Comparison of regression methods and neural networks for colour correction

A. Kucuk[1], G. D. Finlayson[1], R. Mantiuk[2], M. Ashraf[3]

[1]University of East Anglia, UK; [2]University of Cambridge, UK; [3]University of Liverpool, UK

Abstract

Colour correction is the problem of mapping the sensor responses measured by a camera to the display-encoded RGBs or to a standard colour space such as CIE XYZ. In regression-based colour correction, camera RAW RGBs are mapped according to a simple formula (e.g. a linear mapping). Regression methods include least squares, polynomial and root-polynomial approaches. More recently, researchers have begun to investigate how neural networks can be used to solve the colour correction problem.

In this paper, we investigate the relative performance of regression versus a neural network approach. While we find that the latter approach performs better than simple least-squares the performance is not as good as that delivered by either root-polynomial or polynomial regression. The root-polynomial approach has the advantage that it is also exposure invariant. In contrast, the Neural Network approach delivers poor colour correction when the exposure changes.

Introduction

It is well known that the sensors in cameras sample light differently than we do. Indeed, there are pairs of different surfaces that look the same to a human observer but induce different sensor responses in a camera and vice versa. Colour correction algorithms attempt to map the camera responses either to the RGBs that drive a display (e.g. the linear sRGB matching responses for a display with Rec 709 primaries [1]) or, equivalently, to the standard human vision system referenced coordinate systems such as CIE XYZ [2].

Figure 1. *These images are generated from David Foster's hyperspectral reflectance dataset [3] with Nikon D5100 camera responses and D65 illumination. While the left image (a) is representing RAW RGBs, the right one (b) demonstrates the colour corrected sRGB image.*

The colour correction problem is illustrated in Figure 1. On the left is the raw RGB camera response for a scene measured with a Nikon D5100 camera and on the right after linear colour correction to the sRGB display space [1]. The scene is taken from Foster et al. hyperspectral image set [3] and the images are calculated using numerical integration. The numerical integration results in linear colour values so the sRGB non-

linearity [4] was applied to both images for display (otherwise they would appear too dark).
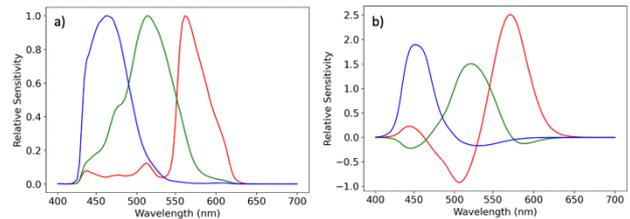

Figure 2. *Normalised sensitivity functions of Nikon D5100 camera (left) and sRGB Sensitivity functions (right).*

According to the sRGB standard [1] the sRGB matching curves 'see' the RGBs that would correct drive a Rec 709 display to reproduce the colours we see (when viewing the scene) correctly. We contrast the Nikon and sRGB sensitivities in Figures 2a and 2b. If we could find a linear transform of the Nikon sensitivities that mapped exactly the sRGB curves then the Nikon camera would be able to measure colours that we perceive [5] [6]. No such linear transform exists.

Colour correction algorithms – to the extent it is possible – attempt to map camera responses to corresponding coordinates in a human referenced colour space. While this referenced colour space could be sRGB we will, henceforth, consider colour correction as the problem of mapping camera responses to XYZ tristimuli [2]. The XYZ colour matching functions are a linear transform from sRGB – and so the latter are calculable from the former. Importantly, the XYZ responses (called XYZ tristimuli) are used directly in the formula that compute perceptual colour difference.

The simplest colour correction algorithm finds a 3×3 matrix mapping camera responses to XYZs by least squared regression. Given a set of camera responses together with their corresponding XYZ tristimuli, a 3×3 colour correction matrix M is found such that:

$$M\boldsymbol{\rho} \approx \boldsymbol{x} \qquad (1)$$

where $\boldsymbol{\rho}$ and $\boldsymbol{x}$ denote the 3-component RGB camera response vector (RAW RGB value) and XYZ tristimulus respectively, both in a linear colour space. Other commonly used methods include polynomial [7] and root-polynomial [8] regressions.

Recently, researchers have investigated how neural networks (NNs) might be used to solve the colour correction problem. Initial results show that NN algorithms support better colour correction than a simple linear regression. Here we take the MacDonald and Meyer method [9] that has recently been developed and compare and contrast its performance with a commonly used set of regression methods.

Broadly, we confirm the finding that the NN approach is significantly better than linear regression but that the polynomial [7] and root-polynomial [8] regressions actually deliver significantly better colour correction than the NN. Further, we also consider the performance of the different approaches when exposure changes (i.e. the correction method is trained with respect to one level of illuminant but then tested with respect to another). Here linear and root-polynomial methods work well but, the polynomial regression and the NN approach perform poorly.

Background

Let $Q_k(\lambda)$ denote the k-th camera spectral response function and $\boldsymbol{Q}(\lambda)$ denote the vector of these functions as in Figure 2a. The camera response to a spectral power distribution $E(\lambda)$ illuminating the j-th reflectance $S_j(\lambda)$ is written as:

$$\boldsymbol{\rho} = \int_\omega \boldsymbol{Q}(\lambda)E(\lambda)S_j(\lambda)d\lambda \tag{2}$$

where ω denotes the visible spectrum (400 to 700 Nanometres) and $\boldsymbol{\rho}$ denotes the vector of RGB responses. Similarly, given the XYZ colour matching $\mathbf{X}(\lambda)$, the tristimulus response $\mathbf{x}$ is written as:

$$\mathbf{x} = \int_\omega \mathbf{X}(\lambda)E(\lambda)S_j(\lambda)d\lambda \tag{3}$$

Suppose, respectively, in $n \times 3$ matrices P and X record (in rows) the camera responses and tristimuli of n surface reflectances. To find the M in Equation 1 we minimise:

$$\arg\min_M \|PM - X\|_F \tag{4}$$

Where $\| \, . \, \|_F$ denoted the L2 norm [10]. We can solve for M in closed form using the Moore-Penrose Inverse [11]:

$$M = [P^T P]^{-1} P^T X \tag{5}$$

To extend the regression method we define a basis function $f_e^o()$ where the subscript e denotes the type of expansion — here $e{=}p$ and $e{=}r$ respectively denotes polynomial and root-polynomial expansions — and the superscript o denotes the order of the expansion. As an example, if we are using the 2nd order root-polynomial expansion [8] then we write:

$$f_r^2(\boldsymbol{\rho}) = \left[\, \text{r g b} \sqrt{rg} \ \sqrt{rb} \ \sqrt{bg} \,\right]^T \tag{6}$$

Again we can use Equations 4 and 5 to solve for the regression matrix M. Though, M will be non-square (and depend on the number of terms in the expansion). For our second order root-polynomial expansion, the columns of P will be the 6 terms in the root-polynomial expansion (P is a $n \times 3$ matrix) and M will be 6×3.

Optimizing for L2 norm in Equation 4 may be undesirable because the Euclidean differences in the XYZ colour space do not correspond to the perceived differences in colour. Instead, it is more desirable to optimize for the differences in perceptually uniform colour spaces, such as CIELAB [2] - or using colour difference formulas, such as CIE Delta E 2000 [12]. Let us denote the magnitude of the difference vector between a mapped camera response vector and its corresponding ground truth CIELAB value as:

$$\Delta^{M,o,e} = \|C(M^T f_e^o(\boldsymbol{\rho}), \boldsymbol{w}) - C(\boldsymbol{x}, \boldsymbol{w})\| \tag{7}$$

Where C() maps input vectors according to the CIELAB function to corresponding Lab triplets and the superscripts e and o are as before. The parameter $\boldsymbol{w}$ denotes the XYZ tristimulus of a perfect white diffuser and is required to calculate CIELAB values. To find the best regression matrix according to we would need to minimize:

$$\arg\min_M \sum_{i=1}^n \Delta_i^{M,o,e} \tag{8}$$

Unfortunately, there is no closed form solution for minimising Equation 8. Instead, a search-based strategy such as the Nelder-Mead simplex method [13] can be used to find M (though there is no guarantee that the global optimum result is found, [13] is a local minimiser).

Rather than use regression we could - in line with the ever-expanding area of machine learning - deploy an Artificial Neural Net to solve for colour correction. In the context of this paper, we will consider the network proposed by MacDonald and Meyer [9], shown in Figure 3.
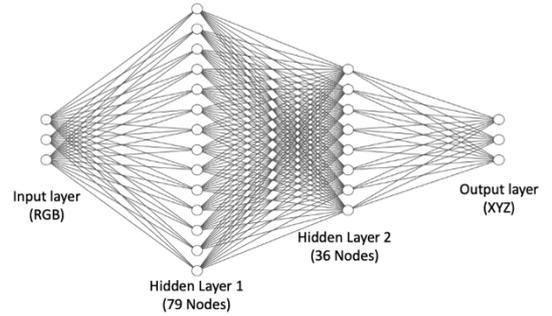

Figure 3. *The demonstrative architecture of the neural network. Input and output layers consist of 3 nodes which are RGB and XYZ respectively. In between, there are 2 hidden layers formed by 79 and 36 nodes.*

A potential problem of the Neural Network shown in Figure 3 is that colour correction is much more computationally expensive. There are 3189 'connections' in the network indicating O(3189) multiplications and additions need to be carried out as data flows from left to right. The complexity of the 2nd order root-polynomial correction in contrast has 3 square root operations and (when the 6x3 correction matrix is applied) 18 multiplications and 15 additions i.e. it is 2 orders of magnitude quicker to compute.

Experiments

For the experiments, we used the Simon Fraser University (SFU) reflectance set [14], which is a composite set and comprises 1995 spectral surface reflectances including, the 24 Macbeth colour checker patches, 1269 Munsell chips, 120 Dupont paint chips, 170 natural objects, as well as another 407 additional surfaces.

In our experiments, this reflectance set is viewed under D65 illumination. Camera responses - for the Nikon D5100 (see Figure 2) and XYZ tristimuli are calculated by numerical integration. In Figure 4, in CIE 1931 chromaticity diagram we plot the xy chromaticities of the SFU dataset. We also show the gamut of colours achievable using Rec 709 primaries (white triangle). It is evident that the SFU reflectance set comprises a wide range of colours.
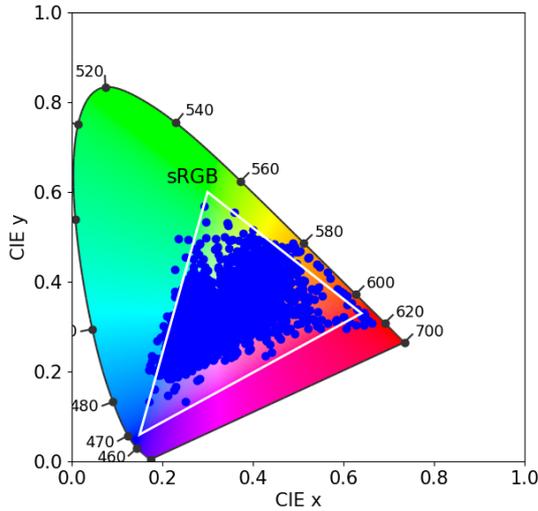
Figure 4. *Gamut of SFU dataset on the CIE 1931 chromaticity diagram. The white triangle shows the sRGB gamut.*

We run our experiments according to a 5-fold cross-validation procedure. Here each data set is randomly split into 5 folds with ~400 reflectance samples in each fold. Four of the folds are then used to train the colour correction algorithms and then the fifth fold is used as testing data. We repeat this process 5 times (each fold is used once as a test set).

We evaluate the following algorithms:

(i) **LS:** denotes Least Squares Regression (denoted)
(ii) **LS-P:** denotes Least Squares Polynomial Regression. Here we use the 2nd order expansion which maps each 3 vector to a 10 vector.
(iii) **LS-RP:** Least Squares Root-Polynomial Regression. Again a 2nd order expansion is used which for root-polynomials has 6 terms
(iv) **LS-Lab**
(v) **LS-P-Lab**
(vi) **LS-RP-Lab**
(vii) **NN**

Where **Lab** denotes that we use the CIELAB loss values for training with Nelder-Mead simplex method [13] (see Equations 7 and 8) to solve for the linear (iv), polynomial (v) and root-polynomial regressions (vi) where CIELAB error is minimised. The regressions (i) through (iii) - minimising error in XYZ tristimuli space - are found in closed form using the Moore Penrose inverse (Equations 5 and 6).

We compare the regression algorithms with the 3x79x36x3 architecture **Neural Network (NN)** from [9] illustrated in Figure 3. As in the original study, we train the network to minimise CIE Delta E 2000 [12] using the Adam optimizer with a learning rate of 0.001. Since we were working with relatively small datasets, we had to increase the number of epochs from 65 (used in the original study) to 500 for the neural network to learn a good mapping. We also used 20% of the training data for the validation set and applied the early stopping technique, which means the training finishes automatically if there is no improvement in validation loss for a certain number of epochs

(which is 100 in our model), with a call-back method. We used the best model according to the validation loss. Our model used a mini-batch gradient descent with a batch size of 8.

In addition to comparing the methods under the constant exposure level, we also investigate how they perform when the exposure level changes. We take our models trained for a fixed intensity of D65 and then change the exposure by multiplying the RGB and XYZ values by a scalar (in the range 0.2 to 5). By construction, the LS, LS-RP, LS-Lab, and LS-RP-Lab models are exposure invariant, so their correction performance is unaffected by exposure change. So, we will only evaluate and report results for the performance of LS-P, LS-P-Lab and NN since they are not exposure invariant.

Finally, we note that the colour correction experiments here differ from those reported in the original NN paper [9]. The data in [9] is not publicly available. Moreover, because the code that implements [9] is also not available, we reimplemented their method.

Results

We report the CIELAB and CIE Delta E 2000 error results for our 7 algorithms for SFU dataset in Table 1 and Table 2, respectively. As discussed earlier, we run a 5-fold cross validation experiment. This means that there are 5 sets of colour correction results and for each of these we calculate the Mean, Max, Median and 95% (percentile) errors. The figures in the Tables 1 and 2 report these error statistics averaged over the 5 sets. In bold red we show the best result per statistic. Note that since the 3 algorithms trained to minimize CIELAB Delta E (i.e. those with the suffice '-**Lab**') the corresponding algorithms' results in Table 2 (for Delta E 2000) could clearly be improved further i.e. if the regression matrices were found that best minimized the DE2000 error.

Table 1: CIELAB Delta E statistics

	Mean	Max	Med	95%
LS	1.62	15.47	0.93	5.32
LS-P	1.29	11	0.78	4.01
LS-RP	1.19	13.97	**0.7**	3.62
LS-Lab	1.48	10.62	0.9	4.63
LS-P-Lab	1.17	8.47	0.77	3.62
LS-RP-Lab	**1.10**	**7.36**	0.72	**3.39**
NN	1.40	12.26	0.93	4.06

Table 2: CIE Delta E 2000 statistics

	Mean	Max	Med	95%
LS	0.94	7.71	0.7	2.62
LS-P	0.79	4.63	0.59	2.18
LS-RP	0.72	7.13	**0.49**	2.14
LS-Lab	0.91	5.52	0.67	2.45
LS-P-Lab	0.75	4.25	0.54	2.08
LS-RP-Lab	**0.69**	**3.81**	0.52	**1.96**
NN	0.85	4.18	0.66	2.15

We see that, although the neural network algorithm returns significantly better results compared with the standard Least Squares Regression model, it delivers poorer colour correction compared with the other regression methods. The Polynomial and Root-Polynomial deliver low error, and both can be improved using the differences in the CIELAB space as a loss function. The **-Lab** variants of the regression algorithms have significantly lower maximum errors. The reader might be interested if we train the neural network with the CIELAB loss function or use CIE Delta E 2000 on the classical methods, we found that these do not change the rank ordering of the results.

In Figure 5 and Figure 6, we plot the CIELAB and CIE Delta E 2000 error distributions as violin plots for 4 of our algorithms. Inside each 'violin', the white dot, the horizontal line, the black bar and the black line indicate respectively the median, mean, interquartile range and 1.5 interquartile range. The tails of the violin plots are long because of the large maximum values (outliers). The width of the violin captures the probability density of the error distribution [15]. Again, the distributions show that the Root-Polynomial Regression with the CIE Lab loss shows the best performance.
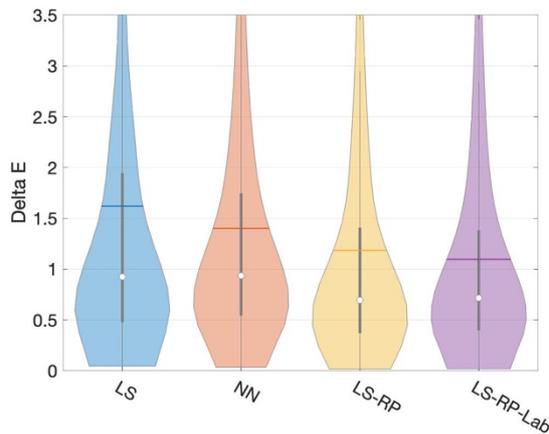

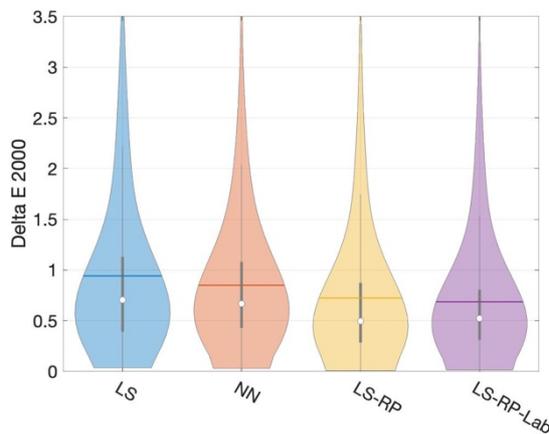*Figure 5. A comparison of CIELAB error distribution of 4 methods on SFU Dataset.*

Figure 6. A comparison of CIE Delta E 2000 error distribution of 4 methods on SFU Dataset.

We performed the sign test [16] to establish if there is a statistically significant difference between the NN and LS-RP-Lab results. Both the p-values for CIELAB and CIE Delta E 2000 are less than 0.0001 in the 99% confidence level. The difference in algorithm performance is statistically significant.

Now we calculate the mean CIELAB Delta E (again in the 5-fold cross validation way) for **LS-P**, **LS-P-Lab** and **NN** when exposure changes. Since these algorithms (but not the other four) are known not to be invariant to exposure. To test for colour correction as exposure changes, we 'train' the 3 algorithms with an exposure of 1 and then test given the exposures shown (see the exposure factors in the first column of Table 3). That is, in testing, we multiply the camera response RGBs and target XYZs by the exposure factors. For comparison, we also demonstrate **LS-RP-Lab** results as an exposure invariant solution in Table 3.

Table 3: Mean Delta E statistics in different exposure levels.

Meth. Expo.	LS-P	LS-P-Lab	NN	LS-RP-Lab
0.2	1.69	1.46	2.6	1.10
0.5	1.47	1.30	1.57	1.10
1	1.29	1.17	1.40	1.10
2	1.91	1.75	1.92	1.10
5	7.55	8.98	3.77	1.10

In Table 3, we report the mean cross validated CIELAB Delta E errors as exposure changes. Clearly, as the testing exposure departs from 1 (the exposure level where we trained the regressions and the Neural Network) there is markedly worse colour correction performance except for **LS-RP-Lab** which is an example of exposure invariant methods. The Neural Network at an exposure of 5, returns very poor colour correction performance.

Conclusion

In this article, we compared the performance of a recently introduced neural network model with 3 regression methods: Least Squares, Polynomial and Root-Polynomial. Although the results showed that the neural network works better than the Least Squares Regression, it delivers poorer colour correction than either the polynomial or root-polynomial regressions. Further, consistent with prior art research, we found that the regression methods could be improved by a search-based optimisation targeted toward minimizing Delta E error.

Another benefit of using the Least Squares or the Root-Polynomial Regression models is that they are intrinsically exposure invariant. When we solve for the regression with respect to one light level, the resulting regression matrix (that delivers colour correction) works equally well when the light level changes. In contrast, the Neural Network model does not generalise to different exposure conditions. Indeed, as the light level changes the NN we tested delivers poor colour correction performance.

References

[1] M. Anderson, R. Motta, S. Chandrasekar and M. Stokes, "Proposal for a Standard Default Color Space for The Internet—srgb" *Color and Imaging Conference,* Vol. 1996, No. 1, pp. 238-245 (1996, January).

[2] R. W. G. Hunt and M. R. Pointer, "Measuring colour", *John Wiley & Sons*, 4th ed. (2011).

[3] D. H. Foster, S. M. C. Nascimento and M. J. Foster, "Frequency of Metamerism in Natural Scenes" *Journal of the Optical Society of America A,* Vol. 23, pp. 2359-2372 (2006).

[4] H. Farid, "Blind Inverse Gamma Correction" *IEEE Transactions on Image Processing* 10.10 pp. 1428-1433 (2001).

[5] B. K. P. Horn, "Exact Reproduction of Colored Images" *Computer Vision, Graphics, and Image Processing*, Vol. 26, No. 2, pp. 135-167 (1984).

[6] R. Luther, "Aus dem Gebiet der Farbreizmetrik" *Zeitschrift fur Technische Physik*, Vol. 8, pp. 540-558 (1927).

[7] G. D. Finlayson and M. S. Drew, "Constrained Least-Squares Regression in Color Spaces" *Journal of Electronic Imaging*, Vol. 6(4), pp. 484–493 (1997).

[8] G. D. Finlayson, M. Mackiewicz and A. Hurlbert, "Color Correction Using Root-Polynomial Regression" *IEEE Transactions on Image Processing*, Vol. 24, No. 5 (2015).

[9] L. MacDonald and K. Mayer, "Camera Colour Correction using Neural Networks" *London Imaging Meeting*, pp. 54-57 (2021).

[10] R. A. Horn and C. R. Johnson, "Norms for vectors and matrices" *Matrix analysis*, pp. 313-386 (1990).

[11] R. Penrose, "A Generalized Inverse for Matrices", *Mathematical Proceedings of The Cambridge Philosophical Society,* Vol. 51, No. 3, pp. 406-413 (1955, July).

[12] G. Sharma, W. Wu and E. N. Dalal, "The CIEDE2000 Color-Difference Formula: Implementation Notes, Supplementary Test Data, and Mathematical Observations" *Color Research and Application* pp. 21–30 (2005).

[13] J. C. Lagarias, J. A. Reeds, M. H. Wright and P. E. Wright, "Convergence Properties of the Nelder-Mead Simplex Method in Low Dimensions" *SIAM Journal of Optimization*, 9(1) pp.112-147, (1998).

[14] K. Barnard, L. Martin, B. Funt and A. Coath, "A Data Set for Colour Research" *Color Research and Application*, Vol. 27, No. 3, pp. 147-151, (2002).

[15] J. L. Hintze and R. D. Nelson, "Violin Plots: A Box Plot-Density Trace Synergism" *The American Statistician*, *52*(2), pp. 181-184, (1998).

[16] S. D. Hordley and G. D. Finlayson, "Reevaluation of Color Constancy Algorithm Performance" *JOSA A*, *23*(5), pp. 1008-1020, (2006).