with the increase of the K value for $K < 10$, and it stays roughly at the same level for $K > 15$. On the other hand, the variance of the mean error stays flat for $K < 15$, and increases with the increase of the K value. It's interesting to find that the maximum error decreases almost monotonically with the increase in the K value. Although the maximum error keeps decreasing and the mean error stays flat for $K \geq 15$, because of increase of the variance, we believe the K value for the least squares approximation for the color conversion problem, based on the 1410 data points, is 15.

2.5 Color Conversions Using the RBF Data Fitting Algorithm in the L_2 Norm

The CMYK to CIELab conversion was performed using the RBF data fitting algorithm in the L_2 norm. In this experiment 1410 data points were used for the CMYK to CIELab color conversion. These data were clustered using the combination of the LBG algorithm and the covariance matrix algorithm which produced 188 clusters. In all 165 clusters were removed.

There are two optimization stages. At first stage we try to find a good starting function and radius for the $\mathbb{R}^4 \rightarrow \mathbb{R}^3$ optimization. We find the best models of $\mathbb{R}^4 \rightarrow \mathbb{R}$ for the CMYK to L^* , a^* , b^* conversions respectively. The Gaussian, multiquadratic, inverse multiquadratic, thin plate spline, and cubic functions were selected. Theoretically, we can make all function combinations, and apply the least squares data fitting algorithm to $\mathbb{R}^4 \rightarrow \mathbb{R}^3$. However, the number of all possible function combinations is huge. Although theoretically it will give us the best solution, we don't believe that the little accuracy we gained in practice justifies the computational expense. Newton's method was used to optimize the

radii for the Gaussian, multiquadratic, and inverse multiquadratic functions. The K-fold cross validation was set to $K = 15$. The color conversion results showed that the Gaussian, multiquadratic, and inverse multiquadratic function always performed better than the thin plate spline and cubic functions, and the multiquadratic function performs the best for all three output components.

In the next modeling stage, we set f_L , f_a , and f_b to multiquadratic with initial r_L , r_a , and r_b obtained from the above experiments. The multidimensional Newton's method was applied to optimize these parameters. The result is that the multiquadratic function with $r_L = 0.4$, $r_a = 0.6$, and $r_b = 0.55$ is the optimized solution for the CMYK to CIELab conversion based on our experiment data.

For the CIELab to CMY conversion, we optimized both data range (δ_{ml} and δ_{mu}) for the least squares approximation and the color conversion model. The algorithm is describe as the following:

- Let $\delta_{ml} = 0$, $\delta_{mu} = 0$, and $k = 0$, then $s_k = \{(L, a, b)_{jk} \mid 7 \leq C_{abjk}^* \leq 30\}$.
 - Perform clustering algorithm to find the clusters.
 - Perform optimization of $\mathbb{R}^3 \rightarrow \mathbb{R}$ for the CIELab to C, M, and Y conversions respectively.
 - Perform the multidimensional Newton's method of $\mathbb{R}^3 \rightarrow \mathbb{R}^3$ for the CIELab to CMY conversion.
 - Record the mean error and maximum error.
- Let $\delta_{ml} = 2.5$, $\delta_{mu} = 2.5$, and let $s_k = \{(L, a, b)_{jk} \mid 7 - 2.5k \leq C_{abjk}^* \leq 30 + 2.5k, k = 1, 2, \dots, \}$.
 - Repeat the above optimization process.
 - Program stops when all 866 data points are included in s_k .

The initial data set contains 365 data points from the 866 CMY values for the least squares approximation, and 64 clusters were obtained. Least square errors were obtained by calculating the differences between the original values and the estimated values. Again, we found that the Gaussian, multiquadratic, and inverse multiquadratic functions out performed the thin plate spline and cubic functions. Our results indicated that the best model is generated with all 866 data points. The optimized models indicated: Gaussian function for the cyan component with $r_C = 0.2$, the inverse multiquadratic for the magenta component with $r_M = 0.5$, and multiquadratic function for the yellow component with $r_Y = 1.2$.

2.6 Summary

In this chapter, we discussed the data fitting technique using RBFs including:

- The clustering algorithm with the combination of the LBG algorithm and the covariance matrix-based algorithm.
- The RBFs selection and Newton's method for the parameter optimization.
- The approximation algorithms with L_p norm with emphasis on L_1 , L_2 , and L_∞ norm for the neutral, intermediate, and saturated color areas.
- The K-fold cross validation method.
- The optimization of the overlapping area between two different models.

We also gave a detailed descriptions how we applied the least squares data fitting algorithm to the color conversions of the CMYK to CIELab and the CIELab to CMY in the intermediate color area.

Our experimental results indicated that the RBFs approximation models are accurate and efficient. The multiquadratic, gaussian and inverse multiquadratic functions performed better than cubic and thin plate spline.

Chapter 3

OVERDETERMINED LINEAR SYSTEMS IN THE L_∞ SENSE

In saturated color areas, the perceptual color difference tolerance is much higher than the magnitude of machine and instrument errors. The data outliers no longer play a significant role in the color conversion. As discussed in Chapter 1, the maximum error is heavily weighted in the color conversion quality assessments. Our goal for the color conversion in this area is thus to reduce the maximum approximation error. Therefore, we are interested in solving the problem via the L_∞ approximation, also referred as the Chebyshev norm. We know from norm inequalities (see Theorem 1.1) that if we control L_∞ norm, then the L_1 and L_2 error will be also small.

There has been considerable research on function approximation using the L_∞ norm ([65, 66, 67, 69, 70, 71, 72, 73, 74, 75, 76, 77]). Many algorithms have been developed for this type of application ([82], [83], etc). They are all locally equivalent to the exchange algorithm [85]. However, there is limited research on scattered data fitting techniques using the L_∞ especially for the large sparse problems [68, 75, 79, 80, 84]. For the large sparse problems, the main technique is to convert the problem to a linear programming problem, and use the linearized subproblem as the basis [85]. Among all the algorithms, Barrodale and Phillips' dual algorithm (BP algorithm) and Bartels, Conn, and Li's primal algorithm (BC algorithm) are the most reputable algorithms for solving overdetermined linear systems $\mathbb{R}^n \rightarrow \mathbb{R}$ in the Chebyshev norm [68, 75]. Jonasson and Jonasson and Madsen made simple modifications on the linearized subproblem to speed up the convergence [79, 80]. The effectiveness of their algorithms has yet to be tested on large-scale problems [85].

The BP dual algorithm was analyzed and compared with the BC primal algorithm in [68]. It was believed that the superiority of the BP dual algorithm was due to the effective choice of a suitable starting point for the exchange algorithm embedded in the BP dual algorithm. It was also believed that the primal approach was also preferable for problems arising from the function approximation if care was taken for the initial data point in the algorithm [68].

The BP and BC algorithms are especially attractive in view of the similarity of their formulations with the RBFs approach. In this chapter, we first present the details of the BP dual algorithm using simplex method, and then the BC primal algorithm using direct search method including:

- Problem formulation in primal and dual forms.
- Discussion of the simplex method in the BP dual algorithm.
- Discussion of the exchange algorithm.
- Discussion of direct search method in the BC primal algorithm.

As the BP and BC algorithms are for $\mathbb{R}^n \rightarrow \mathbb{R}$, later, we discuss the extension of BP dual algorithm for $\mathbb{R}^3 \rightarrow \mathbb{R}^3$ for the color conversion from CIELab to CMY. Although the performance is not an overriding concern for the color conversion problem, we still would like to gain a basic understanding of the algorithm efficiency of the BP and BC methods. We present our observations lastly in this chapter.

3.1 Approximation Problem

The general form of the Chebyshev problem is now described. We use slightly different notations to describe this general form. Later we relate the notations to our application.

Let

$$Ax = b \tag{3.1}$$

where

$$A = [a_1, a_2, \dots, a_n] \in \mathbb{R}^{m \times n}$$

where $m > n \geq 2$ and

$$b^T = [\beta_1, \beta_2, \dots, \beta_m] \in \mathbb{R}^m$$

Our objective is to find the vector $x \in \mathbb{R}^n$ such that

$$\|Ax - b\|_\infty = \max_{1 \leq i \leq m} |a_i^T x - \beta_i| \tag{3.2}$$

is minimized

Following Equation 3.1, let L be a linear space spanned by the functions $\{\phi_i\}$

$$L = \langle \phi_1(z), \phi_2(z), \dots, \phi_n(z) \rangle$$

where each ϕ_i is continuous on $[a, b]$, $j = 1, \dots, n$.

A function $\phi \in L$ can be expressed as

$$\phi(z) = \sum_{j=1}^n \alpha_j \phi_j(z) \in L$$

where $z \in [a, b]$. As we can see, the RBF is a perfect example of such a space.

Given $m > n \geq 2$, data points $\{(z_i, y_i), i = 1, 2, \dots, m\}$, we seek to determine $\alpha = (\alpha_1, \dots, \alpha_n)$ to minimize

$$e(\alpha_1, \dots, \alpha_n) = \left\| y_i - \sum_{j=1}^n \alpha_j \phi_j(z_i) \right\|_{\infty} \quad (3.3)$$

$$= \max_{1 \leq i \leq m} |y_i - \sum_{j=1}^n \alpha_j \phi_j(z_i)| \quad (3.4)$$

To relate the notations in Equation 3.1, we set

$$x = \alpha = \{\alpha_1, \alpha_2, \dots, \alpha_n\}$$

$$A = \begin{pmatrix} \phi_1(z_1) & \phi_2(z_1) & \dots & \phi_n(z_1) \\ \dots & \dots & \dots & \dots \\ \phi_1(z_m) & \phi_2(z_m) & \dots & \phi_n(z_m) \end{pmatrix}$$

and

$$b = (y_1 \quad y_2 \quad \dots \quad y_m)^T$$

3.2 Formulation of Linear Programming Problem

Now we are ready to convert the L_{∞} approximation problem to a standard form of the linear programming.

3.2.1 Primal Problem Formulation

First, we set up the problem in the primal form [68, 75]. Let ξ denote the maximum error.

$$\xi = \max_{1 \leq i \leq m} e_i(\alpha_1, \alpha_2, \dots, \alpha_n)$$

where e is defined in Equation 3.1.

The primal problem is formulated as:

Minimize ξ

subject to

$$\xi + \sum_{j=1}^n \alpha_j \phi_j(z_i) \geq y_i \quad (3.5a)$$

$$\xi - \sum_{j=1}^n \alpha_j \phi_j(z_i) \geq -y_i \quad (3.5b)$$

$\forall i$ where

$$\xi \geq 0$$

and

$\alpha_1, \alpha_2, \dots, \alpha_n$ are unrestricted

3.2.2 Dual Problem Formulation

The standard form of the corresponding dual linear program is transformed from the above primal linear program [88]:

$$\max \sum_{i=1}^m (\underline{\sigma}_i - \underline{\tau}_i) y_i$$

subject to

$$\sum_{i=1}^m (\underline{\sigma}_i - \underline{\tau}_i) \phi_{ji} = 0 \quad (3.6a)$$

$$\sum_{i=1}^m (\underline{\sigma}_i + \underline{\tau}_i) \leq 1 \quad (3.6b)$$

and

$$\underline{\sigma}_i, \underline{\tau}_i \geq 0, \quad i = 1, 2, \dots, m$$

where

$$\phi_{ji} = \phi_j(z_i).$$

3.3 The Exchange Algorithm and Theorems

Before we discuss in detail how the BP dual algorithm and the BC primal algorithm work, we need to discuss the the exchange algorithm and related theorems which almost all the Chebyshev approximation algorithms were based upon.

Let \mathcal{A} be a linear subspace of $\mathcal{C}[a, b]$, $f \in \mathcal{C}[a, b]$, L be any closed subset of $[a, b]$, $p^* \in \mathcal{A}$,

$$e^*(x) = f(x) - p^*(x)$$

and

$$L_M = \{x \in L \mid e^*(x) \text{ obtains extreme values}\}$$

The following lemma provides a necessary and sufficient condition for a minimax solution.

Lemma 3.3.1 (Theorem 7.1 in [78])

p^* is the best approximation from \mathcal{A} to f , iff $\nexists p \in \mathcal{A}$, s.t.,

$$e^*(x)p(x) = [f(x) - p^*(x)]p(x) > 0, \forall x \in L_M$$

i.e., $e^*(x), p(x)$ are of same signs at all extreme points.

The following condition defines a class of functions upon which the Exchange Algorithm can be applied.

The Haar Condition:

- Let \mathcal{A} be an $(n+1)$ -dimensional linear subspace of $C[a, b]$. \mathcal{A} satisfies the Haar condition iff $\forall p \in \mathcal{A}$, the number of roots of the equation

$$\{p(x) = 0; a \leq x \leq b\}$$

is less than the dimension of \mathcal{A} .

- A derived condition states that if $\{\xi_i^* : i = 1, 2, \dots, k\} \subset [a, b], k \leq n$ then $\exists p \in \mathcal{A}$, s.t., p changes signs at $\xi_i : i = 1, 2, \dots, k$ and p has no other roots.

As we can see the RBFs in the form of Equation 2.1 meet the Haar condition because of the linear part in the equation. Now we state the key theorem for the Exchange Algorithm.

Theorem 3.3.2 Minimax Characterization Theorem (MCT. Theorem 7.2 in [78])

Let \mathcal{A} be an $(n+1)$ -dimensional linear subspace of $C[a, b]$ that satisfies the Haar condition, and let f be any function in $C[a, b]$. Then p^* is the best minimax approximation from \mathcal{A} to f , iff, there exist $(n+2)$ points $\{\xi_i^* ; i = 0, 1, \dots, n+1\}$, s.t., the conditions

$$(1) \quad a \leq \xi_0^* < \xi_1^* < \dots < \xi_{n+1}^* \leq b,$$

$$(2) \quad |f(\xi_i^*) - p^*(\xi_i^*)| = \|f - p^*\|_\infty, i = 0, 1, \dots, n+1,$$

and

$$(3) \quad f(\xi_{i+1}^*) - p^*(\xi_{i+1}^*) = -[f(\xi_i^*) - p^*(\xi_i^*)], i = 0, \dots, n \text{ alternative signs}$$

are obtained.

Now we discuss the minimax approximation theorem on a discrete point set:

Theorem 3.3.3 ([78], Theorem 8.2) in [78]

Let \mathcal{A} be a finite-dimension subspace of $C[a, b]$ that satisfies the Haar condition. Let $\{x_i : i = 1, 2, \dots, m\}$ be a set of distinct points from $[a, b]$, where m is not less than the dimension of \mathcal{A} . For any f in $C[a, b]$, let the one-point exchange algorithm be applied to calculate the element of \mathcal{A} that minimizes expression

$$\max_{i=1,2,\dots,m} |f(x_i) - p(x_i)|, p \in \mathcal{A}$$

Then the required approximation to f is obtained in a finite number of iterations.

The color conversion problem is composed of a set of discrete data points. We would like to sketch the main idea of the exchange algorithm for this case. The Exchange Algorithm is based on the MCT. It is illustrated in Figure 3.1.

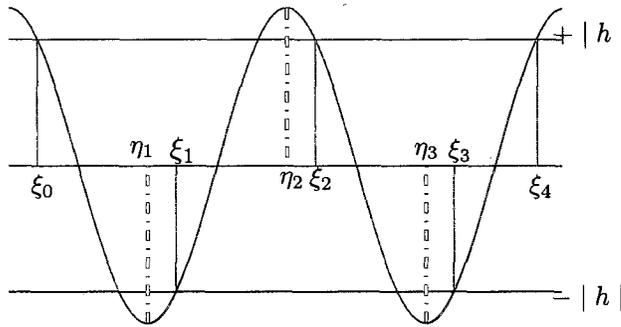


Figure 3.1: Illustration of the main idea of the Exchange Algorithm.

For each iteration k , it adjusts a reference $\{\xi_i : i = 0, 1, \dots, n + 1\}$, then finds a trial approximation p_k , s.t.,

- Condition (3) in the MCT is satisfied.
- Condition (2) in the MCT might not be satisfied.

On a discrete set, i.e., f is known on a set of points $\{x_i, i = 1, 2, \dots, m\}$, to find trial approximation p_k in each iteration, let

$$h = f(\xi_0) - p_k(\xi_0)$$

and solve

$$f(\xi_i) - p(\xi_i) = (-1)^i h, i = 0, 1, 2, \dots, n + 1$$

The iteration stops when p_k satisfies all the conditions in the MCT.

3.4 Barrodale and Phillips' Algorithm for Solving Dual Problem

The BP dual algorithm is based on the standard form of the simplex method composed of three stages. It was believed as the most efficient algorithm for the large scattered data fitting problem. We discuss the algorithm in detail in the following subsections.

3.4.1 The Condensed Tableau in the BP Dual Algorithm

The BP dual algorithm introduces artificial variables α_j in Equation 3.6a and slack variable ξ in 3.6b [75]. The underscores were added to each variable to differentiate the dual variables from the primal. The above dual problem becomes

$$\text{Maximize } \sum_{i=1}^m (\underline{\sigma}_i - \underline{\tau}_i) y_i$$

subject to

$$\sum_{i=1}^m (\underline{\sigma}_i - \underline{\tau}_i) \phi_{ji} + \underline{\alpha}_j = 0, \quad j = 1, 2, \dots, n$$

$$\sum_{i=1}^m (\underline{\sigma}_i + \underline{\tau}_i) + \underline{\xi} = 1$$

$$\underline{\xi}, \underline{\alpha}_j, \underline{\tau}_i \geq 0, \quad i = 1, 2, \dots, m$$

Denoting the column vectors formed by the coefficients of $\underline{\sigma}_i$, $\underline{\tau}_i$, $\underline{\alpha}_j$, and $\underline{\xi}$ by \underline{s}_i , \underline{t}_i , \underline{a}_j , and \underline{w} , then

$$\underline{s}_i + \underline{t}_i = 2\underline{w}$$

Let \underline{b} denote the right-hand side of Equation 3.5, then

$$\underline{b} = \underline{w} = (0 \quad 0 \quad \dots \quad 0 \quad 1)^T = e_{n+1}^T$$

Because of the above relationships, the initial dual formulation is converted to a condensed simplex tableau. The initial tableau is stated in Table 3.1, where $\{\underline{\sigma}_i, \underline{\tau}_i\}$, $i = 1, 2, \dots, m$ are nonbasic variables, and $\{\underline{\alpha}_j, j = 1, 2, \dots, n\}$, and $\underline{\xi}$ are basic variables.

3.4.2 The Three-Stage BP Dual Algorithm

We now discuss each of the three stages in the BP dual algorithm based on the setup of the condensed tableau. The first two stages are to get the efficient starting point for the stage three which is equivalent to the exchange algorithm for the linear minimax approximation.

Table 3.1: Condensed initial simplex tableau.

| Basis | s_1 | s_2 | ... | s_m |
|------------------------|--------------|--------------|-----|--------------|
| $\underline{\alpha}_1$ | $\phi_{1,1}$ | $\phi_{1,2}$ | ... | $\phi_{1,m}$ |
| $\underline{\alpha}_2$ | $\phi_{2,1}$ | $\phi_{2,2}$ | ... | $\phi_{2,m}$ |
| \vdots | | \vdots | | \vdots |
| $\underline{\alpha}_n$ | $\phi_{n,1}$ | $\phi_{n,2}$ | ... | $\phi_{n,m}$ |
| ξ | 1 | 1 | ... | 1 |
| Marginal Cost | $-y_1$ | $-y_2$ | ... | $-y_m$ |

Let Φ denote a $m \times n$ matrix:

$$\Phi = \begin{pmatrix} \phi_{1,1} & \phi_{2,1} & \dots & \phi_{n,1} \\ \phi_{1,2} & \phi_{2,2} & \dots & \phi_{n,2} \\ \dots & & & \\ \dots & & & \\ \phi_{1,m} & \phi_{2,m} & \dots & \phi_{n,m} \end{pmatrix}$$

The rank of Φ is denoted as k .

The First Stage of the BP Dual Algorithm

In the first stage, the first k simplex iterations were performed. The algorithm is to find the maximum residual at each iteration and reduce it to zero. The nonbasic variables that can enter the basis to become basic variables are $\underline{\sigma}_i$'s, and basic variables that can leave the basis to become nonbasic variables are strictly restricted to $\underline{\alpha}_i$'s. Basic variable ξ remains zero in the basis. As the end of stage 1, there are k residues at zero.

- $\underline{\sigma}_t$ that enters the basis corresponds to that with the largest absolute reduced cost. i.e., choosing the largest residual in the primal problem.
- The pivot is selected from this column as the largest absolute value corresponding to an $\underline{\alpha}_j$ in the basis.
- At the t 'th iteration, with the corresponding $\underline{\sigma}_t$ entering the basis, there are t zero residuals. i.e.,

$$y_t - \sum_{j=1}^n \alpha_j \phi_j(z_t) = \xi = 0$$

The Second Stage of the BP Dual Algorithm

In the second stage, the $(k + 1)$ th simplex iteration is performed, the variable ξ is forced to leave the basis, which means the error ξ in the primal problem is increased above zero. At the end of the second stage, the $k + 1$ residuals of magnitude equal to resulting value of ξ .

- The $(k + 1)$ th pivotal column is chosen also corresponding to the largest absolute marginal cost, i.e., equivalent to choosing the largest residual in the primal problem. Exchange σ_j 's and τ_j 's if the marginal cost is negative.
- Care needs to be taken to make $n + 1$ th row a legitimate pivotal row. If there are positive values in the first n elements in the pivotal column, these values need to be changed to negative by adding twice of the corresponding rows to the $n + 1$ th pivotal row, and change the sign of those original rows.

The Third Stage of the BP Dual Algorithm

The third stage is equivalent to the exchange algorithm for a linear minimax approximation.

- The pivotal column is chosen corresponding to the most negative marginal cost.
- The pivotal row is chosen by ratio selection rule [88].
- The iteration continues until all marginal cost is nonnegative.
- Every iteration increases the value of ξ until $\xi = e^*$, where e^* is the minimum value in Equation 3.3.

3.5 Bartels and Conn's Algorithm for Solving Primal Problem

The BP dual algorithm can be expressed equivalently using the interior point method based on the natural primal problem formulation [68]. The authors believed that the primal method was much more nature than the dual method, and was superior to the dual approach when care is taken in the choice of the starting point for the primal approach. To rewrite the primal formulation in Equation 3.6, let

$$c_0 = (1 \ 0 \ 0 \ \dots \ 0)_{n+1}^T$$

$$c_i = \begin{pmatrix} 1 \\ -a_i \end{pmatrix}_{n+1}$$

$$c_{m+i} = \begin{pmatrix} 1 \\ +a_i \end{pmatrix}_{n+1}$$

$$\delta_i = -y_i$$

$$\delta_{m+i} = y_i, i = 1, 2, \dots, m$$

$$v = \begin{pmatrix} \xi \\ \alpha \end{pmatrix}_{n+1}$$

3.5.1 Conversion from a Constrained Problem to an Unconstrained Optimization Problem

The above primal problem is converted to an unconstrained optimization problem via a piecewise linear penalty function. Let $\mu > 0$ be a fixed parameter. Define

$$p(v, \mu) = \mu c_0^T v - \sum_{j=1}^{2m} \min(0, c_j^T v - \delta_j) \quad (3.7)$$

For any arbitrary $v \in \mathbb{R}^{n+1}$, Equation 3.7 can be expanded into

$$\begin{aligned} p(v, \mu) &= \mu c_0^T v - \sum_{j \in I^0} \min(0, c_j^T v - \delta_j) \\ &\quad - \sum_{j \in I^+} \min(0, c_j^T v - \delta_j) \\ &\quad - \sum_{j \in I^-} \min(0, c_j^T v - \delta_j) \end{aligned}$$

where

$$\begin{aligned} I^0 &= I^0(v) = \{j | c_j^T v = \delta_j\} = \{j_1, j_2, \dots, j_k\} \\ I^+ &= I^+(v) = \{j | c_j^T v > \delta_j\} \\ I^- &= I^-(v) = \{j | c_j^T v < \delta_j\} \end{aligned}$$

For any $d \in \mathbb{R}^{n+1}$, and any $\lambda \geq 0$ sufficiently small,

$$p(v + \lambda d, \mu) = p(v, \mu) + \lambda [\mu c_0^T d - \sum_{j \in I^-} c_j^T d - \sum_{j \in I^0} \min(0, c_j^T d)] \quad (3.8)$$

$$= p(v, \mu) + \lambda h^T d + \lambda \sum_{j \in I^0} \sigma_j^- c_j^T d \quad (3.9)$$

where

$$h = \mu c_0 - \sum_{j \in I^-} c_j$$

and

$$\sigma_j^- = \sigma_j^-(d) = \begin{cases} 0 & \text{if } c_j^T d \geq 0 \\ 1 & \text{if } c_j^T d < 0, \end{cases} \quad j \in I^0.$$

Define the matrix

$$N = [c_{j_1}, c_{j_2}, \dots, c_{j_k}]$$

and let

$$P = I - N(N^T N)^{-1} N^T$$

It's clear that matrix P is the orthogonal projector onto the null space of N^T .

3.5.2 Search Direction and Step Size

Now we need to define search direction d such that $p(v + \lambda d, \mu) < p(v, \mu)$. Let

$$d = -Ph \tag{3.10}$$

Case I: $Ph \neq 0$

Apply d defined in Equation 3.10 to Equation 3.9,

$$\begin{aligned} c_j^T d &= -c_j^T Ph \\ &= -c_j^T PPh \\ &= -(Pc_j)^T Ph \\ &= 0 \end{aligned}$$

Since P is the orthogonal projector onto the null space of N^T . The Equation 3.8 becomes:

$$p(v + \lambda d, \mu) = p(v, \mu) + \lambda h^T d \tag{3.11}$$

and

$$h^T d < 0$$

i.e.,

$$p(v + \lambda d, \mu) < p(v, \mu)$$

Thus, d serves as a descent direction for the function p .

Case II: $Ph = 0$

Assuming columns of N are linearly independent, then

$$h = \sum_{i=1}^k \eta_i c_{ji}.$$

Then Equation 3.8 is changed to

$$p(v + \lambda d, \mu) = p(v, \mu) + \lambda \sum_{i=1}^k [\eta_i + \sigma_{ji}^-] c_j^T d$$

Subcase a: $\eta_{i_*} < 0$ for some $i_* \in I^0$

Choose d , s.t.,

$$c_j^T d = 0, j \in I^0, j \neq i_*$$

$$c_{i_*}^T d = 1$$

This implies that $\sigma_j^- = 0, j \in I^0$, and equation (12) becomes to equation (15), and because

$$h^T d = \eta_{i_*} c_{i_*}^T d = \eta < 0$$

Then

$$p(v + \lambda d, \mu) < p(v, \mu)$$

Subcase b: $\eta_{i_} \geq 0$ for all $i_* \in I^0$*

If no constraints are being violated, it is the optimal. Otherwise, reduce μ , and start another iteration.

{choose step size λ }

$$\Lambda^{(1)} \stackrel{def}{=} \left\{ \lambda \mid \lambda = \frac{\delta_j - c_j^T v}{c_j^T d}, \lambda > 0, j \in I^+ \cup I^- \right\}$$

$$\lambda = \min \Lambda^{(1)}$$

update

$$v = v + \lambda d.$$

3.6 The BC Primal Algorithm in Color Conversion

The L_∞ norm approximation is applied to color conversions from CMYK to CIELab and CIELab to CMY. The formulation of this color conversion in the primal form is described in the following using the CIELab to CMY color conversion as the example:

$$\text{Minimize } \xi = \xi_c + \xi_m + \xi_y$$

subject to

$$\xi_c + \sum_{j=1}^{n+4} \alpha_{cj} F_{cj}(z_i) \geq c_i \quad (3.12a)$$

$$\xi_c - \sum_{j=1}^{n+4} \alpha_{cj} F_{cj}(z_i) \geq -c_i \quad (3.12b)$$

$$\xi_m + \sum_{j=1}^{n+4} \alpha_{mj} F_{mj}(z_i) \geq m_i \quad (3.12c)$$

$$\xi_m - \sum_{j=1}^{n+4} \alpha_{mj} F_{mj}(z_i) \geq -m_i \quad (3.12d)$$

$$\xi_y + \sum_{j=1}^{n+4} \alpha_{yj} F_{yj}(z_i) \geq y_i \quad (3.12e)$$

$$\xi_y - \sum_{j=1}^{n+4} \alpha_{yj} F_{yj}(z_i) \geq -y_i \quad (3.12f)$$

where

$$\xi \geq 0$$

$$\alpha_1, \alpha_2, \dots, \alpha_{n+4} \text{ unrestricted}$$

$$z_i = \{L, a, b\}_i, i = 1, \dots, m$$

$$F_{ji} = [1, z_i, \phi_1(z_i), \phi_2(z_i), \dots, \phi_n(z_i)]_{4+n}$$

$$\phi_{ji} = \phi_j(z_i)$$

ϕ_j is one of the RBFs, n is the number of clusters, and m is the number of color data points.

The problem can be rewritten in this form:

$$\text{Minimize } c_0^T v$$

subject to

$$c_j^T v \geq \delta_j, j = 1, 2, \dots, 6m$$

where

$$c_0 = (1 \ 0 \ \dots \ 1 \ 0 \ \dots \ 1 \ 0 \ \dots \ 0)_{3n+15}^T$$

$$v = \begin{pmatrix} \xi_c & \alpha_{c1} & \dots & \alpha_{cn} & \xi_m & \alpha_{m1} & \dots & \alpha_{mn} & \xi_y & \alpha_{y1} \\ \dots & \alpha_{yn} & & & & & & & & \end{pmatrix}_{3n+15}^T$$

$$c_i = (1 \ -\alpha_{ci} \ 0 \ \dots \ 0_{2n+8})_{3n+15}^T$$

$$c_{m+i} = (1 \ \alpha_{ci} \ 0 \ \dots \ 0_{2n+8})_{3n+15}^T$$

$$c_{2m+i} = (0 \ \dots \ 0_{n+4} \ 1 \ -\alpha_{mi} \ 0 \ \dots \ 0_{n+4})_{3n+15}^T$$

$$c_{3m+i} = (0 \ \dots \ 0_{n+4} \ 1 \ \alpha_{mi} \ 0 \ \dots \ 0_{n+4})_{3n+15}^T$$

$$c_{4m+i} = (0 \ \dots \ 0_{2n+8} \ 1 \ -\alpha_{yi})_{3n+15}^T$$

$$c_{5m+i} = (0 \ \dots \ 0_{2n+8} \ 1 \ \alpha_{yi})_{3n+15}^T$$

$$\delta_i = -C_i$$

$$\delta_{m+i} = C_i$$

$$\delta_{2m+i} = -M_i$$

$$\delta_{3m+i} = M_i$$

$$\delta_{4m+i} = -Y_i$$

$$\delta_{5m+i} = Y_i$$

where $i = 1, \dots, m$.

The primal problem is divided into the following steps using the modified BC algorithm:

- Convert the primal problem with constraints to an unconstrained problem via a piecewise linear penalty function. Let $\mu > 0$ be a fixed parameter. Define

$$p(v, \mu) = \mu c_0^T v - \sum_{j=1}^{6m} \min(0, c_j^T v - \delta_j)$$

For any arbitrary $v \in \mathbb{R}^{3n+12}$,

$$\begin{aligned} p(v, \mu) &= \mu c_0^T v - \sum_{j \in I^0} \min(0, c_j^T v - \delta_j) \\ &\quad - \sum_{j \in I^+} \min(0, c_j^T v - \delta_j) \\ &\quad - \sum_{j \in I^-} \min(0, c_j^T v - \delta_j) \end{aligned}$$

where

$$I^0 = I^0(v) = \{j | c_j^T v = \delta_j\} = \{j_1, j_2, \dots, j_k\}$$

$$I^+ = I^+(v) = \{j | c_j^T v > \delta_j\}$$

$$I^- = I^-(v) = \{j | c_j^T v < \delta_j\}$$

- Find sets I^0 , I^+ , and I^- .
- Update search direction and step size as defined in the previous subsection.

3.7 The BP Dual Algorithm in Multidimensional Color Conversion

The formulation of color conversion in dual is described in the following using the CIELab to CMY color conversion as the example:

$$\text{Maximize } \sum_{i=1}^m (\sigma_{ci} - \tau_{ci})c_i + (\sigma_{mi} - \tau_{mi})m_i + (\sigma_{yi} - \tau_{yi})y_i$$

subject to

$$\sum_{i=1}^m (\sigma_{Ci} - \tau_{Ci})F_{C_{j_i}}^T = 0 \quad (3.13a)$$

$$\sum_{i=1}^m (\underline{\sigma}_{Ci} + \underline{\tau}_{Ci}) \leq 1 \quad (3.13b)$$

$$\sum_{i=1}^m (\underline{\sigma}_{Mi} - \underline{\tau}_{Mi}) F_{Mji}^T = 0 \quad (3.13c)$$

$$\sum_{i=1}^m (\underline{\sigma}_{Mi} + \underline{\tau}_{Mi}) \leq 1 \quad (3.13d)$$

$$\sum_{i=1}^m (\underline{\sigma}_{Yi} - \underline{\tau}_{Yi}) F_{Yji}^T = 0 \quad (3.13e)$$

$$\sum_{i=1}^m (\underline{\sigma}_{Yi} + \underline{\tau}_{Yi}) \leq 1 \quad (3.13f)$$

and

$$\underline{\sigma}_i, \underline{\tau}_i \geq 0, \quad i = 1, 2, \dots, m$$

where

$$z_i = \{L, a, b\}_i, \quad i = 1, 2, \dots, m$$

$$F_{ji} = [1, z_i, \phi_1(z_i), \phi_2(z_i), \dots, \phi_n(z_i)]_{n+4}$$

$$\phi_{ji} = \phi_j(z_i)$$

and ϕ_j is one of the RBFs, n is the number of clusters, and m is the number of color data points.

The condensed initial simplex tableau for the CIELab to CMY conversion is shown in the next table.

Table 3.2: Condensed initial simplex tableau for the CIELab to CMY conversion.

| Basis | s_{C_1} | ... | s_{C_m} | s_{M_1} | ... | s_{M_m} | s_{Y_1} | ... | s_{Y_m} |
|----------------|-----------------|-----|-----------------|-----------------|-----|-----------------|-----------------|----------|-----------------|
| α_{C_1} | $F_{C_{1,1}}^T$ | ... | $F_{C_{1,m}}^T$ | 0 | ... | 0 | 0 | ... | 0 |
| \vdots | \vdots | | | \vdots | | | | \vdots | |
| α_{C_n} | $F_{C_{n,1}}^T$ | ... | $F_{C_{n,m}}^T$ | 0 | ... | 0 | 0 | ... | 0 |
| α_{M_1} | 0 | ... | 0 | $F_{M_{1,1}}^T$ | ... | $F_{M_{1,m}}^T$ | 0 | ... | 0 |
| \vdots | \vdots | | | \vdots | | | | \vdots | |
| α_{M_n} | 0 | ... | 0 | $F_{M_{n,1}}^T$ | ... | $F_{M_{n,m}}^T$ | 0 | ... | 0 |
| α_{Y_1} | 0 | ... | 0 | 0 | ... | 0 | $F_{Y_{1,1}}$ | ... | $F_{Y_{1,m}}$ |
| \vdots | \vdots | | | \vdots | | | | \vdots | |
| α_{Y_n} | 0 | ... | 0 | 0 | ... | 0 | $F_{Y_{n,1}}^T$ | ... | $F_{Y_{n,m}}^T$ |
| w_C | 1 | ... | 1 | 0 | ... | 0 | 0 | ... | 0 |
| w_M | 0 | ... | 0 | 1 | ... | 1 | 0 | ... | 0 |
| w_Y | 0 | ... | 0 | 0 | ... | 0 | 1 | ... | 1 |
| Marginal Cost | $-C_1$ | ... | $-C_m$ | $-M_1$ | ... | $-M_m$ | $-Y_1$ | ... | $-Y_m$ |

The simplex procedures are described in the following:

- Find the rank of the matrix $F_{n \times m}$, k .
- Perform the first step BP simplex algorithm: k simplex iterations for each of $F_{C_{n \times m}}$ for cyan component, $F_{M_{n \times m}}$ for magenta component, and $F_{Y_{n \times m}}$ for yellow component.
 - Only allow vectors s_C to enter the basis, and only vectors α_C can leave the basis for $F_{C_{n \times m}}$;
 - Only allow vectors s_M to enter the basis, and only vectors α_M can leave the basis for $F_{M_{n \times m}}$;
 - Only allow vectors s_Y to enter the basis, and only vectors α_Y can leave the basis for $F_{Y_{n \times m}}$;
- Perform the second step BP simplex algorithm: Force ξ_C , ξ_M , and ξ_Y to leave the basis. The corresponding vectors entering the basis are s_C , s_M , and s_Y respectively.
 - If the corresponding marginal cost of s_C , s_M , or s_Y is positive, then s_i is changed to t_i using

$$t_i = 2w_{3n+4} - s_i$$

where

$$w_C = (0 \ 0 \ \dots \ 0 \ 1 \ 0 \ 0 \ 0)^T = e_{3n+1}^T$$

$$w_M = (0 \ 0 \ \dots \ 0 \ 0 \ 1 \ 0 \ 0)^T = e_{3n+2}^T$$

$$w_Y = (0 \ 0 \ \dots \ 0 \ 0 \ 0 \ 1 \ 0)^T = e_{3n+3}^T$$

- If s_i is the pivotal column, r is the row vector of pivotal row. For any j th row vector r_j containing $s_{ij} > 0$, where $j \neq 3n + 1$ for $F_{C_{n \times m}}$, $j \neq 3n + 2$ for $F_{M_{n \times m}}$, and $j \neq 3n + 3$ for $F_{Y_{n \times m}}$, change r to $r + 2r_j$, and change r_j to $-r_j$.

- Perform the third step BP simplex iterations until all the marginal costs are non-negative.

3.8 Experiments and Results

The experiments for the L_∞ approximation were designed to answer the following questions:

- What is the reasonable K value for the cross validation?
- What is the optimized color range, i.e., the δ_s value, for the CIELab to CMY color conversion using the L_∞ approximation?
- Is the L_∞ approximation an affective method for reducing the maximum color conversion errors comparing to the maximum error obtained by the least squares approximation?
- Which algorithm is a more efficient algorithm between the BP dual algorithm and the BC primal algorithm for the color conversion problem?

To answer the first three questions, the BP dual algorithm was performed on the CMY to CIELab conversion. All 866 CMY data points were used with 64 cluster centers. We chose the CMY to CIELab conversion because the same data set should be used for the CIELab to CMY color conversion, and the error calculated in the CIELab is more meaningful than that was calculated in the CMYK color space. Notice that, when we convert the minimax problem to a linear programming problem, the parameter optimization for the function with radius became manual and empirical, i.e., we need to run the problem with a set of radius values to determine which parameter is the best for the application.

The maximum error and its variance with respect to the cross validation K value are plotted in Figure 3.2. Different from the results of the least squares approximation, large K value is needed for the L_∞ approximation. Figure 3.2 shows that maximum error decreases for $K \leq 60$, and it stays relatively

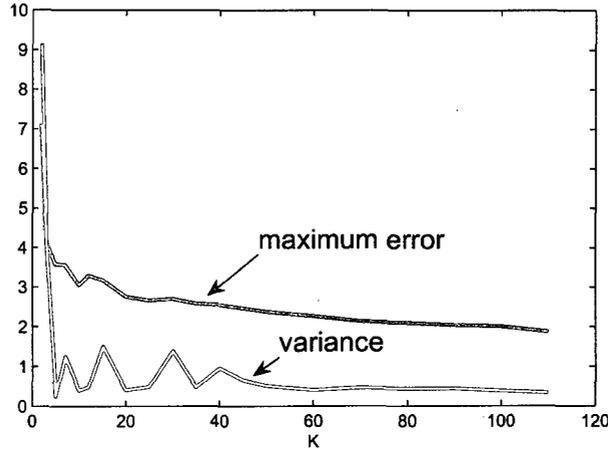


Figure 3.2: The maximum error and its variance as function of cross validation K value.

flat (or decreases very slowly) for $K > 60$. The variance of the maximum errors also stays relatively stable for $K > 60$.

Similar to the experiment we conducted to determine δ_{ml} and δ_{mu} for the least squares approximation, we applied the BP dual algorithm on the CMY data with a series of δ_s value. Again, we found that using all 866 data is the best color data range for the color conversion problem with the L_∞ approach.

Our results also indicated that the maximum error for CMY to CIELab conversion is 2.83 using least squares approximation, and the maximum error is reduced to 2.25 using the L_∞ approximation.

We then implemented both the BC and the BP algorithms on the CIELab to CMY conversion. The 866 CMY data points in the data set II were used. Our results indicated that although both approaches gave similar results, the BP dual algorithm performs much faster than the BC primal algorithm for our application based on the 866 data points. The CPU time of running the BP dual algorithm is around 210 seconds calculated by the MATLAB CPU routine. The CPU time of running the primal algorithm is about 4-6 times more than that running the BP dual algorithm depending on the function selection. By all means, our experiment was not designed for the performance analysis. It's not our intention to make any performance judgment on the BP and BC algorithms. We only use this result to decide what algorithm we use for the L_1 approximation and the color conversion with toner saving problem.

3.9 Summary

The RBFs based L_∞ approximation techniques were applied to the color conversion in the saturated color regions to reduce the maximum errors. The modified multidimensional primal and dual algorithms

based on the BC primal and BP dual algorithms were implemented and tested. Both algorithms were based on the exchange theorem. Our results indicated that the BP dual and the BC primal algorithms gave very similar results. The maximum color conversion error was sufficiently reduced. Based on our experimental color data, the BP dual algorithm performed much faster than the BC algorithm. However, our experiment was not designed for the performance analysis.

Chapter 4

COLOR CONVERSION USING THE L_1 NORM APPROXIMATION

4.1 Neutral Color Conversion Problem

Color conversion in neutral color areas plays a critical role in color reproduction quality, yet the task is difficult. Firstly, the human visual system (HVS) is very sensitive to color changes in neutral color areas. Secondly, the color conversion errors due to machine stability, paper uniformity, and instrument accuracy are more profound relative to the human color difference tolerances. Thus, there are more chances for the existences of outliers. The main goal for building a color conversion model in these areas is to minimize the negative impact of outliers. Therefore the L_1 norm approximation is the natural choice because of its robustness to outliers.

There have been many studies concerning the L_1 solutions to the overdetermined linear system. In early 70's, Barrodale and Roberts presented a simplex algorithm in the primal form to bypass a few iterations [89]. Prior to this algorithm, a few attempts were made to solve this problem in the dual form when m is large [90, 91, 92]. Barrodale conducted an empirical study indicating that solving the primal problem is more efficient [93]. In late 70's Bartels, Conn, and Sinclair presented a technique for solving the L_1 approximation problem by minimizing piecewise differentiable functions [94]. Armstrong and Hultz also presented a technique for a special purpose algorithm [99]. The results of computer comparisons demonstrated that Barrodale and Roberts' algorithm was one of the most efficient algorithms. Later, a simplex version of this algorithm based on a LU decomposition was made by Armstrong, Frome, and Kung further enhancing it [100]. In 80's, Bloomfield and Steiger made further modifications on the Barrodale and Roberts' algorithm by employing a steepest edge criterion [101]. Besides Bartels, Conn, and Sinclair's algorithm, there were also a few other attempts on the direct decent approach [95, 96, 97]. Comparisons of L_1 approximation algorithms were made [98, 102, 103]. Although there was no consensus on which method performed most efficiently, research suggested that both approaches were equivalent, only the implementation details were different [104, 105].

Among all the L_1 algorithms, we selected the Barrodale and Roberts' primal algorithm (BR algorithm) for its simplicity and robustness.

In this Chapter, we discuss:

- The general L_1 approximation problem for overdetermined linear systems.
- The theorems for the discrete L_1 approximation and why the best discrete L_1 approximation calculation is a linear programming problem.
- Barrodale and Roberts' algorithm for solving the primal problem.
- L_1 approximation in color conversion.
- Results of color conversion via the L_1 norm approximation.

4.2 Overdetermined Linear Systems in the L_1 Sense

Similar to the problem setup for the L_∞ , the overdetermined linear system of the general L_1 approximation problem is stated as follows:

$$Ax = b \tag{4.1}$$

where

$$A = [a_1, a_2, \dots, a_n] \in \mathbb{R}^{m \times n}, \quad (m > n \geq 2)$$

and

$$b^T = [\beta_1, \beta_2, \dots, \beta_m] \in \mathbb{R}^m$$

Our objective for the L_1 norm approximation is to find vector $x \in \mathbb{R}^n$ s.t.,

$$\|Ax - b\|_1 = \sum_{i=1}^m \sum_{j=1}^n |a_j^T x_i - \beta_i| \tag{4.2}$$

is minimized

Let L be a linear space spanned by function ϕ_i

$$L = \langle \phi_1(z), \phi_2(z), \dots, \phi_n(z) \rangle$$

where each ϕ_i is continuous on $[a, b]$, $i = 1, \dots, n$. A function ϕ can be expressed as

$$\phi(z) = \sum_{i=1}^n \alpha_i \phi_i(z) \in L$$

where $z \in [a, b] \subset \mathbb{R}^k$.

Given $m > n \geq 2$, data points (z_i, y_i) , determine $\alpha = (\alpha_1, \dots, \alpha_n)$ to minimize

$$e(\alpha_1 \dots \alpha_n) = \left\| y_i - \sum_{j=1}^n \alpha_j \phi_j(z_i) \right\|_1 \quad (4.3)$$

$$= \sum_{i=1}^m |y_i - \sum_{j=1}^n \alpha_j \phi_j(z_i)| \quad (4.4)$$

In Equation 4.1,

$$A = \begin{pmatrix} x = \alpha \\ \phi_1(z_1) & \phi_2(z_1) & \dots & \phi_n(z_1) \\ \dots & \dots & \dots & \dots \\ \phi_1(z_m) & \phi_2(z_m) & \dots & \phi_n(z_m) \end{pmatrix}$$

and

$$b = (y_1 \quad y_2 \quad \dots \quad y_m)^T$$

If we defined the function in the RBF form, then it becomes:

$$\phi(z) = \alpha_0 + \sum_{p=1}^k \alpha_p z(p) + \sum_{j=k+1}^{n+k} \alpha_j \phi(\|z - c_j\|) \quad (4.5)$$

where $x = (\alpha_0, \alpha_1, \dots, \alpha_k, \alpha_{k+1}, \dots, \alpha_{n+k})$.

A becomes

$$A = \begin{pmatrix} 1 & z(1) & z_1(2) & \dots & z_1(k) & \phi_1(z_1) & \phi_2(z_1) & \dots & \phi_n(z_1) \\ \dots & \dots \\ 1 & z_m(1) & z_m(2) & \dots & z_m(k) & \phi_1(z_m) & \phi_2(z_m) & \dots & \phi_n(z_m) \end{pmatrix}$$

4.3 The Theory of the Best L_1 Approximation

It's obvious that the L_1 and L_∞ norms in $C[a, b]$ and in \mathbb{R}^n are not strictly convex. Let \mathcal{A} be a linear subspace of either the L_1 or the L_∞ normed linear space, then the uniqueness of best approximations from \mathcal{A} to f depends on properties of \mathcal{A} and f .

Let p^* in \mathcal{A} be the best L_1 approximation function to f . Define a sign function as:

$$s^*(x) = \begin{cases} -1, & f(x) < p^*(x) \\ 0, & f(x) = p^*(x) \\ 1, & f(x) > p^*(x) \end{cases} \quad (4.6)$$

The following theorem gives the basic necessary and sufficient condition for the function p^* to be a best L_1 approximation from \mathcal{A} to f .

Theorem 4.3.1 (Theorem 14.1 in [78])

Let \mathcal{A} be a linear subspace of $\mathcal{C}[a, b]$. Let f be any function in $\mathcal{C}[a, b]$, and let p^* be any element of \mathcal{A} , such that the set

$$\mathcal{L} = \{x : f(x) = p^*(x), a \leq x \leq b\}$$

is either empty or is composed of a finite number of intervals and discrete points. Then p^* is a best L_1 norm approximation from \mathcal{A} to f , iff, the inequality

$$\left| \int_a^b s^*(x)p(x)dx \right| \leq \int_{\mathcal{L}} |p(x)|dx \quad (4.7)$$

is satisfied for all p in \mathcal{A} , where s^* is the function 4.6.

A similar theorem for discrete L_1 approximation is stated below:

Theorem 4.3.2 (Theorem 15.2 in [78])

Let the function values $\{f(x_t); t = 1, 2, \dots, m\}$, and fixed positive weights $\{w_t; t = 1, 2, \dots, m\}$ be given. Let \mathcal{A} be a linear space of functions that are defined on the point set $\{x_t; t = 1, 2, \dots, m\}$. Let p^* be any element of \mathcal{A} . Let \mathcal{E} contain the points of $\{x_t; t = 1, 2, \dots, m\}$ that satisfy the condition

$$p^*(x_t) = f(x_t) \quad (4.8)$$

and let s^* be the sign function

$$s^*(x) = \begin{cases} -1, & f(x_t) < p^*(x_t) \\ 0, & f(x_t) = p^*(x_t) \\ 1, & f(x_t) > p^*(x_t) \end{cases} \quad (4.9)$$

$t = 1, 2, \dots, m$. Then p^* is the function in \mathcal{A} that minimizes the expression

$$\sum_{t=1}^m w_t |f(x_t) - p(x_t)|, p \in \mathcal{A} \quad (4.10)$$

iff, the inequality

$$\left| \sum_{t=1}^m w_t s^*(x_t) p(x_t) \right| \leq \sum_{x_t \in \mathcal{L}} w_t |p(x_t)| \quad (4.11)$$

holds for all p in \mathcal{A}

Theorem 4.3.2 implies that in order to get the best discrete L_1 approximation p^* , the inequality in 4.11 has to be tested to be satisfied for every single element p in \mathcal{A} . In real applications, it's generally not practical. The next theorem shows that an optimal function p^* can be obtained by searching for suitable interpolation points in \mathcal{L} .

Theorem 4.3.3 (Theorem 15.3 in [78])

Let the function values $\{f(x_t); t = 1, 2, \dots, m\}$ and fixed positive weights $\{w_t; t = 1, 2, \dots, m\}$ be given. Let \mathcal{A} be a linear subspace of \mathbb{R}^m , where the component of each vector p in \mathcal{A} have the values $\{p(x_t); t = 1, 2, \dots, m\}$. Then there exists an element p^* in \mathcal{A} , that minimizes expression 4.10, and that has the property that the zero vector is the only element p in \mathcal{A} that satisfies the conditions $\{p(x_t) = 0; x_t \in \mathcal{L}\}$, where the set \mathcal{L} is defined in Theorem 4.3.2

With Theorem 4.3.3, we are able to show that the best discrete L_1 approximation calculation is a linear programming problem. Let $\phi = \{\phi_i; i = 0, 1, \dots, n\}$ be a basis of the space \mathcal{A} of approximation, the expression 4.11 is changed to

$$\sum_{t=1}^m w_t |y_t - \sum_{i=0}^n \lambda_i \phi_i(x_t)| \quad (4.12)$$

where y_t is the output value corresponding to each x_t . Define the bounds $\{u_t \geq 0\}$ and $\{v_t \geq 0\}$ so that

$$-v_t \leq y_t - \sum_{i=0}^n \lambda_i \phi_i(x_t) \leq u_t \quad (4.13)$$

for $\{t = 1, 2, \dots, m\}$. Then the problem is changed to a linear programming problem:

$$\text{Minimize } \sum_{t=1}^m w_t (u_t + v_t)$$

subject to

$$\begin{aligned} -v_t \leq y_t - \sum_{i=0}^n \lambda_i \phi_i(x_t) \leq u_t, \quad t = 1, 2, \dots, m \\ u_t \geq 0 \\ v_t \geq 0 \end{aligned}$$

for $t = 1, 2, \dots, m$.

4.4 Barrodale and Roberts' Algorithm (BR algorithm) for Discrete L_1 Linear Approximation

Because of the success of using the Barrodale's simplex L_∞ algorithm in solving color conversion problem, again, we chose the Barrodale and Roberts' simplex algorithm for solving the color conversion problem via the L_1 approximation. The algorithm is described as follows:

Let

$$f_i - \sum_{j=1}^n \alpha_j \phi_{j,i} = u_i - v_i$$

where $\{u_i, v_i; i = 1, 2, \dots, m\}$ are nonnegative variables. We define nonnegative variables $\{b_j, c_j; j = 1, 2, \dots, n\}$, and the weights $\alpha_j = b_j - c_j$.

The problem formulation of the primal problem is

$$\text{Minimize } \sum_{i=1}^m (u_i + v_i)$$

subject to

$$f_i = \sum_{j=1}^n (b_j - c_j) \phi_{j,i} + u_i - v_i, \quad i = 1, 2, \dots, m$$

and

$$b_j, c_j, u_i, v_i \geq 0$$

The algorithm is implemented in two stages:

- Stage 1: For the first n iterations, the pivotal columns are restricted to b_j and c_j only. The vector entering the basis is the one with the largest nonnegative marginal cost, i.e., $\sum_{i=1}^m \phi_{j,i}$. The vector leaving the basis is chosen among v_i and u_i for the one causing the maximum reduction in the objective function.
- Stage 2: Interchange the nonbasic u_i or v_i with the basic u_i or v_i . Neither b_j and c_j are allowed to leave the basis. The criteria for selecting vectors leaving and entering the basis are the same as that in the Stage 1.
- Interchange basic vectors b_j or c_j with the corresponding nonbasic vectors c_j or b_j if the final tableau at the end of Stage 2 is infeasible.

The key element of making this algorithm efficient is to choose the vector leaving the basis that causes the maximum reduction in the objective function, rather than go through every vertex point in the problem.

4.5 BR Algorithm for Solving the Multidimensional Neutral Color Conversion Problem

We applied the BR algorithm to the neutral color conversion. Let $z_i = \{(L, a, b)_i \mid i = 1, 2, \dots, m\}$.

The color conversion from CIELab to CMY is formulated in the following:

$$\text{Minimize } \xi = \sum_{i=1}^m u_{c_i} + v_{c_i} + u_{m_i} + v_{m_i} + u_{y_i} + v_{y_i}$$

subject to

$$\sum_{j=1}^{n+4} (a_{c_j} - b_{c_j}) F_{c_j}(z_i) + u_{c_i} + v_{c_i} = C_i \quad (4.14a)$$

$$\sum_{j=1}^{n+4} (a_{m_j} - b_{m_j}) F_{m_j}(z_i) + u_{m_i} + v_{m_i} = M_i \quad (4.14b)$$

$$\sum_{j=1}^{n+4} (a_{yj} - b_{yj}) F_{yj}(z_i) + u_{yi} + v_{yi} = Y_i \quad (4.14c)$$

where

$$a_{cj}, b_{cj}, a_{mj}, b_{mj}, a_{yj}, b_{yj}, u_i, v_i \geq 0$$

$$F_{ji} = [1, z(i), \phi_1(z_i), \phi_2(z_i), \dots, \phi_n(z_i)]_{n+4}$$

for C, M, and Y components respectively.

When performing the two-stage BR algorithm, the vectors relating to C, M, and Y can only be exchanged with the vectors of C, M, and Y respectively.

The condensed initial simplex tableau for CIELab to CMY conversion is shown in the next table.

Table 4.1: Condensed initial simplex tableau for the CIELab to CMY conversion in the L_1 sense.

| Basis | R | b_{C_1} | \dots | b_{C_n} | b_{M_1} | \dots | b_{M_n} | b_{Y_1} | \dots | b_{Y_n} |
|---------------|----------------------------------|--------------------------|----------|--------------------------|--------------------------|----------|--------------------------|--------------------------|----------|--------------------------|
| u_{C_1} | C_1 | $F_{C_1,1}$ | \dots | $F_{C_n,1}$ | 0 | \dots | 0 | 0 | \dots | 0 |
| \vdots | \vdots | \vdots | \vdots | \vdots | \vdots | \vdots | \vdots | \vdots | \vdots | \vdots |
| u_{C^m} | C^m | $F_{C_1,m}$ | \dots | $F_{C^m,m}$ | 0 | \dots | 0 | 0 | \dots | 0 |
| u_{M_1} | M_1 | 0 | \dots | 0 | $F_{M_1,1}$ | \dots | $F_{M_n,1}$ | 0 | \dots | 0 |
| \vdots | \vdots | \vdots | \vdots | \vdots | \vdots | \vdots | \vdots | \vdots | \vdots | \vdots |
| u_{M^n} | M^n | 0 | \dots | 0 | $F_{M_1,n}$ | \dots | $F_{M^n,m}$ | 0 | \dots | 0 |
| u_{Y_1} | Y_1 | 0 | \dots | 0 | 0 | \dots | 0 | $F_{Y_1,1}$ | \dots | $F_{Y_n,1}$ |
| \vdots | \vdots | \vdots | \vdots | \vdots | \vdots | \vdots | \vdots | \vdots | \vdots | \vdots |
| u_{Y^n} | Y^n | 0 | \dots | 0 | 0 | \dots | 0 | $F_{Y_1,n}$ | \dots | $F_{Y^n,m}$ |
| Marginal Cost | $\sum_{i=1}^m (C_i + M_i + Y_i)$ | $\sum_{i=1}^m F_{C_1,i}$ | \dots | $\sum_{i=1}^m F_{C^n,i}$ | $\sum_{i=1}^m F_{M_1,i}$ | \dots | $\sum_{i=1}^m F_{M^n,i}$ | $\sum_{i=1}^m F_{Y_1,i}$ | \dots | $\sum_{i=1}^m F_{Y^n,i}$ |

4.6 Experiments and Results

Similar to the L_∞ approximation, the experiments for the L_1 approximation were designed to determine the K value for the cross validation, the optimized color range (i.e., the δ_n value) for the CIELab to CMY color conversion, and the best L_1 approximation model for the CIELab to CMY conversion.

Firstly, the BR primal algorithm was performed on the CMY to CIELab conversion. All 866 CMY data points were used. We chose the CMY to CIELab conversion because the same data set should be used for the CIELab to CMY color conversion, and the error calculated in the CIELab is more meaningful than that was calculated in the CMYK color space.

The mean error and its variance with respect to the cross validation K value are plotted in Figure 4.1.

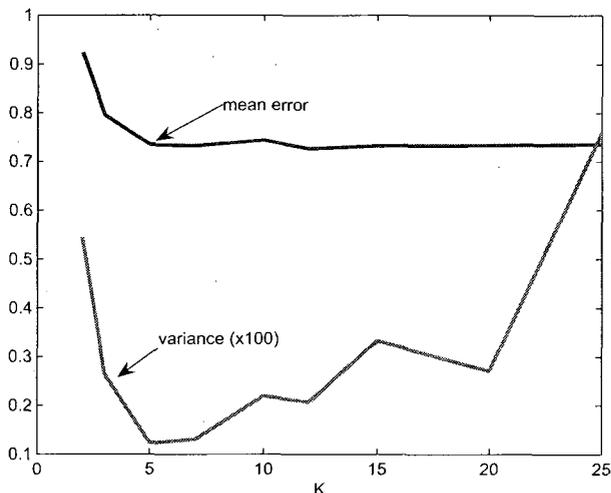


Figure 4.1: The mean error and its variance as a function of cross validation K value.

The results are similar to the cross validation results of the least squares approximation. The mean error decreases with the increase of the K value until $K = 12$. And the mean error stays relatively unchanged for $K > 12$. On the other hand, the variance decreases until $K = 7$, and increases for $K > 7$. As this cross validation experiment is conducted for the model selection, we need to balance between the approximation error and the magnitude of the variance. $K = 12$ is selected for the K -fold cross validation.

Secondly, we repeated the experiments in the L_∞ and L_2 approximations to determine the optimized color range by finding δ_n value. Again, we found that the best accuracy was obtained when all 866 data was used for modeling.

Lastly, we performed the L_1 norm optimization process for the CIE Lab to CMY conversion. Again similar to the L_∞ simplex method, when we convert the L_1 problem to a linear programming problem, the parameter optimization for the function with radius became manual and empirical, i.e., we need to run the problem with a set of radius values to determine which parameter is the best for the application. Our experiments indicate that the BR algorithm is an efficient algorithm for reducing the color conversion errors in the L_1 norm, and the best model for the CIE Lab to CMY conversion is the Gaussian function with radius of 0.3, the multiquadratic function with radius of 0.2, and the Gaussian function with radius of 0.5 for f_C , f_M , f_Y respectively.

4.7 Summary

The RBF based L_1 approximation technique was applied to the color conversion in the neutral color regions to minimize the effectiveness of outliers. The modified multidimensional L_1 algorithm was developed based on the BR primal algorithm. Our experimental results indicated that this algorithm was robust and efficient.

Chapter 5

ONE DIMENSIONAL COLOR MANIFOLDS IN THE CMYK COLOR SPACE

The most difficult and challenging task in color printing is to reduce the cost of printing while maintaining quality. Traditionally, the printer color conversion from CMY (or CIELab) to CMYK has been achieved by under color removal (UCR) and under color addition (UCA). UCR is the process of removing an equal amount of cyan, magenta, and yellow, and adding the same amount of black. More explicitly, define the color components as:

$$CMY = \{C, M, Y\},$$

$$CMYK = \{C, M, Y, K\},$$

$$Lab = \{L, a, b\},$$

We take the smallest value of the C, M, Y components, i.e., let

$$K = \min\{C, M, Y\}$$

and then use K to convert a CMY value to a CMYK value using the following equations:

$$C' = C - K,$$

$$M' = M - K,$$

$$Y' = Y - K,$$

Because of the nonlinearity of toners/inks, this UCR model is not close to being accurate, and introduces unacceptable color conversion errors. Colors resulting from the UCR process are dull, muddy, and hue shifted.

To mitigate this problem, the UCR process is followed by the UCA correction which consists of adding a small amount of cyan, magenta, and yellow back to the C' , M' , Y' values respectively to make

colors richer and less hue shifted. However, this method is empirical and labor intensive. The conversion error is still quite large.

A hypothetical solution to color toner/ink reduction problem consists of approximating a nonlinear continuous and differentiable functions mapping from CIELab to CMY, and CMYK to CIELab. As demonstrated in this chapter, the development of a color conversion model using RBFs makes accurate CMY to CMYK conversions possible.

In this chapter, we discuss an optimized GCR algorithm, a method of obtaining one-dimensional manifolds in the CMYK color space and an optimal toner/ink selection algorithm which results no quality degradation.

5.1 Computing an Optimal CMYK Color

The algorithm for optimized CMY to CMYK conversion is described as following: Assume we have accurate continuous functions describing CMYK to CIELab conversion and CIELab to CMY conversion:

$$g(L, a, b) \rightarrow (C, M, Y) \in (\mathbb{R}^+, \mathbb{R}^+, \mathbb{R}^+)$$

$$f(C, M, Y, K) \rightarrow (L, a, b)$$

For any CIELab value within the printer gamut, $Lab_0 = (L_0, a_0, b_0)$,

$$g(L_0, a_0, b_0) = (C_0, M_0, Y_0)$$

If $C_0 > 0$ & $M_0 > 0$ & $Y_0 > 0$, seek a value (C'_0, M'_0, Y'_0, K'_0) , s.t.,

$$f(\underbrace{C'_0, M'_0, Y'_0, K'_0}_X) \approx \underbrace{(L_0, a_0, b_0)}_Y$$

We propose to approximate the CMYK value $x_n = (C'_n, M'_n, Y'_n, K'_n)$ iteratively such that for n large enough we have

$$\|f(x_n) - y_n\| < \varepsilon$$

for some prescribed tolerance ε

Newton's method is preferred for its simplicity, efficiency, and accuracy if an initial data point close to the minimizer is found. We discuss the algorithm for finding initial starting points for the Newton's method and multidimensional Newton's method in the following subsections.

5.2 Initializing the Algorithm

The goal of finding an initial point for Newton's method is to seek a value x_0 , s.t.,

$$err = \| f(x_0) - y_0 \| < tol$$

where tol is an initial acceptable tolerance.

One of the problems in current GCR algorithm is that the CIELab value of an equal amount of CMY value is not equal to the CIELab value of that same amount of black toner/ink, K, i.e., for most cases, the CIELab value of a K value corresponds to unequal amount of cyan, magenta, and yellow. If the equal amount of C, M, Y, K values are used as the initial point to Newton's algorithm, the above inequality may not be satisfied. A better initial point is obtained by removing unequal amounts of cyan, magenta, and yellow from the CMY value respectively, and adding the K value. Our experiments indicated that for one data set, the maximum error (err) using this process was improved from the traditional UCR err value of 14.42 to 10.50. The algorithm is described as follows:

- For any K, compute the associated CIELab value using

$$f(0, 0, 0, K) \rightarrow (L^K, a^K, b^K) \quad (5.1)$$

Now find the CMY value corresponding to this K using

$$g(L^K, a^K, b^K) \rightarrow (C^K, M^K, Y^K) \quad (5.2)$$

Now we have established an equivalence

$$K \Leftrightarrow (C^K, M^K, Y^K)$$

Repeat it for a sequence of K values span the range of (0, 1).

- For any given (C, M, Y) value, there exists a maximum K_p value whose corresponding $(C^{K_p}, M^{K_p}, Y^{K_p})$ satisfies

$$C = C^{K_p} \text{ or } M = M^{K_p} \text{ or } Y = Y^{K_p}$$

Seek a sequence of K values, i.e.,

$$K = \{K_i \leq K_p, i = 1, 2, \dots, p\}$$

The corresponding $(C^{K_i}, M^{K_i}, Y^{K_i})$'s can also be removed from the (C, M, Y) value.

- Convert the CMY value (C, M, Y) to a sequence of CMYK values, i.e.,

$$CMYK' = \{(C^{K_i}, M^{K_i}, Y^{K_i}), i = 1, 2, \dots, p\} \quad (5.3)$$

where

$$C^{K_i} = C - C_{K_i}$$

$$M^{K_i} = M - M_{K_i}$$

$$Y^{K_i} = Y - Y_{K_i}$$

5.3 Optimized Gray Component Replacement Algorithm

Given some initial CMY, the optimized GCR algorithm is to determine the associated (C', M', Y', K') , such that the quality

$$\| f(C', M', Y', K') - f(C, M, Y, 0) \|$$

is as small as possible.

Since we believe that are able to find the initial point close enough to the minimizer, the multidimensional Newton's method was used to find the optimal value of (C', M', Y', K') [106].

Let

$$f = \begin{pmatrix} f_L \\ f_a \\ f_b \end{pmatrix}, x = \begin{pmatrix} C \\ M \\ Y \\ K \end{pmatrix}, y = \begin{pmatrix} L \\ a \\ b \end{pmatrix}, y_0 = \begin{pmatrix} L_0 \\ a_0 \\ b_0 \end{pmatrix}$$

where y_0 is the corresponding CIELab value of the original CMY color, i.e., the desired CIELab value for the optimizer of the (C', M', Y', K') .

Let $f(x)$ be the representation of the mapping from CMYK to CIELab and express it as a RBF, i.e.,

$$f_j(x) = A_j x + a_{j0} + \sum_{i=1}^N w_{ji} \phi_j(\|x - c_i\|) \quad (5.4)$$

where $j = 1, 2, 3$ correspond to $\{f_L, f_a, f_b\}$.

Let

$$f(x+h) \approx f(x) + f'(x)h$$

where $f'(x)$ is in the form of Jacobian matrix

$$J = f'(x) = \begin{pmatrix} \partial f_L / \partial C & \partial f_L / \partial M & \partial f_L / \partial Y & \partial f_L / \partial K \\ \partial f_a / \partial C & \partial f_a / \partial M & \partial f_a / \partial Y & \partial f_a / \partial K \\ \partial f_b / \partial C & \partial f_b / \partial M & \partial f_b / \partial Y & \partial f_b / \partial K \end{pmatrix} \quad (5.5)$$

for each iteration k ,

$$x_{k+1} = x_k - J(x_k)^{-1}(f(y_k) - f(y_0)) \quad (5.6)$$

The iteration stops when $\|(f(x_n) - y_0)\| \leq 10^{-12}$.

The dimension of the Jacobian matrix is 3×4 . In order to change it to an invertible Jacobian matrix, care needs to be taken in the implementation. The algorithm is described as follows:

- To start, keep the black value of the (C, M, Y, K) color unchanged. The Jacobian matrix in Equation 5.2 is then changed to:

$$J = f'(x) = \begin{pmatrix} \partial f_L / \partial C & \partial f_L / \partial M & \partial f_L / \partial Y \\ \partial f_a / \partial C & \partial f_a / \partial M & \partial f_a / \partial Y \\ \partial f_b / \partial C & \partial f_b / \partial M & \partial f_b / \partial Y \end{pmatrix}$$

- Solve Equation 5.2 using SVD to obtain x_{k+1} .

$$x_{k+1} = \begin{pmatrix} C_{k+1} \\ M_{k+1} \\ Y_{k+1} \\ K \end{pmatrix}$$

- At each iteration, check if the C,M,Y values in the converted (C, M, Y, K) are in the range of $[0, 1]$, i.e., if $x_{k+1}^i \geq 1$ or $x_{k+1}^i \leq 0$, where $i = 1, 2, 3$ correspond to C, M and Y component respectively. If the value of the i th component is out of the range, let

$$x_{k+1}^i = \begin{cases} 1, & x(i) \geq 1 \\ 0, & x(i) \leq 0 \end{cases}$$

Keep the i th component unchanged for the rest of the iterations, remove the column that corresponds to this component from the Jacobian matrix J , and relax the constraint on the black component by adding the partial derivatives with respect to black in the matrix J . For example, if $x(2) \geq 1$, the magenta component is removed. The Jacobian matrix J_{k+1} becomes to

$$J_{k+1} = \begin{pmatrix} \partial f_L / \partial C & \partial f_L / \partial Y & \partial f_L / \partial K \\ \partial f_a / \partial C & \partial f_a / \partial Y & \partial f_a / \partial K \\ \partial f_b / \partial C & \partial f_b / \partial Y & \partial f_b / \partial K \end{pmatrix}$$

and

$$x_{k+1} = \begin{pmatrix} C_{k+1} \\ 1.0 \\ Y_{k+1} \\ K_{k+1} \end{pmatrix}$$

Our results indicate any given CMY value ($C > 0$ & $M > 0$ & $Y > 0$) will converge to a CMYK value with the CIELab value difference practically zero.

5.4 Finding a Level Set for a Given Color

As described in the previous sections, each black level K , i.e., $CMYK = (0, 0, 0, K)$, can be mapped to a CMY value of (C_k, M_k, Y_k) . For any CMY value ($C > 0$ & $M > 0$ & $Y > 0$), there exists a set of values $\{K_i, i = 1, 2, \dots, p\}$ that their corresponding $\{C^{k_i}, M^{k_i}, Y^{k_i}, i = 1, 2, \dots, p\}$ can be removed from C,M, and Y, i.e.,

$$C \geq C^{k_i} \text{ \& } M \geq M^{k_i} \text{ \& } Y \geq Y^{k_i}, i = 1, 2, \dots, p$$

This means that for one CMY value, there exists a set of CMYK values $\{C - C^{k_i}, M - M^{k_i}, Y - Y^{k_i}, K_i, i = 1, 2, \dots, p\}$ can be used for the initial points to the multidimensional Newton's optimization algorithm.

We seek all solutions to the problem $f(C, M, Y, K) = (L_0, a_0, b_0)$. In other words, find the set of values $f^{-1}(L_0, a_0, b_0)$. We believe this set of values is a one-dimensional manifold, and we discuss it based on the pre-image theorem.

Definition 5.4.1 *Let $f : X \rightarrow Y$ be a smooth map between manifolds. We say that a point is a regular value of f if for all $x : x \in f^{-1}(y)$ the map $df : T_x X \rightarrow T_y Y$ is surjective. Here, $T_x X$ and $T_y Y$ are the tangent spaces of X and Y at the points x and y .*

Theorem 5.4.2 Pre-image Theorem *Let $f : X \rightarrow Y$ be a smooth map, and let $y \in Y$ be a regular value of f . Then $x : x \in f^{-1}(y)$ is a submanifold of X with $\dim f^{-1}(y) = \dim X - \dim Y$.*

In our application, Let

$$A = \{\text{set of elements of CMYK}\}$$

$$B = \{\text{set of elements of CIELab}\}$$

A function $f = (f_L, f_a, f_b)$ that maps A to B is in the form of RBFs described in Equation 5.4. So f is continuous and differentiable. For a CIELab value $y \in B$, if the Jacobian matrix described in Equation 5.5 at y is full rank, i.e., rank is equal to 3, then y is a regular point of f . Based on the Theorem 5.4.2, there exists a manifold $S = f^{-1}(y)$, $S \subseteq A$ with dimension of one.

We designed our algorithm for finding these one-dimensional manifolds based on the pre-image theorem. The detailed algorithm is described as follows:

- For any CIELab value y that is inside the printer gamut, obtain the CMY value using

$$g(y) \rightarrow (C, M, Y)$$

- Check if $\min(C, M, Y) = 0$. If it's true, it means y is not a regular point since its Jacobian matrix will be rank deficient. Stop the program.
- If $\min(C, M, Y) > 0$, perform the initializing algorithm to obtain the sequence of the CMYK value $CMYK'$ described in Equation 5.3.
- Check the Jacobian matrix at each starting point, and decide if y is a regular point. If y is not a regular point, stop the program.
- If y is a regular point, perform the optimized GCR algorithm to obtain optimized value for each starting point, i.e., obtain the 1-dimensional manifold $S \subseteq A$, and $S = f^{-1}(y)$.

5.5 The Optimal Toner/ink Saving Solution

When there exists a level set of CMYK values $S = f^{-1}(y)$ that correspond to the same CIELab value, $y \in B$, the optimal solution for the toner/ink selection can be obtained for a given objective function. The objective function can be toner/ink cost based in which the objective function is to minimize the cost of printing, or toner/ink coverage based, which is illustrated in the following example:

$$\min(C + M + Y + K)$$

then the minimum amount of $C + M + Y + K$ is obtained from the level set. For example, the data in Figure 5.2, for the original CMY value $\{0.7, 1, 1\}$, the total amount of toner/ink is 270%. The minimum total amount of toners/inks in the CMYK level set is 84.2% corresponding the CMYK value of $\{0.066186, 0.031126, 0, 0.74461\}$. Thus the total amount toner savings is 185.8%

Because every value in a CMYK level set corresponds to the same CIELab value, the toner savings is achieved with no quality degradation.

5.6 Experiments and Results

A set of regular points in CIELab were identified. The manifolds of these CIELab color were obtained. Examples of a CMYK level set for $CMY = \{0.55, 0.4, 0.55\}$ and $CMY = \{0.7, 1.0, 1.0\}$ are plotted in Figure 5.1 and Figure 5.2 respectively. The figures show that cyan, magenta, yellow and black are changing continuously, and each curve has a starting point and an ending point.

Each CMYK data point in the manifold was converted back to the CIELab value using $f(C, M, Y, K) \rightarrow (L, a, b)$. These CIELab values were compared to the original CIELab regular point. Our results indicated that the maximum CIELab error was 1.27, and the mean error was 0.523.

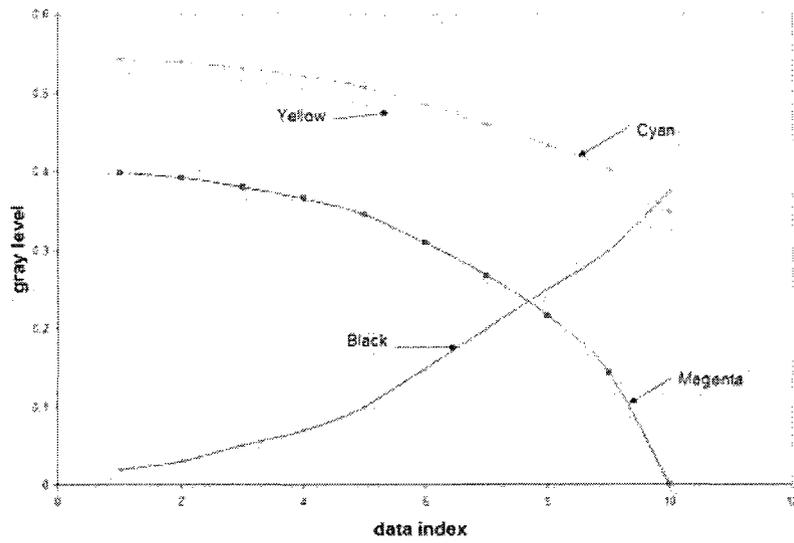


Figure 5.1: The level set of CMYK values for $CMY = \{0.55, 0.4, 0.55\}$.

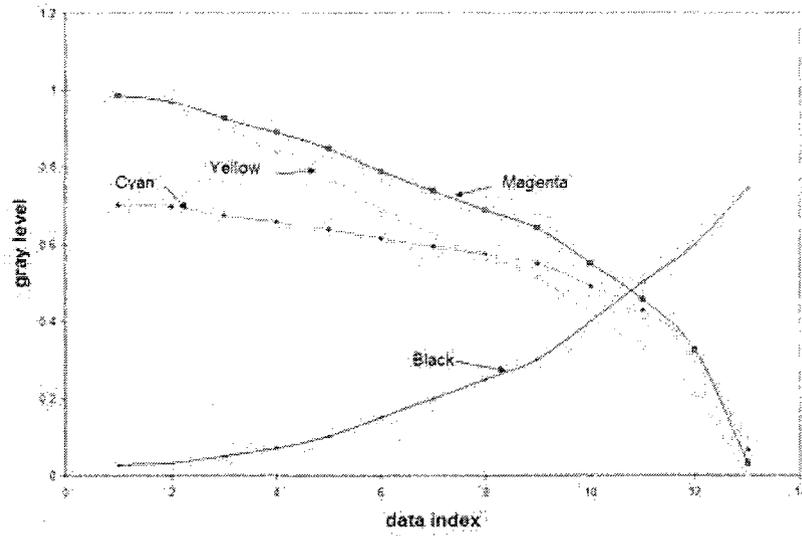


Figure 5.2: The level set of CMYK values for $CMY = \{0.7, 1.0, 1.0\}$.

5.7 Summary

For the color conversion from CMYK to CIELab using RBFs, if a point y in the CIELab color space is a regular point, the pre-image theorem implies that there exists a one-dimensional manifold $S = f^{-1}(y)$ in the CMYK color space. In this chapter, we described a detailed algorithm of finding the one-dimensional manifold in the CMYK color space for any given regular point in CIELab.

Firstly, we described our optimized GCR algorithm. Since we were able to find the starting points close enough to the minimizer, a multi-dimensional Newton's method was used as the optimization algorithm to find the CMYK color whose corresponding CIELab value is practically equal to the corresponding CIELab value of the original CMY color (the difference of these two CIELab colors is less than 10^{-12}).

Secondly, we discussed the algorithm for finding the one-dimensional manifold in the CMYK color space based on the optimized GCR algorithm. A set of regular points in the CIElab space are identified. For each regular point, a set of CMYK colors were obtained as the starting points to the optimized GCR algorithm. The optimized GCR algorithm was performed for each starting point to obtain the one-dimensional manifold in the CMYK color space.

Lastly, we discuss our optimal solutions for the toner/ink selection. Because of the availability of the one-dimensional manifold in the CMYK color space, we are able to define objective functions for toner savings, cost savings, etc., with no quality degradation.

Our results indicated that the optimized GCR algorithm and one-dimensional manifold algorithm are accurate and efficient.

Chapter 6

COLOR CONVERSION WITH INK LIMITATIONS

One of the challenging tasks in color printing relates to the limitation of the amount of toner/ink coverage on a page to avoid excessive bleed-through. This task is particularly important for inkjet printing. When there exists a limit for the maximum amount of toner/ink coverage allowed per spot in the color printing, the color conversion in highly saturated color regions, or dark neutral color regions is effected. As discussed in the previous chapters, the L_1 norm approximation is the most suitable solution for solving the neutral color conversion problem, and the L_∞ approximation is the best approach for color conversion in saturated color regions. For both approaches, we employed RBFs with linear programming.

The color conversion effected by this limit is the CIELab to CMY color conversion. The existence of a toner/ink limitation can be added as a constraint in the linear programming problem formulations. In this chapter we discuss:

- The L_1 approximation problem formulation for solving the neutral color conversion with the toner/ink limitation.
- A modified BR L_1 approximation algorithm for solving the multidimensional neutral color conversion with the toner/ink limitation.
- The L_∞ approximation problem formulation for solving the color conversion with the toner/ink limitation.
- A modified BP L_∞ approximation algorithm for solving the multidimensional color conversion with the toner/ink limitation.

6.1 The Ink Limitation Problem Using the L_1 Approximation

Let $g(z)$ be the representation of the mapping from CIELab to CMY in the form of RBF:

$$g_k(z_j) = A_k z_j + a_{k0} + \sum_{i=1}^n \alpha_{ki} \phi_k(\|z_j - c_i\|) \quad (6.1)$$

where $k = 1, 2, 3$ corresponding to $\{g_C, g_M, g_Y\}$.

$$z_j = \{L, a, b\}_j$$

where $i = 1, 2, \dots, n$ and $j = 1, 2, \dots, m$. The dimension of A_k is 1×3 corresponding to the dimension of the domain. The dimension of a_0 is 1. Therefore, there are $n + 4$ coefficients in Equation 6.2. The equation of 6.1 can be written as

$$g_k(z_j) = \underbrace{(z_j, 1, \phi_k(\|z_j - c_1\|), \phi_k(\|z_j - c_2\|), \dots, \phi_k(\|z_j - c_n\|))}_{F_k(z_j)} \underbrace{\begin{pmatrix} A_k^T \\ a_{k0} \\ \alpha_{k1} \\ \alpha_{k2} \\ \vdots \\ \alpha_{kn} \end{pmatrix}}_{w_k} \quad (6.2)$$

$$= \sum_{i=1}^{n+4} w_{ki} F_{ki}(z_j) \quad (6.3)$$

Define the L_1 error for each data point of component of C, M and Y using nonnegative variables:

$$g_k(z_j) - \sum_{i=1}^{n+4} w_{ki} F_{ki}(z_j) = u_{kj} - v_{kj}$$

and define the coefficients for each component using nonnegative variables:

$$w_{ki} = b_{ki} - c_{ki}$$

where $u_{kj}, v_{kj}, b_{ki}, c_{ki} \geq 0$

6.1.1 The Primal Problem Formulation in the L_1 Sense

The primal linear programming problem in the L_1 sense is described below. The constraint of the toner/ink limit applies to the color data whose toner/ink summation exceeds L .

$$\text{Minimize } \xi = \sum_{j=1}^m u_{1j} + v_{1j} + u_{2j} + v_{2j} + u_{3j} + v_{3j}$$

subject to

$$\sum_{i=1}^{n+4} (a_{1j} - b_{1j}) F_{1j}(z_j) + u_{1j} + v_{1j} = C_j \quad (6.4a)$$

$$\sum_{j=1}^{n+4} (a_{2j} - b_{2j}) F_{2j}(z_j) + u_{2j} + v_{2j} = M_j \quad (6.4b)$$

$$\sum_{j=1}^{n+4} (a_{3j} - b_{3j}) F_{3j}(z_j) + u_{3j} + v_{3j} = Y_j \quad (6.4c)$$

$$\sum_{j=1}^{n+4} (a_{1j} - b_{1j}) F_{1j}(z_{ji}) + \sum_{j=1}^{n+4} (a_{2j} - b_{2j}) F_{2j}(z_{ji}) + \sum_{j=1}^{n+4} (a_{3j} - b_{3j}) F_{3j}(z_{ji}) \leq L \quad (6.4d)$$

where

$$J = \{j_l \mid C_{j_l} + M_{j_l} + Y_{j_l} > L\} \subseteq \{i = 1, 2, \dots, m\}$$

$$u_{kj}, v_{kj}, b_{ki}, c_{ki} \geq 0, \quad k = 1, 2, 3, \quad i = 1, 2, \dots, n, \quad \text{and } j = 1, 2, \dots, m.$$

6.1.2 The Modified BR L_1 Approximation with the Toner/ink Limitation

Based on the formulation described above, the condensed initial simplex tableau for the CIE Lab to CMY conversion with the toner/ink limitation L is shown in Table 6.1. Let p denote the number of color data points that exceed the toner/ink limit. The slack variables s_i 's need to be added in Equation 6.4d.

Table 6.1: Condensed initial simplex tableau for the CIELab to CMY conversion with toner/ink limitation in the L_1 sense.

| Basis | R | b_{C_1} | ... | b_{C_n} | b_{M_1} | ... | b_{M_n} | b_{Y_1} | ... | b_{Y_n} |
|---------------|---------------------------------------|----------------------------|-----|-----------------------------|-----------------------------|-----|-----------------------------|-----------------------------|-----|-----------------------------|
| u_{C_1} | C_1 | $F_{11,1}$ | ... | $F_{1n,1}$ | 0 | ... | 0 | 0 | ... | 0 |
| \vdots | \vdots | \vdots | | \vdots | \vdots | | \vdots | \vdots | | \vdots |
| u_{C_m} | C_m | $F_{11,m}$ | ... | $F_{1n,m}$ | 0 | ... | 0 | 0 | ... | 0 |
| u_{M_1} | M_1 | 0 | ... | 0 | $F_{21,1}$ | ... | $F_{2n,1}$ | 0 | ... | 0 |
| \vdots | \vdots | \vdots | | \vdots | \vdots | | \vdots | \vdots | | \vdots |
| u_{M_n} | M_m | 0 | ... | 0 | $F_{21,n}$ | ... | $F_{2n,m}$ | 0 | ... | 0 |
| u_{Y_1} | Y_1 | 0 | ... | 0 | 0 | ... | 0 | $F_{31,1}$ | ... | $F_{3n,1}$ |
| \vdots | \vdots | \vdots | | \vdots | \vdots | | \vdots | \vdots | | \vdots |
| u_{Y_n} | Y_m | 0 | ... | 0 | 0 | ... | 0 | $F_{31,n}$ | ... | $F_{3n,m}$ |
| s_1 | L | F_{11,j_1} | ... | F_{1n,j_1} | F_{21,j_1} | ... | F_{2n,j_1} | F_{31,j_1} | ... | F_{3n,j_1} |
| \vdots | \vdots | \vdots | | \vdots | \vdots | | \vdots | \vdots | | \vdots |
| s_p | L | F_{11,j_p} | ... | F_{1n,j_p} | F_{21,j_p} | ... | F_{2n,j_p} | F_{31,j_p} | ... | F_{3n,j_p} |
| Marginal Cost | $\sum_{j=1}^m (C_j + M_j + Y_j) + pL$ | $\sum_{j=1}^{m+p} F_{1,j}$ | ... | $\sum_{j=1}^{m+p} F_{1n,j}$ | $\sum_{j=1}^{m+p} F_{21,j}$ | ... | $\sum_{j=1}^{m+p} F_{2n,j}$ | $\sum_{j=1}^{m+p} F_{31,j}$ | ... | $\sum_{j=1}^{m+p} F_{3n,j}$ |

The modified BR's two stages algorithm is describe as the following [89]:

- Stage 1: There are three sets of n iterations for the C, M, and Y components respectively. In each of the n iterations, the pivotal columns are restricted to b_i and c_i only. The vector entering the basis is the one with the largest nonnegative marginal cost. The vector leaving the basis is chosen from v_j and u_j for the one causing the maximum reduction in the objective function [89]. The interchange of vectors leaving and entering the basis can only be performed within the same color component.
- Stage 2: Interchange the nonbasic u_j or v_j with the basic u_j , v_j , or s_{j_i} . Neither b_i or c_i are allowed to leave the basis. The criteria for selecting vectors leaving and entering the basis are the same as that in the Stage 1. The interchange of vectors v_j and u_j leaving and entering the basis can only be performed within the same color component. However, w_{j_i} can be interchanged with the v_j and u_j vector for any color component. Once the vector s_{j_i} leaves the basis, it's not allowed to enter back onto the basis. The iteration stops when all the marginal costs are non-positive.
- Interchange basic vectors b_i or c_i with the corresponding nonbasic vectors c_j or b_j if the final tableau at the end of Stage 2 is infeasible.

6.2 Solving the Ink Limitation Problem Using the L_∞ Approximation

As discussed in Chapter 3, the BP dual algorithm is the preferred algorithm for the color conversion using L_∞ approximation. For the color conversion with the toner/ink limitation, a constraint needs to be added in the problem formulation.

6.2.1 The Problem Formulation in the L_∞ Sense

Based on Equation 6.3, the L_∞ errors are defined for component of C, M and Y:

$$\xi_c = \max_{1 \leq j \leq m} \left\| \sum_{i=1}^{n+4} w_{c_i} F_{c_i}(z_j) - C_j \right\|_\infty$$

$$\xi_m = \max_{1 \leq j \leq m} \left\| \sum_{i=1}^{n+4} w_{m_i} F_{m_i}(z_j) - M_j \right\|_\infty$$

$$\xi_y = \max_{1 \leq j \leq m} \left\| \sum_{i=1}^{n+4} w_{y_i} F_{y_i}(z_j) - Y_j \right\|_\infty$$

First we need to formulate the problem in the primal form.

$$\text{Minimize } \xi = \xi_c + \xi_m + \xi_y,$$

subject to

$$\xi_c + \sum_{i=1}^{n+4} \alpha_{c_i} F_{c_i}(z_j) \geq C_j \quad (6.5a)$$

$$\xi_c - \sum_{i=1}^{n+4} \alpha_{c_i} F_{c_i}(z_j) \geq -C_j \quad (6.5b)$$

$$\xi_m + \sum_{i=1}^{n+4} \alpha_{m_i} F_{m_i}(z_j) \geq M_j \quad (6.5c)$$

$$\xi_m - \sum_{i=1}^{n+4} \alpha_{m_i} F_{m_i}(z_j) \geq -M_j \quad (6.5d)$$

$$\xi_y + \sum_{i=1}^{n+4} \alpha_{y_i} F_{y_i}(z_j) \geq Y_j \quad (6.5e)$$

$$\xi_y - \sum_{i=1}^{n+4} \alpha_{y_i} F_{y_i}(z_j) \geq -Y_j \quad (6.5f)$$

$$-\sum_{i=1}^{n+4} (\alpha_{c_i} F_{c_i}(z_{j_i}) + \alpha_{m_i} F_{m_i}(z_{j_i}) + \alpha_{y_i} F_{y_i}(z_{j_i})) \geq -L \quad (6.5g)$$

where

$$\xi_c, \xi_m, \xi_y \geq 0$$

$$\alpha_1, \alpha_2, \dots, \alpha_{n+4} \text{ unrestricted}$$

$$z_j = \{L, a, b\}_j, j = 1, 2, \dots, m$$

$$J = \{j_l \mid C_{j_l} + M_{j_l} + Y_{j_l} > L\} \subseteq \{j = 1, 2, \dots, m\}$$

$$F_j = [1, z_j, \phi_1(z_j), \phi_2(z_j), \dots, \phi_n(z_j)]_{n+4}$$

Transformed from the primal formulation, the dual formulation is describe as the following:

$$\text{Maximize } \sum_{i=1}^m [(\underline{\sigma}_{c_i} - \underline{\tau}_{c_i})C_i + (\underline{\sigma}_{m_i} - \underline{\tau}_{m_i})M_i + (\underline{\sigma}_{y_i} - \underline{\tau}_{y_i})Y_i] - L \sum_{i_k=1}^p \rho_{i_k}$$

subject to

$$\sum_{j=1}^m (\underline{\sigma}_{c_j} - \underline{\tau}_{c_j}) F_{c_{ij}}^T - \sum_{j_i=1}^p \rho_{j_i} F_{c_{ij_i}}^T = 0 \quad (6.6a)$$

$$\sum_{j=1}^m (\underline{\sigma}_{c_j} + \underline{\tau}_{c_j}) \leq 1 \quad (6.6b)$$

$$\sum_{j=1}^m (\underline{\sigma}_{m_j} - \underline{\tau}_{m_j}) F_{m_{ij}}^T - \sum_{j_i=1}^p \rho_{j_i} F_{m_{ij_i}}^T = 0 \quad (6.6c)$$

$$\sum_{j=1}^m (\underline{\sigma}_{m_j} + \underline{\tau}_{m_j}) \leq 1 \quad (6.6d)$$

$$\sum_{j=1}^m (\underline{\sigma}_{y_j} - \underline{\tau}_{y_j}) F_{y_{ij}}^T - \sum_{j_i=1}^p \underline{\rho}_j F_{y_{ij_i}}^T = 0 \quad (6.6e)$$

$$\sum_{j=1}^m (\underline{\sigma}_{y_j} + \underline{\tau}_{y_j}) \leq 1 \quad (6.6f)$$

and

$$\underline{\sigma}_j, \quad \underline{\tau}_j, \quad \underline{\rho}_j \geq 0, \quad j = 1, 2, \dots, m$$

where

$$z_j = \{L, a, b\}_j, \quad j = 1, 2, \dots, m$$

$$J = \{j_l \mid C_{j_l} + M_{j_l} + Y_{j_l} > L\} \subseteq \{j = 1, 2, \dots, m\}$$

$$F_j = [1, z_j, \phi_1(z_j), \phi_2(z_j), \dots, \phi_n(z_j)]_{n+4}$$

$$F_{j_l} = [1, z_{j_l}, \phi_{j_1}(z_{j_l}), \phi_{j_2}(z_{j_l}), \dots, \phi_n(z_{j_l})]_{n+4}$$

$$\phi_{ij} = \phi_i(z_j).$$

p denotes the number of color data points that exceed the toner/ink limit.

6.2.2 The Modified BP L_∞ Approximation Algorithm for the Toner/ink Limitation Problem

A modified BP algorithm was used for solving the color conversion from CIELab to CMY with a toner/ink limitation L . The condensed initial simplex tableau is shown in the next table. The algorithm is described below:

- In the first stage of the BP algorithm, the first n simplex iterations were performed [75].
 - Only α_c 's can be moved out of the basis. The vector entering the basis has to be selected from \underline{s}_c which corresponds to that with the largest absolute reduced cost.
 - Only α_m 's can be moved out of the basis. The vector entering the basis has to be selected from \underline{s}_m which corresponds to that with the largest absolute reduced cost.
 - Only α_y 's can be moved out of the basis. The vector entering the basis has to be selected from \underline{s}_y which corresponds to that with the largest absolute reduced cost.
- In the second stage of the BP algorithm [75]:
 - ξ_c is forced to be moved out of the basis. The vector entering the basis has to be selected from the remaining nonbasic vector \underline{s}_c 's. No α_c is allowed to move back onto the basis.

- ξ_m is forced to be moved out of the basis. The vector entering the basis has to be selected from the remaining nonbasic vector s_m . No α_m is allowed to move back onto the basis.
- ξ_y is forced to be moved out of the basis. The vector entering the basis has to be selected from the remaining nonbasic vector s_y . No α_y is allowed to move back onto the basis.
- In the third stage of the BP algorithm [75]:
 - The pivotal column is chosen corresponding to the most negative marginal cost.
 - The pivotal row is chosen by ratio selection rule.
 - The interchange of nonbasic vectors s_i 's and basic vectors s_j 's can only be performed within the same color component. The iteration continues until all marginal costs corresponding to the nonbasic vectors s_c , s_m , and s_y , are nonnegative. So far, no vector s_{cmy} is allowed to enter the basis.
- The fourth stage is added to perform simplex iteration interchanging nonbasic vectors including all the remaining vectors s_c , s_m , and s_y , and s_{cmy} and basic vectors in the basis. Nonbasic vectors s_c , s_m , and s_y can only interchange with basic vectors of s_c , s_m , and s_y respectively. The nonbasic vectors s_{cmy} can interchange with any basic vectors of s_c , s_m , and s_y . The algorithm stops when all the marginal costs are nonnegative.

The function ϕ is selected satisfying the Haar condition as described in Chapter 3.

Table 6.2 is the condensed initial simplex tableau for the CIELab to CMY conversion with the toner/ink limitation L .

Table 6.2: Condensed initial simplex tableau for the CIELab to CMY conversion with the Toner/ink Limitation.

| Basis | sC_1 | sC_m | sM_1 | sY_1 | sY_m | $sCMY_{j_1}$ | $sCMY_{j_p}$ |
|---------------|---------------|---------------|---------------|---------------|---------------|-----------------|-----------------|
| αC_1 | $F_{C_1,1}^T$ | $F_{C_1,m}^T$ | 0 | 0 | 0 | F_{C_1,j_1}^T | F_{C_1,j_p}^T |
| \vdots | \vdots |
| αC_n | $F_{C_n,1}^T$ | $F_{C_n,m}^T$ | 0 | 0 | 0 | F_{C_n,j_1}^T | F_{C_n,j_p}^T |
| αM_1 | 0 | $F_{M_1,1}^T$ | $F_{M_1,m}^T$ | 0 | 0 | F_{M_1,j_1}^T | F_{M_1,j_p}^T |
| \vdots | \vdots |
| αm_n | 0 | 0 | $F_{M_n,1}^T$ | 0 | 0 | F_{M_n,j_1}^T | F_{M_n,j_p}^T |
| αY_1 | 0 | 0 | 0 | $F_{Y_1,1}^T$ | $F_{Y_1,m}^T$ | F_{Y_1,j_1}^T | F_{Y_1,j_p}^T |
| \vdots | \vdots |
| αY_n | 0 | 0 | 0 | $F_{Y_n,1}^T$ | $F_{Y_n,m}^T$ | F_{Y_n,j_1}^T | F_{Y_n,j_p}^T |
| w_C | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| w_M | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| w_Y | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| Marginal Cost | $-C_1$ | $-C_m$ | $-M_1$ | $-Y_1$ | $-Y_m$ | L | L |

6.3 Experiment and Results

Because the constraint was placed on the the total amount of cyan, magenta and yellow, the CIELab to CMY conversion was performed. Like the experiments described in Chapter 3 and Chapter 4, all 866 color data with 64 cluster centers were used. The toner/ink limits were set to 2.8, 2.5, 2.0, 1.8 and 1.60 for both L_1 and L_∞ approximations. Various RBFs were used. Our results indicated that both L_1 and L_∞ algorithms were able to convert all colors from CIELab to CMY with the toner coverage less or equal to each limit regardless what RBFs were used. For both L_1 and L_∞ approximations with an added constraint, the color conversion accuracy for the colors within each toner/ink limit is very similar to the results obtained in L_1 and L_∞ approximations with no constraint respectively.

6.4 Summary

The color conversion with the toner/ink limitation problem was solved via both L_1 and L_∞ approximation algorithms. The L_1 and L_∞ approximation algorithms were developed for solving this problem in the neutral and saturated color regions respectively. A constraint with the toner/ink limitation was added in the problem formulation. The L_1 algorithm was a modified BR primal algorithm with this added constraint, while the L_∞ algorithm was developed based on the BP dual algorithm which extended the three-stage algorithm to a four-stage algorithm. Our results indicated that both algorithms are efficient and robust. They are able to convert all colors from CIELab to CMY with any given toner/ink limitation regardless what RBFs are used.

Chapter 7

GAMUT MAPPING

Due to device color gamut mismatches, colors outside an output device gamut need to be mapped onto colors inside the device gamut. The results of the color gamut mapping can be very unpleasing. Firstly, the mapped colors have visible color differences from the original colors. Secondly, many perceptually different colors are mapped to the same color. In the color industry, the current gamut mapping methods are more art than science. There is much research on the optimized gamut mapping directions[24-35]. All were based on psychophysics experiments, such as CUSP, constant lightness, GCUSP, etc[35]. However, there is no universal accepted direction that works well for all color regions and devices.

We believe that the gamut mapping process should be a part of the color conversion process since the color conversion should be continuous and smooth from one neighborhood to its adjacent neighborhoods. Thus, the numerical method ought to play an important role in gamut mapping process. As gamut mapping is subject to the criticism of the human perception, we believe the best computational gamut mapping strategy is to perform numerical gamut mapping guided by a perceptual color difference model.

7.1 Gamut Mapping Algorithm Based on a Numerical and Perceptual Model

The optimized RBF models for the inside gamut color conversions from CIELab to CMY, g_0 , and CMY to CIELab, f_0 are obtained as described in the previous chapters. The algorithm for out-of-gamut conversion from CIELab to CMYK is describe in the following steps:

1. Define the thresholds for color difference in hue angle T_h and color difference in lightness T_L based on the perceptual color difference tolerance.
2. Define the output device gamut boundary gmt_0 specified in the CIELab color space. This boundary was expanded into a few layers with a small increment each time. $\{gmt_i, i = 1, 2, \dots, N\}$. In each layer, the CIELab values are $\{(L, a, b)_{i_j}, j = 1, 2, \dots, m_i\}$. The corresponding color in CIELch are $\{(L, c, h)_{i_j}, j = 1, 2, \dots, m_i\}$, where m_i is the number of the CIELab colors in the layer i .

3. To start, obtain the CMY value using

$$g_0((L, a, b)_{1j}) \rightarrow (C, M, Y)_{1j}$$

where g_0 is the mapping function from CIELab to CMY based on the inside gamut colors. $(L, a, b)_1 = \{(L, a, b)_{1j}, j = 1, 2, \dots, m_1\}$, i.e., the CIELab data in the first expanded layer. $(C, M, Y)_1 = \{(C, M, Y)_{1j}, j = 1, 2, \dots, m_1\}$, m_1 is the number of color data points in gmt_1 .

4. If any C, M, and Y component value in $(C, M, Y)_1$ is smaller than 0 or greater than 1, then the value is changed to 0 or 1 respectively. The $(C, M, Y)_1$ is then changed to $(C, M, Y)'_1$.
5. Obtain $(L, a, b)'_1$ using

$$f_0(C'_{1j}, M'_{1j}, Y'_{1j}, 0) \rightarrow (L'_{1j}, a'_{1j}, b'_{1j})$$

where f_0 is the mapping function from CMYK to CIELab based on the inside gamut colors .

6. Convert $(L, a, b)'_1$ to $(L, C, h)'_1$. Calculate the lightness difference ($\{dL_{1j}, j = 1, 2, \dots, m_1\}$) and the hue difference ($\{dh_{1j}, j = 1, 2, \dots, m_1\}$) for each data point between $(L, C, h)_1$ and $(L, C, h)'_1$.
7. If $dh_{1j} > T_h$ or $dL_{1j} > T_L$, convert $(L, C, h)'_1$ to $(L, C, h)''_1$ by making the following changes in L'_{1j} and h'_{1j} :

$$L = \begin{cases} L_{1j} + T_L, & L'_{1j} > L_{1j} \\ L_{1j} - T_L, & L'_{1j} < L_{1j} \end{cases} \quad (7.1)$$

$$h = \begin{cases} h_{1j} + T_h, & h'_{1j} > h_{1j} \\ h_{1j} - T_h, & h'_{1j} < h_{1j} \end{cases} \quad (7.2)$$

8. Repeat the Steps 2-7 until all the $(C, M, Y)_1$ values are found to meet the criteria of T_h and T_L . Now we mapped all out-of-gamut CIELab values $(L, a, b)_1$ in gmt_1 with $(C, M, Y)_1$.
9. Generate models of g_1 and f_1 using the data points in gmt_1 :

$$g_1((L, a, b)_1) \rightarrow (C, M, Y)$$

and

$$f_1((C, M, Y)_1) \rightarrow (L, a, b)$$

10. Repeat the Steps 2-9, interpolate the CIELab data in $\{gmt_i, i = 2, 3, \dots, N\}$, and find $\{(C, M, Y)_{ij}, j = 1, 2, \dots, m_i\}$.

11. The iteration stops when the last color layer reaches the optimized color ranges defined in [44].

When the RBF models f_i and g_i are generated based on the data points in gamut layer gmt_i , the data used in generating these models contain two parts:

- The data that fit the model g_{i-1} and f_{i-1} that are generated based on the gamut layer $i - 1$.
- The data that does not fit g_{i-1} and f_{i-1} . These data who have the C, M, or Y values either below 0 or above 1 using f_{i-1} are clipped to either 0 and 1, and refined with thresholds T_h and T_L .

In doing so, the models generated for each gamut layer are continuous and smooth. To summarize, the gamut mapping direction was obtained by creating the numerical model guided by a perceptual color difference model. The advantages of this technique are that more levels are obtained from this gamut mapping algorithm; out-of-gamut colors in a close neighborhood vary continuously. The direction of the gamut mapping is not fixed as other methods, but varies smoothly.

7.2 Experiments and Results of Gamut Mapping Algorithm Comparison

The current gamut mapping algorithm was implemented, and compared with one of our previous gamut mapping algorithms. It was found that the current gamut mapping algorithm transformed a connected out-of-gamut color data set to a connected set of color data around the boundary of the device gamut. The results are shown in Figure 7.1 using color data at a hue angle of 300° as the example. The “o”s represent colors outside a printer gamut. There are three sets of these colors: $s_1 = \{L^* = 40, C_{ab}^* = 20, 30, \dots 80\}$; $s_2 = \{L^* = 50, C_{ab}^* = 40, 50, \dots 80\}$; and $s_3 = \{L^* = 60, C_{ab}^* = 30, 40, \dots 80\}$. Our previous gamut mapping algorithm mapped each set of colors s_1 , s_2 , and s_3 onto three single points on the boundary at lightness level of 40, 50 and 60 respectively. In other words, the color differences within each set are lost. Our current gamut mapping algorithm mapped colors in s_1 , s_2 , and s_3 onto three sets of colors near the boundary. The “*”s represent colors after gamut mapped. It is very clear that these gamut mapped colors are differentiable.

It was also found that the gamut mapped colors change smoothly and maintain similar properties relative to the colors in the same neighborhood. These results are shown in Figure 7.2. There are three color ramps in this figure. The middle ramp are the original colors changing smoothly from blue to purple. The first 6 colors from left to right are the colors out-of a printer gamut, and the last two colors in the ramp are the colors inside the printer gamut. The top ramp is converted to a printer color using the minimum distance gamut mapping algorithm. The bottom ramp is converted to the colors of the same printer using the current gamut mapping algorithm. The results showed that the

minimum-distance gamut mapping algorithm converts the first three colors from left to right to the same color, and the next 4 – 6th colors to another color. There is an abrupt change from the 6th color to the 7th color which is inside printer gamut. The colors converted using the current gamut in the last ramp change more smoothly, and maintain the similar hue changes as in the original colors. The change from the 6th color to the 7th color is much smoother.

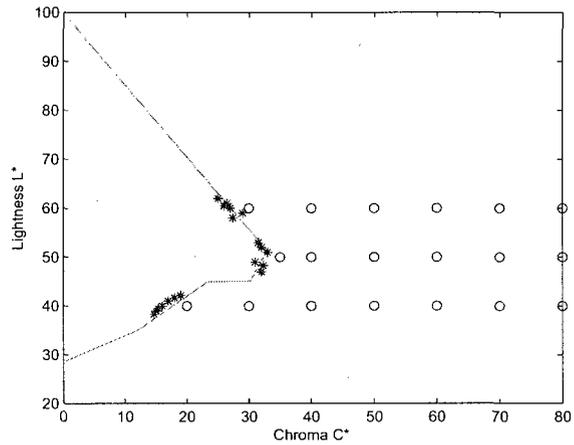


Figure 7.1: Illustration of the current gamut mapping algorithm for the color data at hue angle 300° .

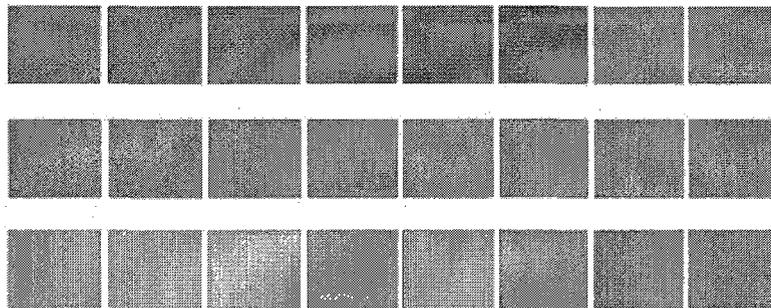


Figure 7.2: Gamut mapping comparison for blue-purple colors.

7.3 Summary

A novel gamut mapping algorithm was developed based on the numerical model guided by a perceptual color difference model. The gamut boundary was expanded into a few layers with a small increment each time. The out-of-gamut CIELab colors in each layer are converted to the CMY values using the RBFs models generated based on the data in the previous layer. These CMY values are corrected by a perceptual color difference model. Our results showed that the current gamut mapping algorithm transformed a connected out-of-gamut color set to a connected set of colors around the boundary of the device gamut. Therefore, more levels are obtained. The out-of-gamut colors in a close neighborhood vary continuously. The direction of the gamut mapping is not fixed as other methods, but varies smoothly.

Appendix I: Color Science Terminology

- **Brightness:** Attribute of a visual sensation according to which an area appears to exhibit more or less light [2].
- **Chroma:** The colorfulness of an area judged in proportion to the brightness of a similarly illuminated area that appears to be white or highly transmitting [2].
- **CMYK color space:** The primary colors cyan, magenta, yellow, and black, used together in printing to effectively create a multitude of other colors. Based on the subtractive color theory, the primary colors are used in four color printing processes.
- **Color:** A visual attribute of things that results from the light they emit, transmit, or reflect.
- **Color component:** A dimension of a color value expressed as a numeric value. A color value may consist of one, two, three, four, or eight components, also referred to as channels. According to the International Color Consortium (ICC) Specifications, there can be up to 15 components for a color space [3].
- **Color conversion:** The process of converting colors from one color space to another.
- **Color gamut:** A range of colors achievable on a given color reproduction medium (or present in an image on that medium) under a given set of viewing conditions. It is a volume in color space.
- **Color gamut boundary:** a surface determined by a color gamut's extremes.
- **Color space:** A model for representing color with intensity values. It specifies how color information is represented.
- **Colorfulness:** Attribute of a visual sensation according to which an area appears to exhibit more or less of its hue [2].
- **Device-dependent color space:** The color spaces tied to a specific piece of equipment.

- **Device-independent color space:** A set of mathematical models that are used to represent colors matching human perceptions. CIE-based color spaces allow color to be expressed in a device-independent way. It ensures colors to be predictably and accurately matched among various color devices.
- **Gamma:** The light emitted from a CRT is not linearly proportionate to the electric signal levels driving it, thus a distortion is created in the image shown on the screen, where contrast loses its linearity. This non-linearity is expressed by an exponential function, and the power to this exponential function is defined as gamma.
- **Hue:** Attribute of a visual sensation according to which an area appears to similar to one, or to proportions of two, of the perceived colors red, yellow, green, and blue [2].
- **Lightness:** The brightness of an area judged relative to the brightness of a similarly illuminated area that appears to be white or highly transmitting [2].
- **Luminance:** In a given direction, at a point in the path of a beam, the luminous intensity per unit projected area [2].
- **Neutral colors:** Colors that have low value of chroma, such as black, white, beige, silver, brown, etc.
- **Perceptual color space:** CIE based color space, such as CIELab and CIEXYZ.
- **RGB color space:** Additive colors are transmitted light used in video monitors and televisions. Red, green and blue light are referred as the additive primary colors. When used in various degrees of intensity and variation, they create all other colors of light; when superimposed equally, they create gray.
- **Saturation:** The colorfulness of an area judged in proportion to its brightness [2].
- **sRGB:** A standard RGB system created by HP and Microsoft. It is well defined and is designed to match typical home and office viewing conditions.
- **Subtractive system:** It's also referred as the CMYK system. The subtractive color process is based on light reflected from an object and passed through pigments or dyes that absorb certain wavelengths, allowing others to be reflected. In theory, the combination of these three colors should produce black, but the fourth color black in CMYK is needed to produce true black.
- **White point:** Color stimulus to which color space values are normalized. [ISO 12231]

Appendix II: Matlab Code

Center Selection and Clustering

```
1 function ccl=finalCenters_e(data,inW,sr)
2
3 cc0=initialCenters(data,inW);
4 [ccl,indx]=clusterCenter_e(data,cc0,sr);
5
6 go=1; while go>0
7     indx=refineClusters_e(data,ccl,indx,inW,sr);
8     cc=findCenters_e(data,indx);
9     go=length(ccl)-length(cc);
10    [ccl,indx]=clusterCenter_e(data,cc,sr);
11 end
```

```
1 function [cc]=initialCenters(data,inW)
2
3 len=length(data);
4 num=floor(len/inW);
5
6 rn=randomNumberGenerator(len,num);
7
8 for j=1:num
9     cc(j,:)=data(rn(j),:);
10 end
```

```
1 indx=refineClusters_e(data,cc,indx,inW,sr)
2
3 [m,n]=size(indx);
4 len=length(cc);
5 cc(:,8)=1:len;
6
7 cnt1=0;
8 for i=1:len
9     cnt=0;
10    for j=1:m
11        if indx(j,2)==i
12            cnt=cnt+1;
13            d(cnt)=indx(j,1);
14            tmp(cnt,:)=data(indx(j,1),:);
15        elseif indx(j,2)>i
16            break;
17        end
18    end
19
20    M=cov(tmp(:,5:7));
21    rk=rank(M);
22    if rk>=3
23        clear tmp d;
24        continue;
```

```

25     elseif rk<3
26         p=i-cnt1;
27         cc(p,:)=[];
28         tindx=sortrows(indx,1);
29         for k=1:cnt
30             ind=findCluster_e(tmp(k,:),cc,sr);
31             tindx(d(k),2)=ind;
32         end
33         indx=sortrows(tindx,2);
34         cnt1=cnt1+1;
35         clear d tindx tmp;
36     end
37 end

```

```

1 function cc=findCenters_e(data, indx);
2
3 n=length(indx);
4 cnt=1;
5 tmp(1,:)=data(indx(1,1),:);
6 p=0;
7
8 for i=2:n
9     if (indx(i-1,2)==indx(i,2))
10        cnt=cnt+1;
11        tmp(cnt,:)=data(indx(i,1),:);
12    else
13        p=p+1;
14        cc(p,:)=mean(tmp,1);
15        clear tmp;
16        cnt=1;
17        tmp(cnt,:)=data(indx(i,1),:);
18    end
19 end
20 end
21 cc(p+1,:)=mean(tmp,1);
22 clear tmp;

```

```

1
2 function [cc,indx]=clusterCenter_e(data,cc,sr)
3
4 go=1;
5 while (go>0.001)
6     cc=sortrows(cc,[5,6,7]);
7     [len,col]=size(cc);
8     cc(:,8)=1:len;
9     indx=findClusters_e(data,cc,sr);
10    ncc=findCenters_e(data,indx);
11    if (length(ncc)==len)
12        go=norm(ncc-cc(:,1:7))
13    end
14    cc=ncc;
15 end

```

RBF functions

```

1 function v=rbfFunction(x,r,name)
2 %x input value
3 %r ratio
4
5 %name: name of the function
6
7     if name=='g'

```

```

8         v=exp(-x*x'/r^2); %gaussian
9     elseif name=='c'
10        v=norm(x)^3; %cubic
11     elseif name=='l'
12        v=norm(x); %linear
13     elseif name=='t'
14        tmp=norm(x);
15        if tmp==0
16            tmp=0.000001;
17        end
18        v=tmp^2*log(tmp); %thin plate spline
19     elseif name=='m'
20        v=sqrt(x*x'+r*r); %multiquadric
21     elseif name=='inv'
22        if x==0
23            x=0.000001;
24        end
25        v=(r^2+x*x')^(-1/2);
26     %}
27     elseif name=='g+m'
28        v=0.5*exp(-x*x/(r*r))+0.5*sqrt(x*x+r*r);
29     elseif name=='m+c'
30        v=0.5*sqrt(x*x+r*r)+0.5*x^3;
31     elseif name=='m+t'
32        if x==0
33            x=0.000001;
34        end
35        v=0.5*sqrt(x*x+r*r)+0.5*x^2*log(x);
36     elseif name=='g+c'
37        v=0.5*exp(-x*x/(r*r))+0.5*x^3;
38     elseif name=='g+t'
39        if x==0
40            x=0.000001;
41        end
42        v=0.5*exp(-x*x/(r*r))+0.5*x^2*log(x);
43     elseif name=='inv+c'
44        if x==0
45            x=0.000001;
46        end
47        v=0.5*(x*x+r*r)^(-1/2)+0.5*x^3;
48     elseif name=='inv+t'
49        if x==0
50            x=0.000001;
51        end
52        v=0.5*(x*x+r*r)^(-1/2)+0.5*x^2*log(x);
53     %}
54     end

```

```

1 function [p]=matrixP(data,cc,inW,r,name)
2 %data used cmyklab or labcmyk is normalized
3
4 %col, number of dimensions in the input data, cmyk 4, lab 3
5
6 %cc centers %r ratio in rbf functions
7
8 %name name of rbf functions
9
10 [inL, ind]=size(data); [cn,cm]=size(cc);
11
12 for i=1:inL
13     for j=1:cn
14         x=data(i,1:inW)-cc(j,1:inW);
15         p(i,j)=rbfFunction(x,r,name);
16     end
17 end

```

```

1 function [A]=getRBFMatrix(data,p,inW)
2
3 [m,n]=size(data); A=ones(m,1); A=[A,data(:,1:inW),p];
4
5 \edn{lstlisting}
6
7 \begin{lstlisting}
8 function [w]=rbfWeight(A,data,inW,outDim,fname2)
9
10 %data data that used for generating model cmyklab or labomyk
11 %inW, number of dimensions in the input of data
12
13 %outDim: the number of dimension in the output of data,
14 e.g.,cmyklab 1 is %1, a is 2, b is 3.
15
16 %fname2, the name of method use in solving least squares, eg.,
17 svd
18
19 %w, the weighting factor for output dim, and cc
20
21 b=data(:,inW+outDim); w=leastSquare(A,b,fname2);

```

```

1 function [out]=getRBFResults(A,wt)
2
3 %data data that used for generating model cmyklab or labomyk
4 %inW, number of dimensions in the input of data %matrix p from
5 matrixP %wt the weight for rbf cnt0=0; cnt1=0; out=A-wt;
6 [m,n]=size(out); for i=1:m
7     for j=1:n
8         if(out(i,j)<0)
9             out(i,j)=0;
10        elseif (out(i,j)>1)
11            out(i,j)=1;
12        end
13    end
14 end

```

```

1 function out=RBFConversion(tdata,CC, wt,r,fname)
2
3 [m,n]=size(tdata); [row,col]=size(CC);
4
5 outW=col-n;
6
7 for i=1:outW
8     tp=matrixP(tdata,CC,n,r(i),char(fname(i)));
9     tA=getRBFMatrix(tdata,tp,n);
10    out(:,i)=getRBFResults(tA,wt(:,i));
11    clear tp tA ;
12 end

```

```

1 function [maxDif,avgDif]=testRBF(tdata,mdata,cc,inW,r,fname)
2
3 [m,n]=size(tdata);
4 outW=n-inW;
5
6 for i=1:outW
7     p=matrixP(mdata,cc,inW,r(i),char(fname(i)));
8     mA=getRBFMatrix(mdata,p,inW);
9     tp=matrixP(tdata,cc,inW,r(i),char(fname(i)));
10    tA=getRBFMatrix(tdata,tp,inW);
11
12
13    wt=rbfWeight(mA,mdata,inW,i,'svd');
14    out(:,i)=getRBFResults(tA,wt)*100;

```

```

15     if(i>1)
16         out(:,i)=out(:,i)*2.55-128;
17     end
18     clear wt p mA tp tA ;
19 end for i=1:m
20     % dif(i)=deltaE00(out(i,:), tdata(i,inW+1:inW+3));
21     dif2(i)=sqrt((out(i,1)-tdata(i,inW+1))^2+(out(i,2)-tdata(i,inW+2))^2+(out(i,3)-tdata(i,inW+3))^2);
22 end avgDif=mean(dif2); maxDif=max(dif2);

```

```

1 function [x,resnorm]=tstoptimization(mdata,CC,col,inW,fname)
2
3 m=length(mdata); n=length(CC);
4
5 for i=1:m
6     xdata(i,1)=1;
7     xdata(i,2:5)=mdata(i,1:4);
8     for j=1:n
9         xdata(i,j+5)=norm(mdata(i,1:inW)-CC(j,1:inW));
10    end
11 end
12
13
14
15 ydata=mdata(:,col+inW); size(ydata) len=n+6;
16
17 x0(1)=0; x0(2:len)=1;
18
19 if col==1
20     if fname=='m'
21         f=rbfld_mL(x0,xdata);
22         [x,resnorm] = lsqcurvefit(@rbfld_mL,x0,xdata,ydata);
23     elseif fname=='g'
24         f=rbfld_gL(x0,xdata);
25         [x,resnorm] = lsqcurvefit(@rbfld_gL,x0,xdata,ydata);
26     elseif fname=='inv'
27         f=rbfld_imL(x0,xdata);
28         [x,resnorm] = lsqcurvefit(@rbfld_imL,x0,xdata,ydata);
29     end
30 else
31     if fname=='m'
32         f=rbfld_mab(x0,xdata);
33         [x,resnorm] = lsqcurvefit(@rbfld_mab,x0,xdata,ydata);
34     elseif fname=='g'
35         f=rbfld_gab(x0,xdata);
36         [x,resnorm] = lsqcurvefit(@rbfld_gab,x0,xdata,ydata);
37     elseif fname=='inv'
38         f=rbfld_imab(x0,xdata);
39         [x,resnorm] = lsqcurvefit(@rbfld_imab,x0,xdata,ydata);
40     end
41 end

```

```

1 function [tdata,mdata]=partitionKFold(data,k,n)
2
3 row=length(data); len=floor(row/k); str=(n-1)*len+1; stp=n*len;
4
5 tdata=data(str:stp,:); if (n>1 & n<k)
6     mdata=[data(1:str-1,:); data(stp+1:end,:)];
7 elseif n==1
8     mdata=data(stp+1:end,:);
9 elseif n==k
10    d=row-k*len;
11    if d==0
12        mdata=data(1:str-1,:);
13    else
14        mdata=[data(1:str-1,:);data(stp+1:end,:)];

```

```

15     end
16 end

```

```

1 function out=kfoldCV(data, pdata, cc,k,
2 inW,r, fname)
3
4 [m,n]=size(data);
5
6 p = randperm(m); for i=1:m
7     ndata(i,:)=pdata(p(i),:);
8 end
9
10 for i=1:k
11     [tdata,mdata]=partitionKFold(ndata,k,i);
12     [maxDif(i),avgDif(i)]=testRBF(data,mdata,cc,inW,r, fname);
13 end
14 maxdf=mean(maxDif);
15 avgdf=mean(avgDif);
16 vardf=var(avgDif);
17 out=[avgdf vardf maxdf];
18 clear tdata mdata maxDif avgDif;

```

BP Dual Algorithm

```

1 function [A,b]=input2BPDual(data, cc, inW, col,r,name)
2
3 %data used cmyklab or labcmyk is normalized
4
5 %col, number of dimensions in the input data, cmyk 4, lab 3,
6 etc
7 %cc centers
8
9 %r ratio in rbf functions
10 %col output dim, 1,2, 3,..
11
12 p=matrixP(data,cc,inW, r, name);
13
14 A=getRBFMatrix(data,p,inW); b=data(:,inW+col);

```

```

1 function [M,wid2]=input2BPDual_r_1Dim(rdata, rCC,
2 r,name,Tlimit)
3
4 %data modeling data %cc centers for dimension 1,2,3 %r is an
5 array for the radius for the 3 dimension %name is char array
6 for the functions names for the three dimensions
7
8 [m,n]=size(rdata);
9
10
11 [A1,b1]=input2BPDual(rdata, rCC, 3, 1, r(1),char(name(1)));
12 [A2,b2]=input2BPDual(rdata, rCC, 3, 2, r(2),char(name(2)));
13 [A3,b3]=input2BPDual(rdata, rCC, 3, 3, r(3),char(name(3)));
14
15
16
17 wid2=0; for i=1:m
18     if (b1(i)+b2(i)+b3(i))>Tlimit
19         wid2=wid2+1;
20         B1(wid2,:)=A1(i,:);
21         B2(wid2,:)=A2(i,:);
22         B3(wid2,:)=A3(i,:);
23     end

```

```

24 end
25
26
27 [mA,nA]=size(A1);
28
29 tmp=zeros(nA,mA); tmp1=ones(1,mA); tmp0=zeros(1,mA);
30 tmp2=zeros(1,wid2); b4=ones(1,wid2)*Tlimit;
31
32
33 M=[A1' tmp tmp -B1'; tmp A2' tmp -B2'; tmp tmp A3' -B3'; tmp1
34 tmp0 tmp0 tmp2;tmp0 tmp1 tmp0 tmp2; tmp0 tmp0 tmp1 tmp2];
35 row=3*nA+3; M(row+1,1:mA)=-b1'; M(row+1,mA+1:2*mA)=-b2';
36 M(row+1,2*mA+1:3*mA)=-b3'; M(row+1,3*mA+1:3*mA+wid2)=b4;

```

```

1 function [a,er]=BPDual(mdata, cc, inW, col,r,fname)
2 [A,b]=input2BPDual(mdata, cc, inW, col, r,fname); M=A';
3 [n,m]=size(M); M(n+1,:)=1; M(n+2,:)=b';
4
5 k=rank(A);
6
7 [A1,cR,indx]=BPStep1(M,k); [A2,indx]=BPStep2(A1,cR,indx);
8 [a,er]=BPStep3_1(A2,indx);

```

```

1 function [a,er]=BPDual.ts(mdata, cc, r,name,Tlimit)
2
3 [A,wid2]=input2BPDual_r_Aldim(mdata, cc, inW,r,name,Tlimit);
4
5 [A1,indx]=BPStep1.ts_1(A,wid2);
6 [A2,indx]=BPStep2.ts_1(A1,indx,wid2);
7 A3=BPStep3.ts_1(A2,indx,wid2);
8 [a,er]=BPStep4.ts_1(A3,indx,wid2);

```

```

1 %This function is to calculate the first step of Barrodale and
2 Phillips due %algorithm using condensed simplex method
3
4 function [A1,indx]=BPStep1.ts_1(A,wid2)
5
6 %A is the two-d matrix including constrains and marginal cost
7 from step 1 %cR is the marginal cost row where the marginal
8 cost is zeros for base variables
9
10 [m,n]=size(A); wid=(n-wid2)/3; len=(m-4)/3;
11
12
13 rhs=zeros(m,1); rhs(m-3:m-1)=1; A(:,n+1)=rhs;
14
15 cnt=0; mc={A(end,1:3*wid),-A(end,1:3*wid)};
16
17 for i=1:m-4
18     [minV, mcol]=sort(mc);
19     for l=1:6*wid
20         col=mcol(l);
21         if col>3*wid
22             col=col-3*wid;
23         end
24         num=floor((col-1)/wid);
25         len1=num*len+1;
26         len2=(num+1)*len;
27         pCol=zeros(m-4,1);
28         pCol(len1:len2)=A(len1:len2,col);
29
30         for j=1:cnt
31             pCol(indx(j,2))=0;
32         end

```

```

33     [maxP, row]=max(abs(pCol));
34     if maxP>10^(-12)
35         A=simplexCal(A, row, col);
36         cnt=cnt+1;
37         indx(cnt, 1)=col;
38         indx(cnt, 2)=row;
39         mc=[A(end, 1:3*wid), -A(end, 1:3*wid)];
40         for p=1:cnt
41             mc(indx(p, 1))=0;
42             mc(indx(p, 1)+3*wid)=0;
43         end
44         break
45     end
46 end
47 end
48
49 A1=A; indx=sortrows(indx, 1); indx=[zeros(3, 2); indx];

```

```

1 %This function is to calculate the first step of Barrodale and
2 Phillips due algorithm using condensed simplex method
3
4 function [A2, indx]=BPStep2.ts.1(A1, indx, wid2)
5
6 %A is the two-d matrix including constrains and marginal cost
7 from step 1 %col is the coloumn number for w (error) vector
8 [m, n]=size(A1); wid=(n-1-wid2)/3; len=length(indx); A2=A1;
9 clear A1
10
11
12 for i=1:3
13     row=m-4+i;
14     w=zeros(m, 1);
15     w(row)=1;
16     str=(i-1)*wid+1
17     stp=i*wid
18     mc=zeros(1, 6*wid);
19     mc(str:stp)=A2(end, str:stp);
20     mc(str+3*wid:stp+3*wid)=-A2(end, str:stp);
21     min(mc)
22
23     for q=1:len
24         if indx(q, 1)>=0.5
25             if indx(q, 1)<=3*wid
26                 mc(indx(q, 1))=0;
27                 mc(indx(q, 1)+3*wid)=0;
28             end
29         end
30     end
31     [minV, mcol]=sort(mc);
32     [minV(1) mcol(1)]
33
34     for j=1:6*wid
35         if minV(j)<10^(-12)
36             if mcol(j)>3*wid
37                 col=mcol(j)-3*wid;
38                 A2(:, col)=2*w-A2(:, col);
39             else
40                 col=mcol(j);
41             end
42             for p=1:m-1
43                 if (A2(p, col)>10^(-12) & p~=row)
44                     A2(row, :)=A2(p, :)+2*A2(row, :);
45                     A2(p, :)=A2(p, :)-A2(row, :);
46                 end
47             end
48             if A2(row, col)>10^(-12)

```

```

49         A2=simplexCal(A2,row,col);
50         indx(i,1)=col;
51         indx(i,2)=row;
52         break
53     end
54 else
55     break
56 end
57 end
58 end clear mc

```

```

1 %This function is to calculate the first step of Barrodale and
2 Phillips due
3 %algorithm using condensed simplex method
4
5 function [A3]=BPStep3.ts.1(A2,indx,wid2)
6
7 %A is the two-d matrix including constrains and marginal cost
8 from step 1 %nc is the column number for w(error) column A3=A2;
9 clear A2; [m,n]=size(A3); wid=(n-1-wid2)/3; L=(m-4)/3;
10 len=length(indx);
11
12 lastRow=zeros(6*wid,1); lastRow(1:3*wid)=A3(end,1:3*wid);
13
14 for i=1:3
15     str=3*wid+(i-1)*wid+1;
16     stp=3*wid+i*wid;
17     lastRow(str:stp)=2*A3(end,indx(i,1))-lastRow((i-1)*wid+1:i*wid);
18 end
19
20 for i=1:len
21     lastRow(indx(i,1))=100;
22     lastRow(indx(i,1)+3*wid)=100;
23 end
24
25 [minV,mcol]=min(lastRow);
26
27 while(minV<0)
28     [tmin,tid]=sort(lastRow);
29     cnt1=0;
30     for i=1:6*wid
31         cnt1=cnt1+1;
32         if tmin(i)<10^(-12)
33             if tid(i)>3*wid
34                 col=tid(i)-3*wid;
35                 num=floor((col-1)/wid)+1;
36                 A3(:,col)=2*A3(:,indx(num,1))-A3(:,col);
37             else
38                 col=tid(i);
39             end
40             l=floor((col-1)/wid)+1;
41             len1=(l-1)*L+1;
42             len2=l*L;
43             t=[A3(len1:len2,col);A3(m-4+1,col)];
44
45
46             for j=1:L+1
47                 if t(j)==0
48                     t(j)=10^(-12);
49                 end
50
51             end
52
53             rat=A3(len1:len2,end)./t(1:L);
54             ratt=A3(m-4+1,end)/t(end);
55             rat(L+1)=ratt;

```

```

56         maxRat=abs(max(rat));
57
58
59         cnt=0;
60
61         for s=len1:len2
62             if A3(s,col)<=10^(-12)
63                 rat(s-len1+1)=maxRat+1;
64                 cnt=cnt+1;
65             end
66         end
67         if A3(m-4+1,col)<=10^(-12)
68             rat(end)=maxRat+1;
69             cnt=cnt+1;
70         end
71
72         if cnt<L+1
73             [minR,row]=min(rat);
74             if row==L+1
75                 row=m-4+1;
76             else
77                 row=row+len1-1;
78             end
79             A3=simplexCal(A3,row,col);
80             lastRow=zeros(6*wid,1);
81             lastRow(1:3*wid)=A3(end,1:3*wid);
82
83             for i=1:3
84                 str=3*wid+(i-1)*wid+1;
85                 stp=3*wid+i*wid;
86                 lastRow(str:stp)=2*A3(end,indx(i,1))-lastRow((i-1)*wid+1:i*wid);
87             end
88             for i=1:len
89                 lastRow(indx(i,1))=100;
90                 lastRow(indx(i,1)+3*wid)=100;
91
92             end
93             minV=min(lastRow)
94             miner=A3(end,end)
95             break
96         end
97     else
98         minV=12345678900;
99     end
100
101 end
102 if cnt1==6*wid
103     minV=1234567890;
104 end
105 end

```

```

1 function [a,er]=BFStep4_ts_1(A3,indx,wid2)
2
3 [m,n]=size(A3); wid=(n-1-wid2)/3; len=length(indx);
4
5 mc=zeros(n-1+3*wid,1); mc(1:n-1)=A3(end,1:n-1);
6
7 for i=1:3
8     str=n+(i-1)*wid;
9     stp=n+i*wid-1;
10    mc(str:stp)=2*A3(end,indx(i,1))-mc(str-n+1:stp-n+1);
11 end
12
13 for i=1:len
14     mc(indx(i,1))=100;
15     mc(indx(i,1)+n-1)=100;

```

```

16 end
17
18 minV=min(mc); while (minV<0)
19     [tmin,tid]=sort(mc);
20     cnt1=0;
21     for i=1:(n-1+3*wid)
22         cnt1=cnt1+1;
23         if tmin(i)<10^(-12)
24             if tid(i)>n-1
25                 col=tid(i)-n+1;
26                 num=floor((col-1)/wid)+1;
27                 A3(:,col)=2*A3(:,indx(num,1))-A3(:,col);
28             else
29                 col=tid(i);
30             end
31             t=A3(1:m-1,col);
32             for j=1:m-1
33                 if t(j)==0
34                     t(j)=10^(-12);
35                 end
36             end
37             rat=A3(1:m-1,end)./t(1:m-1);
38             maxRat=abs(max(rat));
39
40             cnt=0;
41             for k=1:m-1
42                 if t(k)<=10^(-12)
43                     rat(k)=maxRat+1;
44                     cnt=cnt+1;
45                 end
46             end
47
48             if cnt<m-1
49                 [minR, row]=min(rat);
50                 A3=simplexCal(A3,row,col);
51                 mc=zeros(n-1+3*wid,1);
52                 mc(1:n-1)=A3(end,1:n-1);
53
54                 for i=1:3
55                     str=n+(i-1)*wid;
56                     stp=n+i*wid-1;
57                     mc(str:stp)=2*A3(end,indx(i,1))-mc(str-n+1:stp-n+1);
58                 end
59
60                 for i=1:len
61                     mc(indx(i,1))=100;
62                     mc(indx(i,1)+n-1)=100;
63                 end
64                 minV=min(mc);
65                 miner=A3(end,end)
66                 break
67             end
68         else
69             minV=12345678900;
70             break;
71         end
72     end
73     if cnt1==wid2
74         minV=1234567890;
75     end
76 end
77
78
79 a=zeros(len-3,1); indx=sortrows(indx,2); if (minV~=1234567890 |
80 minV~=12345678900)
81     for i=1:len-3

```

```

82     a(i)=A3(end,indx(i,1));
83     end
84     er=A3(end,n);
85     else
86     er=0;
87     end
88
89 A4=A3; clear indx A3;

```

Batel's L_1 algorithm

```

1  function [a,er]=BRL1(data, cc, inW, col,r,name)
2
3  %This function is to solve overdetermined linear system with  $L_1$ 
4  norm based on the Eszrodale and Roberts algorithm in 1974 %The
5  code is written specifically for RBF
6
7  %data used cmykiab or labcmyk is normalized %col, number of
8  dimensions in the input data, cmyk 4, lab 3, etc %cc centers %r
9  ratio in rbf functions %col output dim, 1,2, 3,...
10
11 A=input2BRL1(data, cc, inW, col,r,name);
12 [M,ind,ab]=BRL1_step1_1(A); [a,er]=BRL1_step2_1(M,ind,ab);

```

```

1  function [A]=input2BRL1(data, cc, inW, col,r,name)
2
3  %data used cmykiab or labcmyk is normalized %col, number of
4  dimensions in the input data, cmyk 4, lab 3, etc %cc centers %r
5  ratio in rbf functions %col output dim, 1,2, 3,...
6
7  p=matrixP(data,cc,inW, r, name);
8
9  A=getRBFMatrix(data,p,inW); b=data(:,inW+col); A=[A,b];

```

```

1  function [A,B]=input2BRL1_r_1Dim(data, cc,inW,r,name,Tlimit)
2
3  %data modeling data %cc centers for dimension 1,2,3 %r is an
4  array for the radius for the 3 dimension %name is char array
5  for the functions names for the three dimensions
6
7
8  rdata=changeInputOutput(data,inW);
9  rcc=changeInputOutput(cc,inW);
10
11 [m,n]=size(data); outW=n-inW;
12
13 A1=input2BRL1(rdata, rcc, outW, 1,r(1),char(name(1)));
14 A2=input2BRL1(rdata, rcc, outW, 2,r(2),char(name(2)));
15 A3=input2BRL1(rdata, rcc, outW, 3,r(3),char(name(3)));
16
17 [mA,nA]=size(A1); z=zeros(mA,nA-1); tmpA=[A1(:,1:nA-1) z z
18 A1(:,end)]; tmpB=[z A2(:,1:nA-1) z A2(:,end)]; tmpC=[z z A3];
19
20 A=[tmpA;tmpB;tmpC];
21
22 cnt=0; for i=1:m
23     if (A1(i,end)+A2(i,end)+A3(i,end))>Tlimit
24         cnt=cnt+1;
25         B(cnt,1:3*nA-3)=[A1(i,1:nA-1) A2(i,1:nA-1) A3(i,1:nA-1)];
26         B(cnt,3*nA-2)=Tlimit;
27     end
28 end

```

```

1 function [a,er]=BRL1r(data, cc, inW, col,r,name)
2
3 %This function is to solve overdetermined linear system with L1
4 norm based on the Barrodale and Roberts algorithm in 1974
5
6 %The code is written specifically for REF
7
8 %data used cmyklab or labcmyk is normalized
9
10 %col, number of dimensions in the input data, cmyk 4, Lab 3,
11 etc
12 %cc centers
13 %r ratio in rbf functions
14
15 %col output dim, 1,2, 3,.. rdata=changeInputOutput(data,inW);
16
17 rCC=changeInputOutput(cc,inW); [m,n]=size(data); outW=n-inW;
18
19 A=input2BRL1(rdata, rCC, outW, col,r,name);
20 [M,ind,ab]=BRL1.step1(A);
21 [a,er]=BRL1.step2(M,ind,ab);

```

```

1 function [M,ind,ab,ee1,tst]=BRL1.step1(A)
2
3 %This function is to solve overdetermined linear system with L1
4 norm based on the Barrodale and Roberts algorithm in
5 1974..step1 M=A; [m,n]=size(M);
6
7 for i=1:n
8     M(m+1,i)=sum(M(:,i));
9 end
10
11 lastRow=M(end,1:n-1); cnt=0; ab=ones(n-1,1); for p=1:n-1
12     ee1(p)=M(end,end);
13     [maxC,col]=max(abs(lastRow));
14     if maxC==lastRow(col)
15         M(1:m,col)=-M(1:m,col);
16         M(end,col)=maxC;
17         ab(col)=-1;
18     end
19     t=M(1:m,col);
20     for i=1:m
21         if t(i)==0
22             t(i)=10^(-12);
23         end
24         tmp(i)=M(i,end)/t(i);
25     end
26     maxM=max(tmp)+1;
27     for i=1:m
28         if M(i,col)<=10^(-12))
29             tmp(i)=maxM;
30         end
31     end
32
33 if cnt>0
34     for i=1:cnt
35         tmp(ind(i,2))=maxM;
36     end
37 end
38
39 [val,indx]=sort(tmp);
40
41 cnt1=0;
42 for i=1:m
43     if val(i)<maxM

```

```

44         cnt1=cnt1+1;
45     else
46         break
47     end
48 end
49
50 for i=1:cnt1
51     row=indx(i);
52     if (M(end,col)-2*M(row,col)>0 & cnt1>1 & i<cnt1)
53         M(end,:)=M(end,:)-2*M(row,:);
54         M(row,:)=M(row,:);
55     else
56         if (M(end,col)-2*M(row,col)>0)
57             row=indx(cnt1);
58         end
59
60         cnt=cnt+1;
61         tst(p,1)=row;
62         tst(p,2)=col;
63         M=simplexCal(M,row,col);
64
65         ind(cnt,1)=col;
66         ind(cnt,2)=row;
67
68         %{
69         for i=1:cnt
70             if (M(ind(i,2),end)<0)
71                 ab(ind(i,1))=ab(ind(i,1))*(-1);
72             end
73         end
74
75         for i=1:m
76             if M(i,end)<10^(-12)
77                 M(i,:)=M(i,:);
78             end
79         end
80         %}
81         lastRow=M(end,1:n-1);
82
83         for k=1:cnt
84             lastRow(ind(k,1))=0;
85         end
86
87         break;
88     end
89 end
90
91     clear val indx tmp;
92 end ind=sortrows(ind,1); clear A ;

```

```

1  function [a,er,ee,tt]=BRL1_step2(M,ind,ab)
2
3  %This function is to solve overdetermined linear system with L1
4  norm based on the Barrodale and Roberts algorithm in
5  1974_step2
6
7  len=length(ind(:,1)); [m,n]=size(M);
8
9  for i=1:len
10     lastRow(i)=M(end,ind(i,1));
11 end lastRow(len+1:2*len)=-2-lastRow(1:len);
12 [maxC,mcol]=max(lastRow);
13
14 M2=M; out=M2; clear M; ct=0; while (maxC>10^(-12))
15     ct=ct+1;
16     if (mcol>len)

```

```

17         col=ind(mcol-len,1);
18         M2(1:m-1,col)=-M2(1:m-1,col);
19         M2(end,col)=maxC;
20     else
21         col=ind(mcol,1);
22     end
23     t=M2(1:m-1,col);
24     for i=1:m-1
25         if t(i)==0
26             t(i)=10^(-12);
27         end
28         tmp(i)=M2(i,end)/t(i);
29     end
30     maxM=max(tmp)+1;
31     for i=1:len
32         tmp(ind(i,2))=maxM;
33     end
34
35     for i=1:m-1
36         if M2(i,col)<10^(-12)
37             tmp(i)=maxM;
38         end
39     end
40
41     [val,indx]=sort(tmp);
42
43     cnt1=0;
44     for i=1:m-1
45         if val(i)<maxM
46             cnt1=cnt1+1;
47         else
48             break
49         end
50     end
51
52     if cnt1==0
53         break
54     else
55         cnt2=0;
56         for i=1:cnt1
57             row=indx(i);
58             if (M2(end,col)-2*M2(row,col)>0)
59                 M2(end,:)=M2(end,:)-2*M2(row,:);
60                 M2(row,:)=M2(row,:);
61                 cnt2=cnt2+1;
62             else
63                 row=indx(i);
64                 break
65             end
66         end
67         if cnt2<cnt1
68
69             tt(ct,1)=row;
70             tt(ct,2)=col;
71             M2=simplexCal(M2,row,col);
72             ee(ct)=M2(end,end);
73             for i=1:len
74                 if (M2(ind(i,2),end)<0)
75                     ab(ind(i,1))=ab(ind(i,1))*(-1);
76                 end
77             end
78             for i=1:m-1
79                 if M2(i,end)<0
80                     M2(i,:)=M2(i,:);
81
82             end

```

```

83         end
84     end
85     for i=1:len
86         lastRow(i)=M2(end,ind(i,1));
87     end
88     %lastPow(len+1:2*len)=-2-lastRow(1:len);
89     [maxC,mcol]=max(lastRow);
90     maxC
91     %M2(:,end)
92
93
94 end
95
96 % {
97     for i=1:len
98         lastRow(i)=M2(end,ind(i,1));
99     end
100    lastRow(len+1:2*len)=-2-lastRow(1:len);
101    [maxC,mcol]=max(lastRow);
102
103 end
104 % }
105 end
106
107 %M2(:,end) a=zeros(n-1,1); for i=1:n-1
108     for j=1:len
109         if (i==ind(j,1))
110             a(i)=M2(ind(j,2),end)*ab(ind(j,1));
111             break
112         end
113     end
114 end
115
116 er=M2(end,end);
117 clear M2 lastRow ind;

```

Newton color conversion

```

1 function [cmykP]=newtonCMYK2Lab(cmyk0,lab0,labP,wt,CC,r,fname)
2
3 dE=norm(lab0-labP);
4
5 cnt=0; k=zeros(3,1); cnt2=0; cnt3=0;
6
7 while dE>10^(-5)
8     cnt=cnt+1;
9     dfM=RBFDFMatrix(cmyk0,CC,wt,r,fname);
10    b=normalizeLab(labP)-normalizeLab(lab0);
11    if k==0
12        dCMY=inv(dfM(:,1:3))*b';
13        cmyk0=[dCMY; 0]'+cmyk0;
14    else
15        cnt1=0;
16        for i=1:3
17            if k(i)==0
18                cnt1=cnt1+1;
19                dfM2(:,cnt1)=dfM(:,i);
20            end
21        end
22        dfM2(:,cnt1+1)=dfM(:,4);
23        if cnt1==2
24            dCMY=inv(dfM2)*b';
25            if k(1)>0
26                cmyk0=[0;dCMY]'+cmyk0;

```

```

27         elseif k(2)>0
28             cmyk0=[dCMY(1);0;dCMY(2:3)]'+cmyk0;
29         else
30             cmyk0=[dCMY(1:2);0;dCMY(3)]'+cmyk0;
31         end
32     elseif cnt1==1
33         cnt2=cnt2+1
34         dCMY=leastSquares(dfM2,b','svd');
35         tmp=[0;0;0;dCMY(2)];
36         for i=1:3
37             if k(i)==0
38                 tmp(i)=dCMY(1);
39                 break
40             end
41         end
42         cmyk0=tmp'+cmyk0;
43     elseif cnt1==0
44         cnt3=cnt3+1
45         dCMY=leastSquares(dfM2,b','svd');
46         cmyk0(4)=cmyk0(4)+dCMY;
47     end
48 end
49 clear dfM2;
50 for i=1:3
51     if cmyk0(i)>1
52         cmyk0(i)=1;
53         k(i)=1;
54     elseif cmyk0(i)<0
55         cmyk0(i)=0;
56         k(i)=1;
57     end
58 end
59 end
60 lab0=RBFConversion(cmyk0,CC,wt,r,fname);
61 dE=norm(lab0-labP);
62 if cnt>50
63     cmyk0=0;
64     break
65 end
66 end cmykP=cmyk0; clear cmyk0

```

```

1 function [out]=newtonColorConversion(cmy2cmyk,mdata,CC,r,fname)
2
3 [m,n]=size(cmy2cmyk); wt=RBFWeights(mdata,CC,4,r,fname); for
4 i=1:m
5     cmyk0=cmy2cmyk(i,8:11);
6     lab0=cmy2cmyk(i,12:14);
7     labP=cmy2cmyk(i,5:7);
8     cmykP(i,:)=newtonCMYK2Lab(cmyk0,lab0,labP,mdata,wt,CC,r,fname);
9 end out=[cmy2cmyk(:,1:3) cmykP];

```

Bibliography

- [1] *The Role of ICC Profiles in a Color Reproduction System*, International Color Consortium (ICC) White Paper #7.
- [2] R.W.G. Hunt, *Measuring Colour*, Second Edition, Ellis Horwood Limited, England, 1991.
- [3] *Specification ICC.1:2004-10 (Profile version 4.2.0.0) Image technology colour management- Architecture, profile format, and data structure.*
- [4] Y.Qiao, R. Berns, L. Riniff, and E. Montag, *Visual Determination of Hue Suprathreshold Color-Difference Tolerance* Colour Research and Application, **23**:302-313, 1998.
- [5] M.R.Luo, G.Cui, and B. Rigg, *The Development of the CIE 2000 Colour-difference Formula: CIEDE2000*, Colour Research and Application, **26**(5):340, 2001.
- [6] R.S. Berns, D.H. Alman, L. Reniff, G.D. Snyder and M.R. Balonon-Rosen, *Visual Determination of Suprathreshold color-difference Tolerances Using Probit Analysis*, Color Res Appl **16**: 297-316, 1991.
- [7] M.R. Luo and B. Rigg, *Chromaticity-discrimination Ellipses for Surface Colours*, Color Res Appl, **11**:25-42, 1986.
- [8] M.R. Luo and B. Rigg, *BFD(l:c) Color-difference Formula. Part I: Development of the Formula*, J Soc Dyers Colour, **103**:86-94, 1987.
- [9] M.R. Luo and B. Rigg, *BFD(l:c) Color-difference Formula. Part II: Performance of the Formula*, J Soc Dyers Colour, **103**:126-132, 1987.
- [10] *CIE. Technical Report: Industrial Colour-difference Evaluation*, CIE Pub. No. 116, Vienna: Central Bureau of the CIE, 1995.

- [11] CIE. *Technical Report: Improvement to Industrial Colour-difference Evaluation*, CIE Pub. No. 142, Vienna: Central Bureau of the CIE, 2001.
- [12] R. Ulichney, *Digital Halftoning*, The MIT press, Cambridge, Massachusetts, London, England, 1987.
- [13] L.L. Thurstone, *Psychophysical analysis*, Am. J. Psychol. **38**:368-389, 1927.
- [14] L.L. Thurstone, *A law of comparative judgment*, Psychol. Rev. **34**:273-286, 1927.
- [15] H. A. David, *The Method of Paired Comparison*, Hafner Press, New York, 1969.
- [16] A. Artusi and A. Wilkie, *Color Printer Characterization Using Radial Basis Function Networks*, SPIE Proc. The International Society for Optical Engineering, **4300**:70-80, 2001.
- [17] R.L. Hardy, *Multiquadric equations of topography and other irregular surfaces*, J. Geophys. Res. **76**(8):1905-1915, 1971.
- [18] R.L. Hardy, *Theory and applications of the multiquadric biharmonic method, 20 years of discovery 1968-1988*, Comput. Math. Appl. **19**(8/9):163-208, 1990.
- [19] M.J.L. Orr, *Introduction to Radial Basis Function Networks*, Center of Cognitive Science, University of Edinburgh, 1996.
- [20] A. Artusi, P. Campadelli, and R. Schettini, *Boosting Learning Algorithms for the Colorimetric Characterization of Printers*, Proc. WIRN'98: 283-289, Vietri, Giugn, 1998.
- [21] S. Albanese, P. Campadelli, and R. Schettini, *Inkjet Color Printer Calibration by Back-propagation*, Communication Workshop on Evaluation Criteria of Neural Net Efficiency in Industrial Applications, Vietri, November 1995.
- [22] H.R. Kang and P.G. Anderson, *Neural Network Application to the Color Scanner and Printer Calibrations*, Journal of Electronic Imaging **1**:25-135, 1992.
- [23] S. Tominaga, *A color Mapping Method for CMYK Printers*, Proc. 4th IST-SID's Color Imaging Conference: Color Science, Systems and Applications, 1996.
- [24] I. Amidror, *Scattered Data Interpolation Methods for Electronic Imaging Systems: A Survey*, Journal of Electronic Imaging **11**(2):157-176, April, 2002.

- [25] D. H. McLain, *Two dimensional interpolation from random data*, Comput. J. (UK) **19**(2):178-181, 1976; Errata, 384.
- [26] A.J. Worsey and G. Farin, *An n-dimensional Clough-Tocher interpolant*, Constructive Approximation **3**:99-110, 1987.
- [27] D. Shepard, *A Two-dimensional Interpolation Function for Irregularly Space Data*, Proceedings of the 23rd ACM National Conference, 517-524, ACM, NY, 1968
- [28] W.J. Gordon and J.A. Wixom, *Shepard's method of "metric interpolation" to bivariate and multivariate interpolation*, Math. Comput. **32**(141):253-264, 1978.
- [29] R. Sibson, *A brief description of natural neighbor interpolation*, in Interpreting Multivariate Data, V. Barnett Ed., 21-36, Wiley, Manchester, 1981.
- [30] P.C. Hung, *Colorimetric calibration in electronic imaging devices using a look-up table model and interpretations*, J. Electron. Imaging **2**(1):53-61, 1993.
- [31] H.R. Kang, *Color Technology for Electronic Imaging Devices*, SPIE Optical Engineering Press, 1997.
- [32] S.I. Nin, M. Kasson, and W. Plouffe, *Printing CIE Lab Images on a CMYK Printer Using Tri-linear Interpolation*, SPIE Proc. Color hardcopy and Graphic Arts, **1670**:316-324, 1992.
- [33] J. Morovic, *To Develop a Universal Colour Gamut Mapping Algorithms*, Ph.D Thesis, University of Derby, UK, 1998
- [34] M. Ito and N. Katoh, *Three-dimensional gamut mapping using various color difference formulae and color spaces*, Proceedings of the SPIE : Color Imaging, Device-Independent Color, Color Hardcopy, and Graphic Arts IV, 1999.
- [35] J Morovic and M R. Luo, *The Fundamentals of Gamut Mapping: A Survey*, Journal of Imaging Science and Technology, **45**(3):283-290, 2001.
- [36] J. Morovic and M.R. Luo, *The Pleasantness and Accuracy of Gamut Mapping Algorithms*, in ICPS Conf. Proc., Antwerp, Belgium, **2**:39-43, 1998.
- [37] M.C. Stone, W.B. Cowan and J.C. Beatty, *Color Gamut Mapping and the Printing of Digital Color Images*, ACM Transactions on Graphics, **7**:249-292, 1988.

- [38] J. Morovic and P.L. Sun, *The Influence of Image Gamuts on Cross-Media Colour Image Reproduction*, in Proc. 8th IST/SID Color Imaging Conf., IST, Springfield, VA, 324-329, 2000.
- [39] E.D. Montag and M.D. Fairchild, *Psychophysical Evaluation of Gamut Mapping Techniques Using Simple Rendered Images and Artificial Gamut Boundaries*, IEEE Trans. Image Proc., **6**:977-989, 1997.
- [40] T. Hoshino and R.S. Berns, *Color Gamut Mapping Techniques for Color Hard Copy Images*, in SPIE Proc., SPIE, Bellingham, WA, **1909**:152-164, 1993.
- [41] S. Nakauchi, S. Hatanaka and S. Usui, *Color gamut mapping based on a perceptual image difference measure*, Color Res. Appl., **24**:280-291, 1999.
- [42] G.J. Braun and M.D. Fairchild, *General-Purpose Gamut Mapping Algorithms: Evaluation of Contrast-Preserving Rescaling Functions for Color Gamut Mapping*, in Proc. 7th IST/SID Color Imaging Conf., IST, Springfield, VA, 167-172, 1999.
- [43] G. J. Braun, *A Paradigm for Colour Gamut Mapping of Pictorial Images*, PhD. Thesis, Rochester Institute of Technology, Rochester, NY (1999)
- [44] Y. Qiao and J.L. Mitchell, *Multi-Step Gamut Mapping Algorithm For Color Printers with Small Color Gamut*, 2005 Beijing International Conference on Imaging: Technology and Applications for the 21st Century, proceedings, 220, 2005
- [45] D.S. Broomhead and D. Lowe, *Multivariate Functional Interpolation and Adaptive Networks*, Complex Systems, **2**:321, 1988.
- [46] M. Kirby, *Geometric Data Analysis, An Empirical Approach to Dimensionality Reduction and the Study of Patterns*, John Wiley & Sons, NY, 2001.
- [47] J. Moody and C. Darken, *Fast Learning in Networks of Locally-Tuned Processing Units*, Neural Comput., **1**:281, 1989.
- [48] Y. Linde, A. Buzo, and R. Gray, *An Algorithm for Vector Quantization Design*, IEEE Transactions on Communications, **28**(1):84, 1980.
- [49] J.T. Tou and R.C. Gonzalez, *Pattern Recognition*, Reading, MA: Addison-Wesley, 1974.
- [50] T.K. Kohonen, *Self-organization and Associative Memory*, Berlin: Springer-Verlag, 1989.

- [51] Y. Qiao, M. Kirby and L. Ernst *Developing the Computational Radial Basis Function Architecture for Nonlinear Scattered Color Data*, NIP22, 373-377, 2006.
- [52] S. Chen, C.F.N. Cowan, and P.M. Grant, *Orthogonal Least Squares Learning Algorithm for Radial Basis Function Networks*, IEEE Transactions on Neural Networks, **2**(2):302, 1991.
- [53] Z. Wu, *Compactly supported positive definite radial functions*, Advances in Computational Mathematics 4, 283-292, MR 97g:65031, 1995.
- [54] H. Wendland, *Piecewise polynomial, positive definite and compactly supported radial basis functions of minimal degree*, Advances in Computational Mathematics 4, 389-396. MR 96h:41025, 1995.
- [55] M.D. Buhmann, *A New Class of Radial Basis Functions with Compact Support*, Mathematics of Computation, **70**(233):307-318, Jan, 2001.
- [56] B. Efron and R. Tibshirani, *An Introduction to Bootstrap*, Chapman Hall, 1993.
- [57] R. Kohavi, G. John, R. Long, D. Manley and K. Pfleger, *Tools with Artificial Intelligence*, IEEE Computer Society Press, 740-743, 1994.
- [58] B. Efron, *Estimating the Error Rate of a Prediction Rule: Improvement on Cross-validation*, Journal of the American Statistical Association, **78**(382):316-330.
- [59] J. Shao, *Linear Model Selection vis Cross Validation*, Journal of the American Statistical Association **88**(422):486-494, 1993.
- [60] P. Zhang, *On the Distributional Properties of Model Selection Criteria*, Journal of the American Statistical Association **87**(419):732-737, 1992.
- [61] S.M. Weiss and N. Indurkha, *Decision Tree Pruning: Biased or Optimal*, Proceeding of the Twelfth National Conference on Artificial Intelligence, AAAI Press and MIT Press, 626-632, 1994.
- [62] L. Breiman and J.H. Friedman, R.A. Olshen, and C. J. Stone, *Classification and Regression Trees*, Wadsworth International Group, 1984.
- [63] T.L. Bailey and C. Elkan, *Estimating the Accuracy of Learned Concepts*, Proceedings of International Joint Conference on Artificial Intelligence, Morgan Kaufmann Publishers, 895-900, 1993
- [64] H. Wendland, *Piecewise Polynomial, Positive Definite and Compactly Supported radial Functions of Minimal Degree*, Adv. Comp. Math., 4:389, 1995.

- [65] J.R. Rice, *Characterization of Chebyshev Approximation by Splines*, SIAM J. Math. Anal. **4**:557-567, 1967.
- [66] L.L. Schumaker, *Uniform Approximation by Tchebycheffian Spline Functions*, J. Math. Mech. **18**:369-378, 1968.
- [67] G. Nurnberger and M. Sommer, *A Remez Type Algorithm of Spline Functions*, Numer. Math. **41**:117-146, 1983.
- [68] Richard H. Bartels, A.R. Conn, and Y. Li, *Primal Methods Are Better Than Dual Methods For Solving Overdetermined Linear Systems In the L_∞ Sense?*, SIAM, **26**(3):693-726, 1989.
- [69] B. Mulansky, *Chebyshev Approximation by Spline Functions with Free Knots*, IMA J. Num. Anal. **12**:95-105, 1992.
- [70] H. Kawasaki, *A Second-order Property of Spline Functions with One Free Knot*, J. Approx. Theory **78**:293-297, 1994.
- [71] G. Meinardus, G. Nurnberger, M. Sommer, and H. Straus, *Algorithms for Piecewise Polynomials and Splines with Free Knots*, Math. Comp. **53**:235-247, 1989.
- [72] *Approximation by Univariate and Bivariate Splines*, in Second International Colloquium on Numerical Analysis, VSP, 143-153, 1994.
- [73] P.T. Breuer, *A New Method for Real Rational Uniform Approximation*, in Algorithm for Approximation, Clarendon Press Oxford, 265-183, 1987.
- [74] M. Gugat, *An algorithm for Chebyshev Approximation by Rationals with Constrained Denominators*, Constr. Approx. **12**:197-221, 1996.
- [75] I. Barrodale and C. Phillips, *An Improved Algorithm For Discrete Chebyshev Linear Approximation*, in Proc. Fourth Manitoba Conf. on Numerical Mathematics, University of Manitoba, Winnipeg, Canada, 177-190, 1974.
- [76] R.H. Bartels, A.R. Conn, and C. Charalambous, *On Cline's Direct Method For Solving Overdetermined Linear Systems in the L_∞ Sense*, SIAM, **15**(2):255-270, 1978.
- [77] R.H. Bartels, A.R. Conn, and J.W. Sinclair, *Minimization Techniques For Piecewise Differentiable Functions: The L_1 Solution To An Overdetermined Linear System*, SIAM, **15**(2):224-241, 1978.

- [78] M.J.D Powell, *Approximation Theorem and Methods*, Cambridge University Press, 1981.
- [79] K. Jonasson, *A Projected Conjugate Gradient Method for Sparse Minmax Problems*, Numerical Algorithms **5**:309-323, 1993.
- [80] K. Jonasson and K. Madsen, *Corrected Sequential Linear Programming for Sparse Minimax Optimization*, BIT **34**:372-387, 1994
- [81] C. Li, and G.A. Watson, *On Nonlinear Simultaneous Chebyshev Approximation Problems*, J Math Anal and Appl **288**:167-181, 2003.
- [82] Z. Jing and A.T. Fam, *An Algorithm for Computing Continuous Chebyshev Approximations*, Maths of Comp, **48**:691-710, 1987
- [83] K. Jonasson and G.A. Watson, *A Lagrangian Method for Multivariate Continuous Chebyshev Approximation Problems*, Multivariate Approximation Theory 2(W. Schempp and K. Zeller, eds), I.S.N.M. 61, Birkhauser (Basel), 211-221, 1982
- [84] B. Joe and R. H. Bartels, *An Exact Penalty Method for Constrained, Discrete, Linear L_∞ Data Fitting*, SIAM J. Sci. Statist. Comput., **4**:69-84, 1983.
- [85] G.A. Watson, *Choice of Norms for Data Fitting and Function Approximation*, Acta Numerica, 337-377, 1998.
- [86] Al-Subaihi and G.A. Watson, *Fitting Parametric Curves and Surfaces by L_∞ Distance Regression*, BIT 45, pp. 443-461, 2005.
- [87] G.A. Watson, *Solving Data Fitting Problem in L_p Norms with Bounded Uncertainties in the Data*, SIAM Journal on Matrix Analysis and Applications, **22**(4), 2000 .
- [88] S.G. Nash and A. Sofer, *Linear and Nonlinear Programming*, The McGraw-Hill Companies, Inc, 1976.
- [89] I. Rarrodale and F.D.K. Roberts, *An Improved Algorithm for Discrete L_1 Linear Approximation*, SIAM, **10**:839-848, 1973.
- [90] H.M. Wagner, *Linear Programming Techniques for Regression Analysis*, J. Amer. Statist. Assoc., **54**:206-212, 1959.
- [91] P. Rabinowitz, *Applications of Linear Programming to Numerical Analysis*, SIAM Rev., **10**:121-159, 1968.

- [92] I. Barrodale, *On Computing Best L_1 Approximation*, *Approximation Theory*, A. Talbot, ed., Academic Press, New York, 205-215, 1970.
- [93] I. Barrodale, *An Improved Algorithm for Discrete L_1 linear approximation*, TSR 1172, Mathematics Research Center, Madison, Wis., 1972.
- [94] R.H. Bartels, A.R. Conn, and J.W. Sinclair *Minimizing Techniques for Piecewise Differentiable Functions: the L_1 Solution to an Overdetermined Linear System*, SIAM J. Numer. Anal. **15**(2), April 1978.
- [95] A.K. Cline, *A Descent Method for the Uniform Solution of Overdetermined System of Linear Equations*, SIAM J. Numer. Anal. **13**: 293-303, 1976.
- [96] O. J. Karst, *Linear Curve Fitting Using Least Deviations*, J. Amer. Statist. Assoc. **53**, 1958.
- [97] G.O. Wesolowsky, *A New Descent Algorithm for the Least absolute Value Regression problem*, Commun. Statist. **B10**: 479-491, 1981.
- [98] A. N. Sadoski, *L_1 -norm Fit of a Straight Line*, Appl. Statist. **23**: 244-248, 1974.
- [99] R.D. Armstrong and J.W. Hultz, *An algorithm for a Restricted Discrete Approximation Problem in the L_1 Norm*, SIAM J. Numer. Anal. **14**: 555, 1977.
- [100] R.D. Armstrong, F.L. Frome, and D.S. Kung, *A Revised Simplex Algorithm of the Absolute Deviation Curve Fitting Problem*, Commun, Statist. **B8**: 175-190, 1979.
- [101] P. Bloomfield and W. Steiger, *Least Absolute Deviations Curve-fitting*, SIAM J. Sci. Statist. Comput. **1**:290-300, 1980.
- [102] P. Bloomfield and W. Steiger, *Least Absolute Deviations Theory, Applications, and Algorithms*, Birhauser, Boston, Mass., 1983.
- [103] J.E. Gentle, V.A. Sposito, and S.C. Narula, *Algorithms for Unconstrained L_1 Simple Linear Regression*, Computational Statistics & Data Analysis **6**: 335-339, 1988.
- [104] M.R. Osborne, *Finite Algorithm in Optimization and Data Analysis*, John Wiley Chichester, 1985.
- [105] G.A. Watson, *Approximation in Normed Linear Spaces*, Journal of Computational and Applied Mathematics, **121**(1-2):1-36, September 2000.
- [106] E.K.P. Chong and S.H.Zak, *An Introduction to Optimization*, 2nd Edition, Wiley-interscience Publication John Wiley & Sons, Inc., 2001.