https://doi.org/10.2352/issn.2169-2629.2020.28.42

Reconstructing Spectra from RGB Images by Relative Error Least-Squares Regression

Yi-Tun Lin and Graham D. Finlayson
University of East Anglia, Norwich, United Kingdom

Abstract

Spectral reconstruction (SR) algorithms attempt to map RGB- to hyperspectral-images. Classically, simple pixel-based regression is used to solve for this SR mapping and more recently patch-based Deep Neural Networks (DNN) are considered (with a modest performance increment). For either method, the 'training' process typically minimizes a Mean-Squared-Error (MSE) loss. Curiously, in recent research, SR algorithms are evaluated and ranked based on a relative percentage error, so-called Mean-Relative-Absolute Error (MRAE), which behaves very differently from the MSE loss function. The most recent DNN approaches - perhaps unsurprisingly - directly optimize for this new MRAE error in training so as to match this new evaluation criteria.

In this paper, we show how we can also reformulate pixel-based regression methods so that they too optimize a relative spectral error. Our Relative Error Least-Squares (RELS) approach minimizes an error that is similar to MRAE. Experiments demonstrate that regression models based on RELS deliver better spectral recovery, with up to a 10% increment in mean performance and a 20% improvement in worst-case performance depending on the method.

1. Introduction

Hyperspectral imaging devices are widely used in practical applications, such as remote sensing [27, 7, 11, 24], medical imaging [29, 30], scene relighting [16] and digital art archiving [28]. These devices record high-resolution radiance spectra at every image pixel, providing very rich information of material compositions at pixel level. However, existing technologies often suffer from low spatial resolution, low light sensitivity and/or long integration time. Moreover, these devices are often expensive and bulky.

Alternatively, spectral reconstruction (SR) algorithms attempt to recover high-quality hyperspectral images from RGB images. Assuming SR algorithms work adequately well, they clearly provide a low-cost, high-speed and high-resolution hyperspectral imaging solution.

Early SR approaches are mostly *pixel-based*, aiming to find (statistically speaking) the 'best' spectral estimate from each individual RGB. Many of these methods are based on *regression*, including linear and polynomial regressions [13, 8, 17] and the leading sparse coding model [2]. These methods are characterized by simple formulations and generally optimized via closed-form solutions.

More recently, *patch-based* methods have come to the fore. Leading by Deep Neural Network (DNN) solutions (*e.g.* [4, 14, 6]), these methods argue that incorporating 'object-level' information is a key to better spectral recovery. While studies have shown that DNN algorithms do recover spectra more accurately than pixel-based regressions, the performance difference is arguably modest [2]. And, the cost of achieving this performance increment is an expensive training procedure which requires orders of magnitude more data than simple regressions.

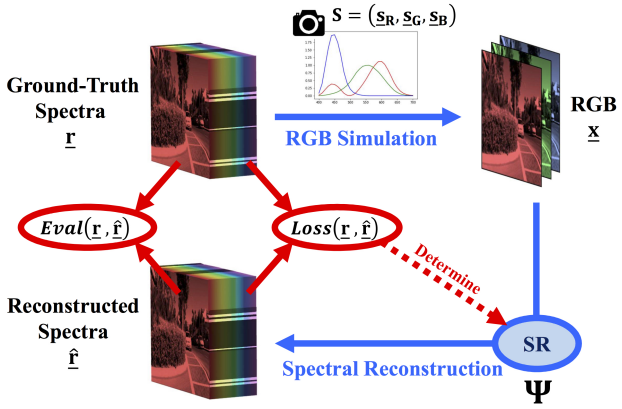

Figure 1. *The training and evaluation framework of spectral reconstruction*

The general workflow followed in SR research is illustrated in Figure 1. Ground-truth hyperspectral images (the top-left 3D image cube, denoted as $\underline{\mathbf{r}}$) are used to generate physically accurate RGB images (denoted as $\underline{\mathbf{x}}$) using a given camera's spectral sensitivities (here we show the CIEXYZ 1964 color matching functions [9] as an example). The SR algorithm models the map from RGB to spectra (*i.e. reconstructing* the spectra $\Psi(\underline{\mathbf{x}}) = \hat{\underline{\mathbf{r}}}$), and its parameters are calibrated such that a 'loss function' (denoted as $Loss(\underline{\mathbf{r}}, \hat{\underline{\mathbf{r}}})$) is minimized. Lastly, after the training process the trained models are evaluated by a given 'evaluation metric' (denoted as $Eval(\underline{\mathbf{r}}, \hat{\underline{\mathbf{r}}})$). This workflow constitutes the *best case* operating conditions for SR, *e.g.* if we recover spectra from real camera's images, this will surely negatively impact on performance. We adopt the shown workflow because it has been used in many published works, and this allows us to benchmark our method against the prior art.

Intuitively, the loss and evaluation functions should be the same: metaphorically, it wouldn't make sense to optimize the design of a car for better speed and then gauge its performance by measuring its fuel efficiency. Likewise, SR algorithms optimized for one metric and yet evaluated by another will surely lead to sub-optimal performance for the evaluation metric at hand.

We see in recent benchmarks - leading by the biggest *NTIRE Spectral Reconstruction Challenges* in 2018 [4] and 2020 [5] - models are evaluated and ranked according to a Mean-Relative-Absolute Error (MRAE). In contrary, all (to our knowledge) pixel-based regression models and most patch-based DNNs (*e.g.* [19, 15, 12, 6]) are trained to minimize the sum of squared error (or equivalently the Mean-Squared-Error (MSE) loss function). While some recent patch-based DNN models have sought not to minimize MSE but rather the expected MRAE, see [23, 4], pixel-based regression methods have not been upgraded to work with the MRAE error.

In this paper, we modify pixel-based spectral reconstruction

algorithms - including, the regression-based models [13, 8, 17] and the leading sparse coding model [2] - to optimize for a percentage spectral error, namely the *Relative Error Least-Squares* (RELS). In RELS we optimize for the L2 relative error instead of (yet similar to) MRAE, simply because it is an easier function to optimize (with a closed-form global optimum).

The rest of the paper is organized as follows. In section 2 we review the spectral reconstruction (SR) problem. In section 3 we detail the proposed Relative Error Least-Squares optimization for SR. Experiments are reported in section 4. This paper concludes in section 5.

2. Background

An RGB camera captures the scene with 3 numbers per pixel, referring to the photo-responses recorded by the respective R, G and B sensors. Each sensor is described by a *spectral sensitivity function*, that is the weighted sensitivities over different wavelengths. In this paper we assume the visible spectrum runs from 400 to 700 nanometers (nm), and the hyperspectral measurements sample the visible spectrum with 10-nm intervals (therefore the hyperspectral images have 31 spectral channels).

Recently many SR algorithms are trained on *images*. As opposed to the research based on 'point measurements' where matched RGB and hyperspectral data can be easily captured, in image-based SR we often *simulate* ground-truth RGBs from hyperspectral images given an explicit formulation of the camera's processing pipeline [18]. Denote $\underline{\mathbf{r}}$ as the 31-dimensional spectral measurement, $\mathbf{S}$ as the 31×3 sensitivity matrix whose columns are the spectral sensitivity functions, the general form of the RGB camera response $\underline{\mathbf{x}} = (R,G,B)^{\mathsf{T}}$ is written as [13]:

$$\Gamma\left(\mathbf{S}^{\mathsf{T}}\underline{\mathbf{r}} + \underline{\varepsilon}\right) = \underline{\mathbf{x}}\,, \tag{1}$$

where Γ is a given non-linear rendering function (accounting for mapping measured colors to display RGB, white balance, tone-mapping, *etc.* [18]), and $\underline{\varepsilon}$ is a 3-component error vector referring to the noise occurred during the RGB capturing. We remark that the formulation in Equation (1) does not include any spatial processing applied in many camera's processing pipelines.

In this paper we consider a special case where Γ is *identity transformation* and $\underline{\varepsilon} = \underline{\mathbf{0}}$ as the noiseless simulation of the RGB camera's linear photo-response (*i.e.* the 'linear RGBs' or 'raw images'). We note that this setting is one of the standard protocols used in previous research.

2.1. Pixel-based Regression Models

Pixel-based regressions formulate the RGB-to-spectrum mapping with various assumptions on the nature of the map. Hereafter we denote an SR algorithm as the function $\Psi:\ \mathbb{R}^3 \mapsto \mathbb{R}^{31}$, where the spectral estimates of $\underline{\mathbf{r}}$ are given by Ψ on input of RGB camera responses, such that $\Psi(\underline{\mathbf{x}}) \approx \underline{\mathbf{r}}$.

The simplest Linear Regression (LR) [13] assumes a linear map from RGBs to spectra:

$$\Psi(\underline{\mathbf{x}}) = \mathbf{M}\underline{\mathbf{x}}\,. \tag{2}$$

Here, the matrix $\mathbf{M}$ is called the *regression matrix* which contains all the model parameters. Recall that the spectra in this paper are modeled as 31-component vectors, so $\mathbf{M}$ is a 31×3 matrix.

Then, the Polynomial Regression (PR) [8] and Root-Polynomial Regression (RPR) [17] serve as simple *non-linear* solutions. These models calculate a series of polynomial terms from each RGB (up to a given order). Examples of the 2^{nd}, 3^{rd} and 4^{th} order PR and RPR expansions are given in Table 1. Denote $\boldsymbol{\varphi}:\mathbb{R}^3\mapsto\mathbb{R}^p$ as the given polynomial expansion of PR or RPR (p is the number of polynomial terms), the spectral reconstruction is modeled by

$$\Psi(\underline{\mathbf{x}}) = \mathbf{M}\boldsymbol{\varphi}(\underline{\mathbf{x}})\,. \tag{3}$$

As an example, the 2^{nd} order root-polynomial expansion has 6 terms (refer to Table 1 in the second row on the right). Given this representation we are to find a 31×6 regression matrix $\mathbf{M}$ that maps the polynomial vectors to spectral estimates.

Yet another regression approach is the Adjusted Anchored Neighborhood Regression (abbreviated as A+) [2]. Same as LR, A+ assumes a linear map from RGB to spectra. But, instead of modeling a universal mapping for all data as LR, A+ seeks to assign different mappings locally in the 'neighborhoods' (that is to say A+ is a *sparse coding* algorithm). In brief, the A+ algorithm partitions the spectral and RGB space into overlapping neighborhoods, and for the RGBs *belong* to the same neighborhood they share the same mapping function (*i.e.* regression matrix):

$$\mathit{neighborhood}(\underline{\mathbf{x}}) = i \;\Rightarrow\; \Psi_i(\underline{\mathbf{x}}) = \mathbf{M}_i\underline{\mathbf{x}}\,. \tag{4}$$

For details about the method, see [2].

Regarding Equation (2), (3) and (4), we henceforth uniformly use the $\boldsymbol{\varphi}$ notation (polynomial expansion): since, LR and A+ effectively use the first order polynomial expansion $\boldsymbol{\varphi}(\underline{\mathbf{x}}) = \underline{\mathbf{x}}$.

2.2. Ordinary Least-Squares (OLS) Minimization

Ordinary Least-Squares (OLS) is a standard way to solve for the regression models in section 2.1, which seeks to minimize the sum of squared errors (MSE loss function) between ground-truth training spectra and predictions:

$$\min_{\mathbf{M}}\left(\sum_{j=1}^{N}||\underline{\mathbf{r}}_j - \mathbf{M}\boldsymbol{\varphi}(\underline{\mathbf{x}}_j)||_2^2\right), \tag{5}$$

where $\underline{\mathbf{r}}_j$'s and $\underline{\mathbf{x}}_j$'s are matched spectrum and RGB pairs in the training set (j indexes individual data pairs), and the number of training pairs is N.

Denoting Φ and $\mathbf{R}$ as respectively $N\times p$ and $N\times 31$ *data matrices* (the rows are respectively matched $\boldsymbol{\varphi}(\underline{\mathbf{x}}_j)$'s and $\underline{\mathbf{r}}_j$'s), Equation (5) can be written as:

$$\min_{\mathbf{M}}||\mathbf{R} - \Phi\mathbf{M}^{\mathsf{T}}||_F^2\,, \tag{6}$$

where $||\cdot||_F$ indicates sum of squares (equivalently, the Frobenius norm). Then, the regression matrix $\mathbf{M}$ is solved in closed form (using the Moore-Penrose inverse [20]) as:

$$\mathbf{M} = \mathbf{R}^{\mathsf{T}}\Phi(\Phi^{\mathsf{T}}\Phi)^{-1}\,. \tag{7}$$

Least-squares, while simple, can potentially over-fit the trained model to the training data [13]. An alternative 'robust' least-squares minimization is given by *Tikhonov regularization* [25, 13]:

$$\min_{\mathbf{M}}\left(||\mathbf{R} - \Phi\mathbf{M}^{\mathsf{T}}||_F^2 + \gamma||\mathbf{M}||_F^2\right), \tag{8}$$

and solved in closed-form [13]:

$$\mathbf{M} = \mathbf{R}^{\mathsf{T}}\Phi(\Phi^{\mathsf{T}}\Phi + \gamma\mathbf{I})^{-1}\,, \tag{9}$$

where $\mathbf{I}$ is the $p\times p$ identity matrix.

Above, γ is a user defined parameter that controls the *penalty* term: $||\mathbf{M}||_F^2$. This term effectively prevents drastic

Table 1. Polynomial expansions for PR and RPR models (the 1-st order PR and RPR are equivalent to LR)

Order	**Polynomials (PR)** [8]	**Root-polynomials (RPR)** [17]
1	R,G,B	R,G,B
2	$R,G,B,R^2,G^2,B^2,RG,GB,BR$	$R,G,B,\sqrt{RG},\sqrt{GB},\sqrt{BR}$
3	$R,G,B,R^2,G^2,B^2,RG,GB,BR,$ $R^3,G^3,B^3,RG^2,GB^2,BR^2,R^2G,G^2B,B^2R,RGB$	$R,G,B,\sqrt{RG},\sqrt{GB},\sqrt{BR},$ $\sqrt[3]{RG^2},\sqrt[3]{GB^2},\sqrt[3]{BR^2},\sqrt[3]{R^2G},\sqrt[3]{G^2B},\sqrt[3]{B^2R},\sqrt[3]{RGB}$
4	$R,G,B,R^2,G^2,B^2,RG,GB,BR,$ $R^3,G^3,B^3,RG^2,GB^2,BR^2,R^2G,G^2B,B^2R,RGB,$ $R^4,G^4,B^4,R^3G,R^3B,G^3R,G^3B,B^3R,B^3G,$ $R^2G^2,G^2B^2,R^2B^2,R^2GB,G^2RB,B^2RG$	$R,G,B,\sqrt{RG},\sqrt{GB},\sqrt{BR},$ $\sqrt[3]{RG^2},\sqrt[3]{GB^2},\sqrt[3]{BR^2},\sqrt[3]{R^2G},\sqrt[3]{G^2B},\sqrt[3]{B^2R},\sqrt[3]{RGB},$ $\sqrt[4]{R^3G},\sqrt[4]{R^3B},\sqrt[4]{G^3R},\sqrt[4]{G^3B},\sqrt[4]{B^3R},\sqrt[4]{B^3G},$ $\sqrt[4]{R^2GB},\sqrt[4]{G^2RB},\sqrt[4]{B^2RG}$

changes in output (the spectra) on small perturbation of the input (the RGBs), such that the model can be more robust against noise and over-fitting. See [25] for more details.

Determining a *suitable* value of γ is, of course, crucial for the model performance. However, there is no closed-form method to determine its best value. Rather, an empirical search is carried out. In the literature, cross validation [10] (which we use in our experiments) and L-curve method [10, 22] are two main ways to determine γ.

3. Relative Error Least-Squares (RELS)

Intuitively, the selection of evaluation metric should reflect the nature of the measuring targets. However, research proposed that MSE should **not** be used to evaluate spectral recovery [3, 2, 4]. These works argued that the Root-Mean-Square Error (RMSE) (the root of MSE) tends to penalize bright pixels more than dark pixels. We remark that this bias should also affect the MSE minimization (*e.g.* OLS) in the training stage, where spectra that are less well exposed are not regarded with the same importance as the bright spectra in an MSE minimization.

3.1. Measuring Relative Error

A natural metric to evaluate methods trained via MSE minimization is Root-Mean-Square Error (RMSE), defined as:

$$RMSE(\underline{\mathbf{r}},\hat{\underline{\mathbf{r}}}) = \sqrt{\frac{1}{31}\left|\left|\underline{\mathbf{r}}-\hat{\underline{\mathbf{r}}}\right|\right|_2^2} = \sqrt{\frac{1}{31}\sum_{k=1}^{31}(r_k-\hat{r}_k)^2}\ , \quad (10)$$

where $\underline{\mathbf{r}}$ is the ground-truth spectrum, $\hat{\underline{\mathbf{r}}} = \Psi(\underline{\mathbf{x}})$ is the recovered spectrum, and r_k is the value of the spectrum in the k^{th} spectral channel.

In contrary, the recent Mean-Relative-Absolute Error (MRAE) [4, 5] is calculated as:

$$MRAE(\underline{\mathbf{r}},\hat{\underline{\mathbf{r}}}) = \frac{1}{31}\left|\left|\frac{\underline{\mathbf{r}}-\hat{\underline{\mathbf{r}}}}{\underline{\mathbf{r}}}\right|\right|_1 = \frac{1}{31}\sum_{k=1}^{31}\left|\frac{r_k-\hat{r}_k}{r_k}\right|. \quad (11)$$

Notice that the division is done 'channel-by-channel'.

In Figure 2 we show an actual example of A+ spectral recovery [2] to help clarify the motivation of why MRAE is preferable. In the left and right panels of Figure 2 we show two sets of spectra, where red curves are the ground-truths and blue curves indicate the recovered spectra. The spectra on the right are *identical* to those on the left, save that they are twice as bright. In the text boxes (foot of Figure 2) the RMSE and MRAE errors are shown, indicating that the RMSE scales with the brightness whereas MRAE does not.

The MRAE in this example seems to better capture the apparent fact that the recovery is equally good in both figures. In fact, the situation as Figure 2 happens very often: the same

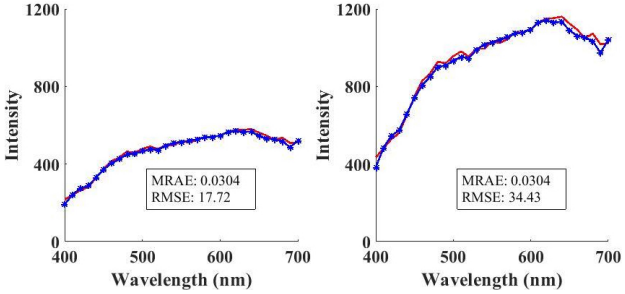

Figure 2. *Compare the reconstructed spectra by A+ algorithm [2] (blue curves) with the ground-truth radiance spectra (red curves). RMSE evaluation scales with the brightness level, while MRAE do not.*

physical object can be represented by spectra scaled with different brightness factor when the exposure settings between scenes and/or the quantity of light across a scene vary.

3.2. Minimizing Relative Error

From an optimization point of view, we could try to minimize MRAE directly:

$$\min_{\Psi}\left(\sum_{j=1}^{N}\left|\left|\frac{\underline{\mathbf{r}}_j-\Psi(\underline{\mathbf{x}}_j)}{\underline{\mathbf{r}}_j}\right|\right|_1\right). \quad (12)$$

This minimization refers to the 'Least Absolute Deviation Regression' which can be solved using linear programming [1]. However, the number of constraints in this linear program is large (meaning that the minimization itself is non-trivial), and also it cannot be solved in closed form. Hence, we in turn propose to minimize an *allied* L2 relative error:

$$\min_{\Psi}\left(\sum_{j=1}^{N}\left|\left|\frac{\underline{\mathbf{r}}_j-\Psi(\underline{\mathbf{x}}_j)}{\underline{\mathbf{r}}_j}\right|\right|_2^2\right), \quad (13)$$

namely, the Relative Error Least-Squares (RELS) optimization.

We carry out separate minimization for each of the 31 spectral bands. Consider the regression models $\Psi(\underline{\mathbf{x}}_j) = \mathbf{M}\varphi(\underline{\mathbf{x}}_j)$, we split the regression matrix into $\mathbf{M} = (\underline{\mathbf{m}}_1,\ \underline{\mathbf{m}}_2,\ \cdots,\ \underline{\mathbf{m}}_{31})^{\mathsf{T}}$. Now, for the spectral channel k, the following expression is to be minimized:

$$\min_{\underline{\mathbf{m}}_k}\left(\sum_{j=1}^{N}\left(\frac{r_{j,k}-\underline{\mathbf{m}}_k^{\mathsf{T}}\varphi(\underline{\mathbf{x}}_j)}{r_{j,k}}\right)^2\right), \quad (14)$$

or, equivalently:

$$\min_{\underline{\mathbf{m}}_k}\left(\sum_{j=1}^{N}\left(1-\underline{\mathbf{m}}_k^{\mathsf{T}}\frac{\varphi(\underline{\mathbf{x}}_j)}{r_{j,k}}\right)^2\right). \quad (15)$$

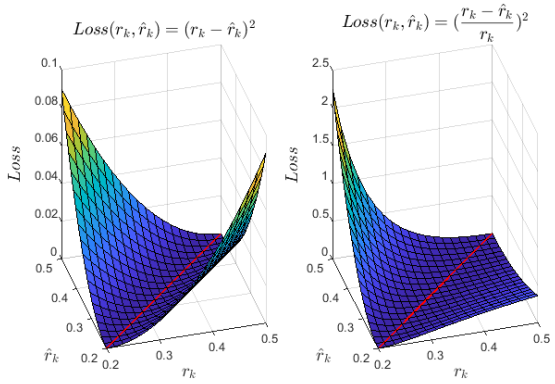

Figure 3. *Visualizing the loss functions of OLS (left) and RELS (right). The diagonal red lines indicate the 'zero-error' line (i.e. $\hat{r}_k = r_k$).*

Next, just like Equation (6) we denote Φ as the $N \times p$ data matrix of $\varphi(\underline{\mathbf{x}}_j)$'s and vectorize the expression into:

$$\min_{\underline{\mathbf{m}}_k} \left(||\underline{\mathbf{1}} - \mathbf{D}_k \Phi \underline{\mathbf{m}}_k||_2^2 \right) . \tag{16}$$

Here, $\underline{\mathbf{1}}$ is an N-dimensional vector with all entries equal to 1, and $\mathbf{D}_k$ is a diagonal matrix of dimension $N \times N$ whose components are the reciprocals of the k-channel values of all ground-truth training spectra (*i.e.* the $r_{j,k}$ in Equation (15)):

$$\mathbf{D}_k = \begin{pmatrix} 1/r_{1,k} & 0 & \cdots & 0 \\ 0 & 1/r_{2,k} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1/r_{N,k} \end{pmatrix} . \tag{17}$$

Finally, the closed form solution of RELS (by channel) is derived again by Moore-Penrose inverse [20, 26]:

$$\underline{\mathbf{m}}_k = \left((\mathbf{D}_k \Phi)^\mathsf{T} (\mathbf{D}_k \Phi) \right)^{-1} (\mathbf{D}_k \Phi)^\mathsf{T} \underline{\mathbf{1}} . \tag{18}$$

As for the Tikhonov-type regularization (as discussed in Equation (8) and (9)), the RELS version is analogously:

$$\underline{\mathbf{m}}_k = \left((\mathbf{D}_k \Phi)^\mathsf{T} (\mathbf{D}_k \Phi) + \gamma_k \mathbf{I} \right)^{-1} (\mathbf{D}_k \Phi)^\mathsf{T} \underline{\mathbf{1}} , \tag{19}$$

where γ_k is a per-channel user-defined penalty parameter.

Note that, here the γ_k is chosen to optimize the penalty for each spectral channel independently. For fair comparison, in the experiments we report later in this paper we also operate OLS channel-by-channel (with separate penalty parameter for each channel).

In Figure 3 we visualize the loss functions of OLS (left figure) and RELS minimization (right figure) for an arbitrary k^{th} spectral channel within a selected window: $r_k, \hat{r}_k \in [0.2, 0.5]$. The dotted diagonal red line in both diagrams is the locus of zero error. It is evident that these two loss functions are quite different from one another - regarding how they evolve away from the zero-error locus - and, this said, it is likely that minimizing OLS and RELS loss will generate different spectral estimates.

4. Experiments

We evaluated four pixel-based regression methods: Linear Regression (LR) [13], 6^{th}-order Polynomial Regression (PR6) [8], 6^{th}-order Root-Polynomial Regression (RPR6) [17] and the Adjusted Anchored Neighborhood Regression (A+) [2], optimized via both the original OLS and our RELS minimizations.

4.1. RGB and Hyperspectral Images

The ground-truth spectral data is from the ICVL hyperspectral database [3], which includes 201 hyperspectral and RGB images of both indoor and outdoor scenes. The dimension of these images is of $1300 \times 1392 \times 31$, where the '31' spectral channels refer to the discrete spectral measurements with 10 nm sampling intervals within the 400-to-700 nm visible range. The RGB images were created by numerical integration using the CIE 1964 color matching functions [9] as the camera spectral sensitivities and following the 'linear RGB' setting (refer to the discussion below Equation (1)).

4.2. Cross Validation

We followed a *cross validation* procedure to train and evaluate the models. In conventional K-fold cross validation, the data is split into K folds: train on $K-1$ folds and test on the remaining fold, and the error statistics are averaged over the K experiments [21]. In our experiment, apart from training and testing we are also to select proper penalty parameters γ_k for the regularized regression.

We split our data set into four equalized folds - I, II, III and IV - and followed a four-experiment setting:

- Train: I + II , Penalty Term: III, Test: IV
- Train: I + II , Penalty Term: IV, Test: III
- Train: III + IV, Penalty Term: I , Test: II
- Train: III + IV, Penalty Term: II , Test: I .

For example, in the first experiment shown above we combine fold I and II as our training data. We use fold III to 'tune' the penalty terms, that is to try different γ_k's (grid search) and choose the one that minimizes the loss introduced by fold III data. Then, given a tuned penalty term we test the models on fold IV (test set). This training regime ensures that all folds are used for testing once. Finally, for each test set we calculated recovery performance statistics (in MRAE) and then averaged them over the four testing folds.

5. Results and Discussion

In Table 2 we present various results. First, the **Mean** (left table) and **99 Percentile** MRAE (right table) were calculated 'per image' and the shown numbers are the averaged results of the test-set images (cross validated). Second, under the **All**, **Bright** and **Dim** columns are respectively the results averaging over all the pixels, the top 50% brightest pixels and the 50% dimmest pixels in each test image (here we define 'brightness' as the L2 norm of the ground-truth spectra). Finally, referring to [17], PR6 performs differently under different testing exposure settings (while LR, RPR6 and A+ performs *exactly the same*), hence we specifically tested PR6 under original, half and double testing exposures: we uniformly scale all the testing ground-truth images by a factor $\xi = 1$, 0.5 and 2. For more information, see [17].

We see that for the overall performance (the '**All**' columns), RELS significantly improves for all algorithms in terms of mean and the worst-case (*i.e.* the 99 percentile) MRAE. More specifically for the leading sparse coding model A+ [2], our RELS minimization provides a 7.2% boost in mean MRAE performance, and for the simplest LR method [13], RELS improves its worst-case MRAE by a remarkable 19.1%. These results show that choosing a loss function matched with the evaluation metric can be very crucial to the performance of pixel-based regression models.

Now, we examine the brightness dependency in the MRAE performance (the '**Bright**' and '**Dim**' columns). Using the orig-

Table 2. The averaged *per-image* mean (left table) and 99 percentile spectral recovery errors (right table). 'All', 'Bright' and 'Dim' refer to all pixels, bright pixels (top 50% of the image) and dim pixels (bottom 50% of the image), respectively. Note that the PR6 model is tested under three exposure settings (*i.e.* $\xi = 1,\ 0.5$ and 2). Other models are tested with the original test-set data (without the exposure manipulation).

	Mean MRAE						99 Percentile MRAE					
	OLS			RELS			OLS			RELS		
	All	Bright	Dim	All	Bright	Dim	All	Bright	Dim	All	Bright	Dim
LR	0.0609	0.0453	0.0765	0.0563	0.0564	0.0561	0.1741	0.1354	0.1612	0.1409	0.1341	0.1225
RPR6	0.0440	0.0378	0.0502	0.0419	0.0430	0.0407	0.1537	0.1211	0.1459	0.1294	0.1157	0.1208
A+	0.0388	0.0341	0.0435	0.0360	0.0372	0.0349	0.1517	0.1188	0.1440	0.1350	0.1272	0.1178
PR6 ($\xi = 1$)	0.0195	0.0208	0.0182	0.0188	0.0208	0.0168	0.0710	0.0679	0.0572	0.0703	0.0683	0.0554
PR6 ($\xi = 0.5$)	0.0964	0.0969	0.0959	0.1099	0.1041	0.1158	0.1669	0.1629	0.1412	0.1881	0.1794	0.1640
PR6 ($\xi = 2$)	0.1306	0.1856	0.0757	0.2129	0.3469	0.0788	1.2084	1.6946	0.1517	2.4516	3.4347	0.1838

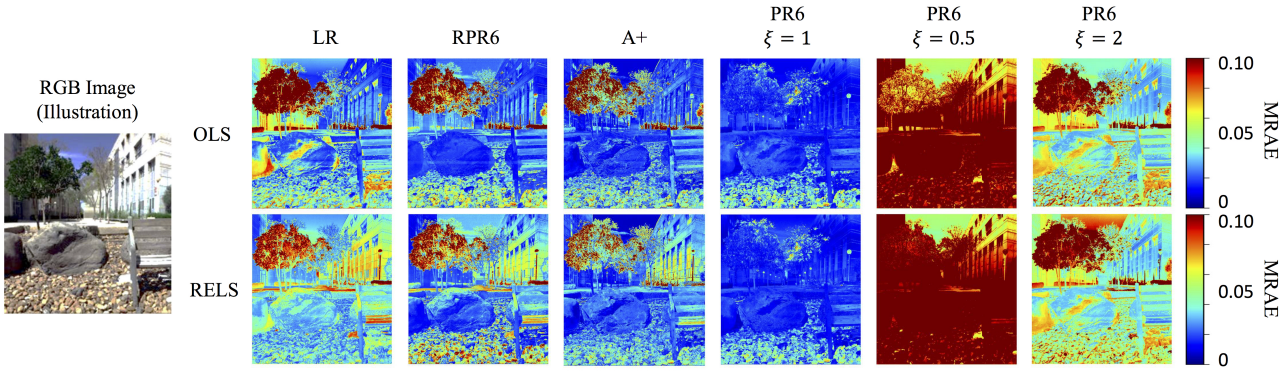

Figure 4. *MRAE error maps for all tested models. Top row: OLS-based models. Bottom row: RELS-based models.*

inal OLS minimization, LR, RPR6 and A+ show clear bias towards bright pixels with substantial discrepancies. In contrary, our RELS minimization effectively *balances* the performance between bright and dim pixels (while providing better overall performance). For the PR6 model under original exposure $\xi = 1$ there is not clear brightness dependency for both OLS and RELS, while the increment in overall performance by using RELS results from the improvement in the performance of dim pixels.

Finally, in agreement with the results in [17], PR6 method deteriorates under varying testing exposure. This teaches that only the brightness-invariant models such as LR, RPR and A+ are suitable for practical use - where scene brightness can vary both spatially and temporally.

In Figure 4 we show the MRAE error maps. We can clearly see the trade-offs in the images: the OLS models recover the spectra of dark tree leaves very badly, but for the bright walls it performs rather well; in contrary, RELS narrows the MRAE difference between the tree leaves and bright walls, claiming a better overall recovery performance.

6. Conclusion

Reconstructing radiance spectra from RGB camera responses is often formulated as a machine learning problem. Most of the approaches by-default minimize the sum of squared errors (a Mean-Squared-Error (MSE) loss function). However, recent works on large-scale benchmark challenge [4, 5] instead rank models using an MRAE metric - a percentage deviation with respect to the ground-truth spectrum. This 'mismatch' between training loss and evaluation metric has only been addressed by some recent Deep Neural Network (DNN) solutions [23, 4] where MRAE is minimized directly.

In this paper we seek to solve this issue for the pixel-based regression methods. We formulate a closed-form Relative Error Least-Squares (RELS) optimization approach which minimizes a relative error loss function that behaves more like MRAE. Our experimental results show that RELS significantly improves the MRAE performance of dim pixels that leads to an overall performance boost for all considered models.

References

[1] *Least Absolute Deviation Regression*, pages 299–302. Springer New York, New York, NY, 2008.

[2] J. Aeschbacher, J. Wu, and R. Timofte. In defense of shallow learned spectral reconstruction from RGB images. In *Proceedings of the International Conference on Computer Vision*, pages 471–479. IEEE, 2017.

[3] B. Arad and O. Ben-Shahar. Sparse recovery of hyperspectral signal from natural RGB images. In *Proceedings of the European Conference on Computer Vision*, pages 19–34. Springer, 2016.

[4] B. Arad, O. Ben-Shahar, R. Timofte, et al. NTIRE 2018 challenge on spectral reconstruction from RGB images. In *Proceedings of the Conference on Computer Vision and Pattern Recognition Workshops*, pages 929–938. IEEE, 2018.

[5] B. Arad, R. Timofte, O. Ben-Shahar, Y. Lin, G.D. Finlayson, et al. NTIRE 2020 challenge on spectral reconstruction from an RGB image. In *Proceedings of the Conference on Computer Vision and Pattern Recognition Workshops*. IEEE, June 2020.

[6] P.V. Arun, K.M. Buddhiraju, A. Porwal, and J. Chanussot. CNN based spectral super-resolution of remote sensing images. *Signal Processing*, 169:107394, 2020.

[7] Y. Chen, X. Zhao, and X. Jia. Spectral–spatial classification of hyperspectral data based on deep belief network. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 8(6):2381–2392, 2015.

[8] D.R. Connah and J.Y. Hardeberg. Spectral recovery using polynomial models. In *Color Imaging X: Processing, Hardcopy, and*

Applications, volume 5667, pages 65–75. International Society for Optics and Photonics, 2005.

[9] Commission Internationale de l'Eclairage. CIE proceedings (1964) Vienna session, committee report E-1.4. 1. 1964.

[10] N.P. Galatsanos and A.K. Katsaggelos. Methods for choosing the regularization parameter and estimating the noise variance in image restoration and their relation. *IEEE Transactions on Image Processing*, 1(3):322–336, 1992.

[11] P. Ghamisi, M. Dalla Mura, and J.A. Benediktsson. A survey on spectral–spatial classification techniques based on attribute profiles. *IEEE Transactions on Geoscience and Remote Sensing*, 53(5):2335–2353, 2014.

[12] X. Han, B. Shi, and Y. Zheng. Residual HSRCNN: Residual hyperspectral reconstruction CNN from an RGB image. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 2664–2669. IEEE, 2018.

[13] V. Heikkinen, R. Lenz, T. Jetsu, J. Parkkinen, M. Hauta-Kasari, and T. Jääskeläinen. Evaluation and unification of some methods for estimating reflectance spectra from RGB images. *Journal of the Optical Society of America A*, 25(10):2444–2458, 2008.

[14] B. Kaya, Y.B. Can, and R. Timofte. Towards spectral estimation from a single RGB image in the wild. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 3546–3555. IEEE, 2019.

[15] S. Koundinya, H. Sharma, M. Sharma, A. Upadhyay, R. Manekar, R. Mukhopadhyay, A. Karmakar, and S. Chaudhury. 2D-3D CNN based architectures for spectral reconstruction from RGB images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 844–851, 2018.

[16] A. Lam and I. Sato. Spectral modeling and relighting of reflective-fluorescent scenes. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 1452–1459. IEEE, 2013.

[17] Y. Lin and G.D. Finlayson. Exposure invariance in spectral reconstruction from RGB images. In *Proceedings of the Color and Imaging Conference*, volume 2019, pages 284–289. Society for Imaging Science and Technology, 2019.

[18] J. Nakamura. *Image sensors and signal processing for digital still cameras*. CRC press, 2017.

[19] S. Nie, L. Gu, Y. Zheng, A. Lam, N. Ono, and I. Sato. Deeply learned filter response functions for hyperspectral reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4767–4776, 2018.

[20] R. Penrose. A generalized inverse for matrices. In *Mathematical proceedings of the Cambridge philosophical society*, volume 51, pages 406–413. Cambridge University Press, 1955.

[21] P. Refaeilzadeh, L. Tang, and H. Liu. *Encyclopedia of Database Systems – Cross Validation*, pages 532–538. Springer US, Boston, MA, 2009.

[22] T. Regińska. A regularization parameter in discrete ill-posed problems. *SIAM Journal on Scientific Computing*, 17(3):740–749, 1996.

[23] Z. Shi, C. Chen, Z. Xiong, D. Liu, and F. Wu. HSCNN+: Advanced CNN-based hyperspectral recovery from RGB images. In *Proceedings of the Conference on Computer Vision and Pattern Recognition Workshops*, pages 939–947. IEEE, 2018.

[24] C Tao, H. Pan, Y. Li, and Z. Zou. Unsupervised spectral–spatial feature learning with stacked sparse autoencoder for hyperspectral imagery classification. *IEEE Geoscience and Remote Sensing Letters*, 12(12):2438–2442, 2015.

[25] A.N. Tikhonov, A.V. Goncharsky, V.V. Stepanov, and A.G. Yagola. *Numerical Methods for the Solution of Ill-posed Problems*, volume 328. Springer Science & Business Media, 2013.

[26] C. Tofallis. Least squares percentage regression. *Journal of Modern Applied Statistical Methods*, 2009.

[27] M.A. Veganzones, G. Tochon, M. Dalla-Mura, A.J. Plaza, and J. Chanussot. Hyperspectral image segmentation using a new spectral unmixing-based binary partition tree representation. *IEEE Transactions on Image Processing*, 23(8):3574–3589, 2014.

[28] P. Xu, H. Xu, C. Diao, and Z. Ye. Self-training-based spectral image reconstruction for art paintings with multispectral imaging. *Applied Optics*, 56(30):8461–8470, 2017.

[29] Y. Zhang, X. Mou, G. Wang, and H. Yu. Tensor-based dictionary learning for spectral CT reconstruction. *IEEE Transactions on Medical Imaging*, 36(1):142–154, 2016.

[30] Y. Zhang, Y. Xi, Q. Yang, W. Cong, J. Zhou, and G. Wang. Spectral CT reconstruction with image sparsity and spectral mean. *IEEE Transactions on Computational Imaging*, 2(4):510–523, 2016.

Author Biography

Yi-Tun (Ethan) Lin is a Ph.D. student in the Colour & Imaging Lab, School of Computing Sciences, University of East Anglia, UK. He received a joint M.Sc. degree in Colour Science in 2018, from University Jean Monnet (France), University of Granada (Spain) and University of Eastern Finland (Finland), and a B.Sc. degree in Physics in 2016, from National Taiwan University, Taiwan. His research interest is physics and machine learning-based spectral reconstruction.

Graham Finlayson is a Professor of Computer Science at the University of East Anglia (UEA) where he leads the Colour & Imaging Lab. Professor Finlayson is interested in 'computing how we see' and his research spans computer science (algorithms), engineering (embedded systems) and psychophysics (visual perception). Significantly, some of Graham's research is implemented and used in commercial products including photo processing software, dedicated image processing hardware and in embedded camera software.