Building Accurate and Smooth ICC Profiles by Lattice Regression

Eric Garcia, Maya Gupta; University of Washington; Seattle, WA

Abstract

A system-optimized framework is presented for learning a multi-dimensional look-up-table (LUT) from training samples. The technique, termed lattice regression, solves for an entire LUT at once by optimizing the three-fold objective of 1) low interpolation error on training data, 2) smooth transitions between adjacent LUT outputs, and 3) a steady overall functional trend. The proposed algorithm is tested for both smoothness and accuracy against state-of-the-art for color management in printers.

Introduction

Controlling image appearance across diverse image displays and capture devices requires characterizing, for each device, the mapping between the device-dependent color space and a device-independent color space or profile connection space (PCS). The standard method to estimate this color mapping for devices that have a complicated color response, such as printers, is by empirical characterization: learn the mapping from a set of training examples (e.g. obtained by measuring CIELab outputs for a number of RGB inputs). For computational efficiency, it is standard practice to estimate the mapping only for a set of regularly-sampled color values and store this as a multidimensional look-up-table (LUT) in an ICC profile [12], which is interpolated at runtime to map image colors. However, estimating the color mapping is challenging because it can be highly nonlinear (especially for printers), and the training examples may be noisy due to device instabilities and/or measurement error.

Many standard regression methods have been applied to the problem of estimating LUTs for color management. These methods generally estimate a function that fits the training samples, and then one evaluates this function at the LUT gridpoints. However, because the effect of interpolating the LUT is not taken into account when this function is estimated, the training samples (representing everything known about the desired transformation) are not guaranteed to be accurately reproduced by interpolating the LUT. This begs the question: is it possible to learn a function that is optimal with respect to the LUT interpolation in a robust manner? In this paper, we propose a framework to answer this question, termed *lattice regression*, that estimates LUTs by minimizing the regularized *interpolation* error on the training data.

Next, we review some of the related work. Then we detail the proposed lattice regression and explain why we hypothesize it will estimate accurate and smooth color mappings. After that, we describe experiments comparing the proposed lattice regression to the state-of-the-art local Tikhonov regression method for an inkjet and a laser printer in terms of color management accuracy and subjective perceptual smoothness. Note that, although these methods can be applied to both forward and inverse device characterization, our experiments focus on the problem of inverse device characterization where the placement of training examples cannot be controlled directly. The paper concludes with a discussion of the results, and some conclusions and open questions.

Related Work

Formally, the empirical estimation problem can be stated as follows: Given a $d \times n$ matrix of inputs $X = \left[x_1, \ldots, x_n\right]$ where $x_j \in \mathbb{R}^d$ and a $p \times n$ matrix of outputs $Y = \left[y_1, \ldots, y_n\right]$ where $y_j \in \mathbb{R}^p$, we are tasked with building the estimated transformation $\hat{f} : \mathbb{R}^d \rightarrow \mathbb{R}^p$. Note that the terms *input* and *output* are used with respect to the color transformation being estimated, not with respect to the device. For instance, when applied to the inverse device characterization of an RGB printer, the input space may be CIELab while the output space may be RGB. Let A be a $d \times m^d$ matrix with columns a_j which are the nodes of a regular lattice (LUT) in $\mathbb{R}^d$ with m nodes per dimension, spanning the achievable input space; herein, A is referred to as an m^d lattice. Corresponding to each lattice node a_j, is an output $b_j \in \mathbb{R}^p$ that will be interpolated when applying the LUT; denote by B the $p \times m^d$ matrix of all such outputs.

Many standard regression methods have been considered for this estimation problem, including neural networks, spline fitting, polynomial regression, and local linear regression [1, 2, 3]. Experiments at Xerox showed that the best accuracy of these four approaches was obtained with local linear regression [3]. For each gridpoint in the LUT, local linear regression fits a hyperplane to the nearby training samples to minimize squared error, then estimates that gridpoint's output value by evaluating the local hyperplane at the gridpoint. We proposed a variant of local linear regression where the neighborhood training samples are chosen to enclose the gridpoint, and the hyperplane fit is regularized with ridge regression, which decreases the estimation variance [4]. These changes provide up to 20% improvement in accuracy over the local linear regression used in the Xerox experiments [4].

Recently, we showed that further reductions in error can result by changing the ridge regularization to a Tikhonov regularization such that each local hyperplane is slanted towards the global linear trend of the training data [7]. Here, for notational simplicity, we assume all the data have been properly standard normalized – see [7] for details. Let $\beta_{\text{global}} \in \mathbb{R}^d$ be the least-squares linear regression coefficient vector fitted to all n training sample pairs $\{(x_i, y_i)\}$. Then for the lattice node $g \in R^d$, the generalized Tikhonov model is $\hat{f}(g) = g^T \beta_{\text{T}} + \bar{y}$, where $\bar{y}$ is the scalar mean of the k neighborhood training sample output values, and

$$\beta_{\text{T}} = \arg\min_{\beta \in \mathbb{R}^d} \sum_{j=1}^{k} (x_j^T \beta - y_j)^2 + \lambda \|\beta - \beta_{\text{global}}\|_2^2.$$

The closed form solution is,

$$\beta_{\mathrm{T}} = \left(X^T X + \lambda I\right)^{-1} \left(X^T Y + \lambda \beta_{\mathrm{global}}\right).$$

Note that a β_{T} is learned for each color plane independently.

Such methods do not, however, consider the effect of interpolation or the fact that the estimated points lie in a regular grid. Correcting for interpolation has been explored previously as a post-processing technique to adjust an existing ICC profile via an iterative technique [5]. A similar iterative correction technique was proposed for the two-dimensional interpolation problem of geospatial analysis [6]. Although the motivation here is the same, the current approach is not an adjustment to an existing LUT. The proposed technique arrives at a closed-form solution for lattice outputs directly from the training data.

The related work discussed above focused on color management accuracy. Another important issue is the smoothness of images transformed with a LUT, and lack of such smoothness is a noted issue with some ICC profiles [8, 9]. A standard solution to increase the smoothness given an ICC profile is to average the LUT gridpoints with their neighbors. Recently, Morovic et al. suggested only averaging neighboring LUT gridpoints along the L^* dimension. Their subjective experiment results showed that this increased perceived smoothness, while three-dimensional averaging did not produce a statistically significant increase in smoothness. Their results also showed that the proposed lightness-only smoothing decreased accuracy roughly 10%. Given an ICC profile, such post-processing smoothing methods may be the only option, but for most devices they will entail a decrease in accuracy and a reduction of the gamut. In this paper, we focus on designing smoothness into the LUT via the estimation of the LUT gridpoints.

Lattice Regression

As foreshadowed in the introduction, the main idea behind the proposed lattice regression is to choose output colors for LUT gridpoints that interpolate the training data accurately. That is, given a training sample input color (e.g. a L*a*b* value), the output color (e.g. device RGB value) interpolated from the LUT should be close to the *known* training sample output (RGB) color. The key insight is that if a linear interpolation technique (e.g. trilinear, pyramidal, or tetrahedral interpolation) is used on the LUT, the operation can be inverted to solve for the node outputs that minimize the squared error of the training data. However, unless the LUT is populated with ample training data, this solution will not necessarily be unique. Furthermore, due to device and measurement noise, it may be beneficial to avoid fitting the training data exactly. For these reasons, two forms of regularization are added. In total, the proposed form of lattice regression trades off three terms: empirical risk, Laplacian regularization, and global bias. These terms will be discussed and formalized in the following subsections.

Empirical Risk

The *empirical risk* term solves for lattice outputs that accurately interpolate the training data. That is, it solves for the $p \times m^d$ matrix of lattice outputs B that minimize the squared interpolation error on the training pairs $\{X, Y\}$. Though conceptually simple, a formal treatment requires a set of functions to annotate lattice operations in order to specify the $m^d \times n$ interpolation matrix W used to interpolate the training inputs X from the lattice A. These lattice operations are presented here primarily for reference, but the essence of this section is as follows: Given fixed node outputs B, the result of interpolating the training data from the LUT can be expressed as $\hat{f}(X) = BW$. Thus, the lattice outputs that minimize the ℓ_2 distortion of the known training outputs Y are given by

$$\arg\min_B \ \mathbf{tr}\Big(\big(BW - Y\big)\big(BW - Y\big)^T\Big). \tag{1}$$

The remainder of this section is devoted to specifying W in detail.

Without loss of generality, assume that the domain is scaled and translated such that the m^d lattice spans $[0, m\text{-}1]^d$, placing nodes at integer coordinates in $\mathbb{R}^d$.

A mapping between an index into the set of lattice nodes and coordinates of this node will prove useful. For this purpose define the function $N_m : \mathbb{N} \to \mathbb{N}^d$ that returns the d-dimensional coordinates of the jth node in an m^d lattice. The kth element of the $d \times 1$ vector $N_m(j)$ is given by

$$N_m(j)_k = \left\lfloor \frac{j-1}{m^{k-1}} \right\rfloor \bmod m$$

for $k = 1, \ldots, d$.

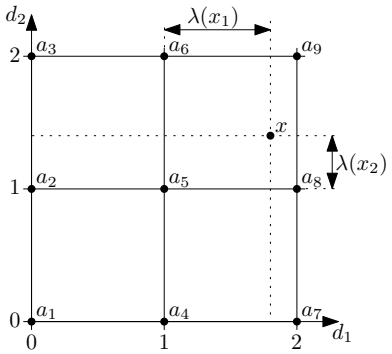

Figure 1. *The 3^2 lattice with test point $x = [1.8, 1.4]^T$.*

Interpolating a test point $x \in \mathbb{R}^d$ requires the ability to index the 2^d nodes of the cell (hypercube) containing it. For this purpose, define the function $c_j(x)$ that returns the index (in the lattice) of the jth node ($j = 1, \ldots, 2^d$) that forms a cell around x. Formally,

$$c_j(x) = \left[m^{d-1}, \ldots, m^0\right]\left(\lfloor x \rfloor + N_2(j)\right) + 1,$$

where the floor function $\lfloor \cdot \rfloor$ is performed element-wise. Note that $\lfloor x \rfloor$ and $N_2(j)$ are vectors and hence the multiplication above is an inner product.

The weights applied to the nodes of the cell containing x will depend on the interpolation method chosen. On the common three-dimensional domain of color spaces, trilinear, pyramidal, and tetrahedral interpolation are all linear weightings [10] and any one of these (or their d-dimensional extensions) could be applied for lattice regression. We will restrict our attention here to the trilinear interpolation weights.

For trilinear interpolation, $v_j(x)$ returns the weight associated with the jth node ($j = 1,\ldots,2^d$) in the cell containing x. Formally,

$$v_j(x) = \prod_{k=1}^{d} \lambda(x_k)^{N_2(j)_k} \left(1-\lambda(x_k)\right)^{1-N_2(j)_k},$$

where x_k is the kth element of x and $\lambda(x_k) = x_k - \lfloor x_k \rfloor$.

With these tools in hand, the operation of interpolating an output for x from the lattice can be expressed in matrix form. Let $w(x)$ be the $m^d \times 1$ sparse vector with kth element

$$w(x)_k = \begin{cases} v_j(x) & \text{if } k = c_j(x) \text{ for } j = 1,\ldots,2^d \\ 0 & \text{otherwise} \end{cases}$$

and for lattice outputs B, x is interpolated as $\hat{f}(x) = Bw(x)$.

Finally, given $X = \left[x_1,\ldots,x_n\right]$, W is defined as the $m^d \times n$ matrix $W = \left[w(x_1),\ldots,w(x_n)\right]$.

Laplacian Regularization

The second term in the lattice regression objective is a *Laplacian regularization* term that penalizes the squared difference of the outputs on adjacent lattice nodes, thereby averaging out device and measurement noise. Additionally, inclusion of this term guarantees a solution to (1), which is generally an underdetermined problem due to the sparsity of training data with respect to the lattice.

The graph Laplacian [11] of the lattice is fully defined by the $m^d \times m^d$ lattice adjacency matrix E with (i,j)-element

$$E_{ij} = \begin{cases} 1 & \text{if } a_i \text{ adjacent to } a_j \\ 0 & \text{otherwise.} \end{cases}$$

Given E, the normalized Laplacian is defined as

$$L = \frac{\text{diag}(\mathbf{1}^T E) - E}{\mathbf{1}^T E \mathbf{1}},$$

where $\mathbf{1}$ is the $m^d \times 1$ all-ones vector.

In quadratic form, the Laplacian measures the average smoothness of the outputs B along adjacent nodes in the lattice:

$$\mathbf{tr}\left(BLB^T\right) = \sum_{k=1}^{p} \left(\frac{1}{\sum_{ij} E_{ij}} \sum_{\{i,j \mid E_{ij}=1\}} (B_{k,i} - B_{k,j})^2 \right). \quad (2)$$

Adding (2) as regularization to (1) yields the modified objective

$$\arg\min_B \ \mathbf{tr}\left(\frac{1}{n}\left(BW - Y\right)\left(BW - Y\right)^T + \alpha BLB^T \right), \quad (3)$$

where the parameter α trades-off the average smoothness with the average fidelity to the training points.

Global Bias

Laplacian regularization rewards smooth transitions between adjacent lattice outputs but there is no incentive in (3) to extrapolate trends beyond the boundary of the cells. Consequently, initial color management experiments using Equation (3) to solve for lattice outputs produced high accuracy in the interior of the gamut, but exhibited severe clipping for saturated, dark and light colors near the edge of the gamut.

A fairly straightforward way to overcome this limitation is the inclusion of a *global bias* term: a penalty for the divergence of lattice node outputs from the overall functional trend of the training data. Since the purpose of this term is to extrapolate beyond the data, this functional form should be somewhat rigid (if it is too flexible it may extrapolate wildly). Though this is a design choice, we propose to use a global trilinear function as a bias. Fitting such a function is equivalent to solving Equation (1) for the 2^d lattice formed from the corners of the original lattice. Trilinear interpolation can then be applied to these corners to solve for outputs $\tilde{B}$ at every node in the original m^d lattice A. Thus, $\tilde{B}$ are the lattice outputs corresponding to the global trilinear function that best fits the data and the global bias term penalizes the average squared divergence from these values:

$$\frac{1}{m^d}\mathbf{tr}\left(\left(B - \tilde{B}\right)\left(B - \tilde{B}\right)^T \right). \quad (4)$$

Lattice Regression Objective Function

Combining the empirical risk minimization, the Laplacian regularization, and the global bias, the proposed lattice regression objective is given by

$$\arg\min_B \ \mathbf{tr}\Big(\frac{1}{n}\left(BW - Y\right)\left(BW - Y\right)^T + \alpha BLB^T + \frac{\gamma}{m^d}\left(B - \tilde{B}\right)\left(B - \tilde{B}\right)^T \Big),$$

where, relative to the empirical risk, α and γ control the average smoothness and the average divergence from the global bias, respectively.

This has the closed-form solution

$$B^* = \left(\frac{1}{n} YW^T + \frac{\gamma}{m^d}\tilde{B} \right) \left(\frac{1}{n} WW^T + \alpha L + \frac{\gamma}{m^d} I \right)^{-1}, \quad (5)$$

where I is the $m^d \times m^d$ identity matrix.

Experiments

The proposed lattice regression (5) was tested for color management accuracy and perceptual smoothness on an HP Photosmart D7260 ink jet printer and a Samsung CLP-300 laser printer. We compared lattice regression to a state-of-the-art color management regression technique: local Tikhonov regression using the enclosing k-NN neighborhood [4, 7].

Although lattice regression has two parameters, resource restrictions allowed only the investigation of one of the parameters. Preliminary visual experiments on the Samsung CLP-300 were performed to determine which parameter has a greater impact on smoothness. In these preliminary experiments, one parameter was held at a baseline value ($\alpha = 1$ or $\gamma = 1$) while the other was varied from $\{.1, 1, 10, 100\}$. Surprisingly, it was found that the α parameter (which favors smooth transitions in device RGB on adjacent lattice nodes) had little affect on the smoothness compared to the γ parameter (which favors a global trilinear function). That is, on the rather un-smooth Samsung CLP-300 printer (see Figure 2) larger values of α did not produce noticeably smoother prints while the larger values of γ did. Therefore, γ was chosen as the variable parameter for these experiments.

Related to our analysis of the effect of α on smoothness, Morovic et al.'s recent study [8] found a correlation between smooth transitions in device RGB between adjacent LUT nodes in the L^* dimension only and visual smoothness of images processed by such LUTs. This suggests that fixing the Laplacian parameter for the chroma dimensions but varying the Laplacian parameter in the L^* dimension may be a useful tool for designing in smoothness, but this was not investigated here.

The size of the matrix that is inverted in (5) can be quite large ($m^d \times m^d$). However, it is also quite sparse and can therefore be solved efficiently by techniques such as sparse Cholesky factorization [15]. Experiments for this paper using 17^3 LUTs were solved in under .5 seconds on an Intel 1.83GHz Core 2 Duo compared to upwards of 60 seconds for the Tikhonov method.

Accuracy Experimental Details

Training samples were measured by printing the Gretag MacBeth TC9.18 RGB target, which has 918 values, including 789 regularly-sampled RGB values and neutral ramps with 189 samples. The targets were measured with an X-Rite iSis spectrophotometer using D50 illuminant at a 2° observer angle and UV filter. A stability test of five measurement rounds of a 918-sample target with this device showed a measurement variability of less than .1 mean and 1 maximum ΔE_{2000}.

Given the 918 training samples, we built 1D gray-balanced calibration LUTs for each color channel (see [3, 4] for details), followed by a $17 \times 17 \times 17$ 3D LUT covering the range $L^* \in [0, 100]$, $a^*, b^* \in [-100, 100]$.

We built five 3D LUTs for lattice regression for regularization parameters $\gamma \in \{.1, 1, 10, 20, 50\}$, and three 3D LUTs for Tikhonov for regularization parameters $\lambda \in \{1, 10, 20\}$. (A greater number of regularization parameters for lattice regression were considered because we had no previous experiments to guide us, unlike the regularized local linear regression). For the LUTs using Tikhonov, the same algorithm used for construction of the 3D LUTs was used in the construction of 1D LUTs. However, since lattice regression was not designed to estimate 1D LUTs, the Tikhonov method with $\lambda = 1$ (the baseline parameter) was used for constructing the 1D LUTs used with the 3D lattice regression LUTs.

We tested the LUTs on 918 in-gamut CIELab values formed by randomly sampling the RGB cube, printing those values on each printer, then measuring the corresponding (in-gamut) CIELab values, and using these as test values.

Perceptual Experimental Details

We tested the perceptual smoothness of a set of in-gamut gradients that range over L^* for a fixed-chroma of primary and secondary colors. The gamut of each device was determined by the alpha-shape [13] with $\alpha = 2000$ (determined by visual inspection of the smoothness of the gamut surface) for the measured TC9.18 target. Figure 2 shows examples of the gradient test chart produced with the lattice regression ($\gamma = 1$) LUT. Although they appear slightly different after printing on the intended printer, these images do illustrate the kinds of contours that occur in the prints.

Thirteen subjects with normal color vision judged the smoothness of printed gradients produced by each method in a ranking task. For each printer, eight gradients were printed (one for each parameter-setting of the regression methods that were investigated). Separately for each printer, the subjects were asked to order the eight targets with the least smooth on the left and the smoothest on the right. This data was used to produce paired comparisons for which were scaled under a Thurstonian model [14]. The experiments were carried out with approximately D50 lighting produced by two 4700K Sollux lamps by Tailored Lighting.

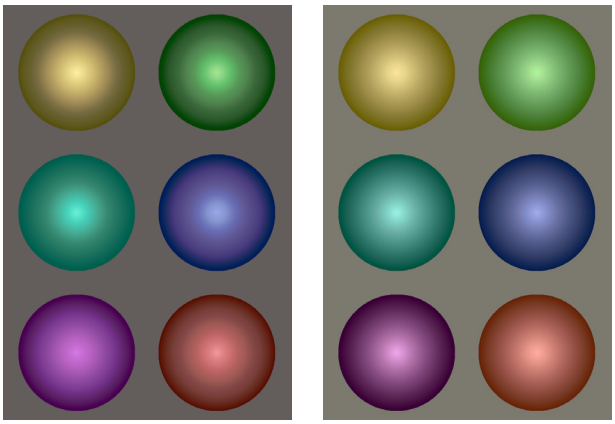

Figure 2. *Figure shows the RGB values for the gradient experiments for the Samsung laser printer (left) and HP inkjet printer (right) estimated with lattice regression with regularization parameter* $\gamma = 1$.

Results

Complete accuracy results in terms of ΔE^*_{2000} are shown in Table 1 and Table 2. One sees that the Tikhonov error rates are robust within the chosen range of λ. For the lattice regression, the errors vary greatly over the chosen range of γ, and in fact the smallest considered value $\gamma = .1$ is preferable.

Figure 3 shows the smoothness versus accuracy for all of the regression methods investigated. The relative perceptual smoothness is determined by Thurstone's law of comparative judgement (Case V) [14] applied to the pairwise comparisons induced by the ranking data. The scale at which differences can be considered statistically significant is indicated by the separation of tick-marks on the vertical axis. As hypothesized, for lattice regression, this range of γ shows a rough trade-off between smoothness and accuracy as one increases the amount of regularization. For Tikhonov regression, the relationship between λ and smoothness is less clear.

On the Samsung laser printer, lattice regression with a small bias term ($\gamma = .1$) produced 23%, 15%, and 27% lower median, 95th percentile, and maximum error compared to the best Tikhonov regression, with comparable smoothness (within the smallest significant difference). With a larger bias term ($\gamma = 10$), lattice regression produces significantly smoother prints, but at the cost of decreased accuracy and it's likely that for an intermediate value of γ, lattice regression would achieve comparable errors while producing significantly smoother prints.

On the HP inkjet printer, lattice regression is not as dominant. Although lattice regression with a small bias term ($\gamma = .01$) produced at least 18% and 12% lower median and 95th percentile error compared to the best Tikhonov regression, the maximum error is increased by 19%. Furthermore, all parameter settings of Tikhonov regression produced smoother results, with lattice regression incurring a very large penalty in accuracy for modest

gains in smoothness.

In practice, one usually does not have the ability to compare the performance for different parameter choices. In a previous experiment [4], we used total RGB estimation ℓ_2 error on 918 randomly sampled in-gamut test points, but this requires printing and measuring 918 more samples, which is costly and time-consuming. In this study we also considered cross-validating based on the ℓ_2 difference between the true RGB and estimated RGB of only 12 randomly sampled in-gamut test points. This is practical because 12 extra samples could be added to the original target sheet and printed at the same time as the training samples, producing a total of 930 measured samples, of which 918 are used to train the algorithm and 12 are used to select the best regularization parameter. Cross-validating with 12 random test points produced the cross-validated parameter choices $\gamma = .1$ for lattice regression for both the HP and Samsung, and $\lambda = 20$ for Tikhonov for both the HP and Samsung. In Table 3 we compare these cross-validated parameter choices for accuracy and perceptual smoothness. For comparing smoothness, the fraction of the time the algorithm was ranked as smoother than the other is shown. This table shows that cross-validating based only on accuracy leads to lower error with the lattice regression and comparable smoothness.

Table 1. Samsung CLP-300 laser printer

	ΔE^*_{2000} Error		
	Median	95 %-ile	Max
Lattice γ=.1	2.33	5.87	9.79
Lattice γ=1	2.77	6.48	11.74
Lattice γ=10	3.25	8.92	15.34
Lattice γ=20	3.80	9.62	17.67
Lattice γ=50	4.19	10.21	19.11
Tikhonov λ=1	3.25	7.20	14.29
Tikhonov λ=10	3.04	6.93	13.49
Tikhonov λ=20	3.14	7.05	13.48

Table 2. HP Photosmart D7260 inkjet printer

	ΔE^*_{2000} Error		
	Median	95 %-ile	Max
Lattice γ=.1	1.03	2.56	6.19
Lattice γ=1	1.22	3.25	5.50
Lattice γ=10	2.38	5.98	10.26
Lattice γ=20	2.81	6.56	10.58
Lattice γ=50	3.26	7.13	10.88
Tikhonov λ=1	1.27	2.94	5.16
Tikhonov λ=10	1.34	3.09	5.28
Tikhonov λ=20	1.38	3.06	6.10

Conclusions and Open Questions

In this paper we proposed a new regression method optimized for estimating LUTs that we term lattice regression. We showed that lattice regression in practice achieves $\sim 20\%$ lower error on an inkjet and laser printer than the state-of-the-art color management estimation method local Tikhonov linear regression

Table 3. Cross-validated Algorithm Comparison

		ΔE^*_{2000} Error		Smooth-ness
		Median	95%-ile	
Samsung	Lattice	2.33	5.87	9/13
	Tikhonov	3.14	7.05	4/13
HP	Lattice	1.03	2.56	2/13
	Tikhonov	1.38	3.06	11/13

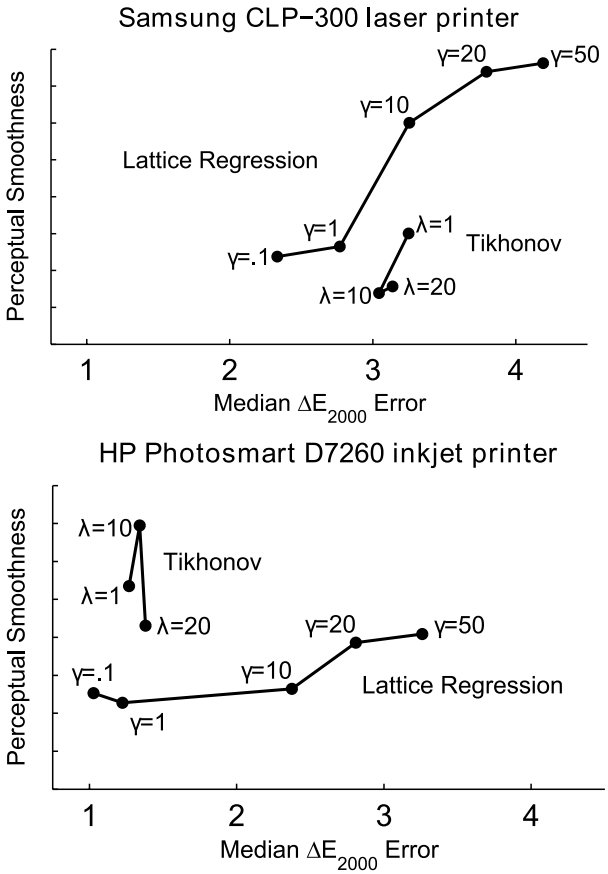

Figure 3. *Figure shows the psychometrically scaled smoothness obtained from the ranking experiment plotted against median ΔE_{2000} error for varying algorithm parameters.*

over enclosing neighborhoods. Additionally, we analyzed the perceptual smoothness of printed gradients, and showed (i) that the regularization parameter in lattice regression trades-off between smoothness and accuracy over the examined parameter range; and (ii) that for cross-validated parameter choices for lattice regression and the Tikhonov method the perceived smoothness is comparable.

A number of open questions remain. First, the cross-validated parameter choices were at the extremes of the considered ranges for both lattice regression and Tikhonov, and a future study should consider a wider range of parameter choices. Second, lattice regression is designed to minimize error on the training samples after trilinear interpolation, but not all color management systems use trilinear interpolation to interpolate the LUT. We hypothesize that lattice regression will do almost as well if

tetrahedral or another linear interpolation method is used, but this requires testing.

For the smoothness tests, we focused on in-gamut gradients only, so as not to confound the estimation of the LUTs with gamut mapping. However, both estimation methods considered here do estimate LUT gridpoints outside the gamut. This raises a number of interesting open questions: Do these methods effectively estimate points outside the gamut? Is the out-of-gamut estimation similar to any popular gamut-mapping strategies? Would it be helpful to design-in gamut mapping to the estimation of the LUT?

Lastly, we had hoped that the smoothing terms in lattice regression would lead to smooth RGB outputs, but the smoothness results were comparable to those with Tikhonov, and so we consider our effort in this respect to be unsuccessful. It is an open question how to design in smoothness to the LUT estimation, and how to automatically choose regression parameters to produce appreciably smooth results.

References

[1] H. R. Kang and P. G. Anderson, Neural network application to the color scanner and printer calibrations, Journal of Electronic Imaging, vol. 1, pp. 25–135 (1992).

[2] A. Artusi and A. Wilkie, Novel color printer color characterization model, Journal of Electronic Imaging, vol. 12, no. 3, pp. 448–458 (2003).

[3] R. Bala, Digital Color Handbook, chapter 5: Device Characterization, CRC Press, pp. 269–384 (2003).

[4] M. R. Gupta, E. K. Garcia, and E. M. Chin, Adaptive local linear regression with application to printer color management, IEEE Trans. on Image Processing, vol. 17, no. 6, pp. 936–945 (2008).

[5] R. Bala, Iterative technique for refining color correction look-up tables, U.S. Patent 5,649,072 (1997).

[6] W. R. Tobler, Lattice Tuning, Geographical Analysis, vol. 11, no. 1, pp. 36–44 (1979).

[7] N. Hrustemovic and M. R. Gupta, Multiresolutional regularization of local linear regression over adaptive neighborhoods for color management, Proc. of the Intl. Conf. on Image Processing (2008).

[8] J. Morovic, A. Albarran, J. Arnabat, Y. Richard, and M. Maria, Accuracy-preserving smoothing of color transformation LUTs, Proc. 16th IS&T Color Imaging Conference pp. 243–246 (2008).

[9] T. Olson, Smooth Ramps: Walking the Straight and Narrow Path through Color Space, Proc. 7th IS&T Color Imaging Conference, pp. 57–64 (1999).

[10] R. Bala and R. V. Klassen, Digital Color Handbook, chapter 11: Efficient Color Transformation Implementation, CRC Press (2003).

[11] F. Chung, Spectral Graph Theory, Nunmber 92 in Regional Conference Series in Mathematics, American Mathematical Society, (1997).

[12] D. Wallner, Building ICC Profiles - the Mechanics and Engineering, chapter 4: ICC Profile Processing Models, pp. 150-167, International Color Consortium (2000).

[13] T. J. Cholewo and S. Love, Gamut boundary determination using alpha-shapes, Proc. 7th IS&T Color Imaging Conference, pp. 200–204 (1999).

[14] P. Engeldrum, Psychometric Scaling, chapter 8: Indirect Interval Scaling - Case V and Paired Comparison, Imcotek Press, pp. 93–108 (2000).

[15] I. S. Duff, Direct methods, Technical Report RAL-98-056, Rutherford Appleton Laboratory, Oxfordshire, UK (1998).

Author Biography

Eric Garcia completed his MS EE (2006) at the University of Washington in Electrical Engineering and continues research there as a doctoral student. He received a BS in Computer Engineering (2004) from Oregon State University. Garcia was an Intel GEM fellow (2004-2007), and his research focuses on statistical learning and color image processing.

Maya Gupta completed her M.S. (1999) and Ph.D. (2003) in Electrical Engineering at Stanford University as a National Science Foundation Graduate Fellow. She received a BS in Electrical Engineering and a BA in Economics from Rice University (1997). From 1999-2003 she worked for Ricoh's California Research Center as a color image processing research engineer. Since 2003, she is an Assistant Professor of Electrical Engineering at the University of Washington. Her awards include the PECASE (2007) and ONR YIP (2007).