Finding representatives in a large dataset of spectral reflectances

Silvio Borer and Sabine Süsstrunk
Laboratory of Computational Neuroscience and
Image and Visual Representation Group
Ecole Polytechnique Fédérale de Lausanne (EPFL), Switzerland

Abstract

We propose a new method to construct representative spectra from a large database of spectral reflectances. The key is the optimisation of a Support Vector type functional. The representatives are constructed such that they sit at positions of high density in the set of spectra. At the same time they are constructed to be as orthogonal as possible. The representatives are expressible as a linear combination of data samples with positive coefficients. Therefore, they are positive and physically realisable. We show the differences of these representatives to representatives found with well-known methods like principal component analysis and k–means clustering.

Keywords: reflectance spectra, clustering, Support Vector algorithm

1. Introduction

The reflectance spectra of objects are not uniformly distributed. Certain colours appear in small variations very often, whereas others are rarely seen. But which colours or spectra are the typical ones, and how can we represent them ? Using a database of reflectance spectra we propose a method to find representatives. Other researchers have already investigated similar questions, but with far smaller databases [6, 11, 3, 7, 4]. As database we use the SOCS database [1], a collection of about $50'000$ reflectance spectra with wavelength in the interval between 400 nm and 700 nm at a 10 nm stepsize. The samples in the database are processed as follows: First, we multiplied the spectra with a $D65$ daylight illuminant. The resulting spectra were normalised in the L_2-norm. Therefore, the processed spectra sit on a hypersphere. The normalised data is used because we are primarily interested in chroma. From our analysis we cannot draw conclusions about colour intensity. Figure 1 shows a boxplot of the preprocessed samples in the database. The box extends from the first to the third quartile. The dividing line of the box is the median. The whiskers extend to the extreme values.

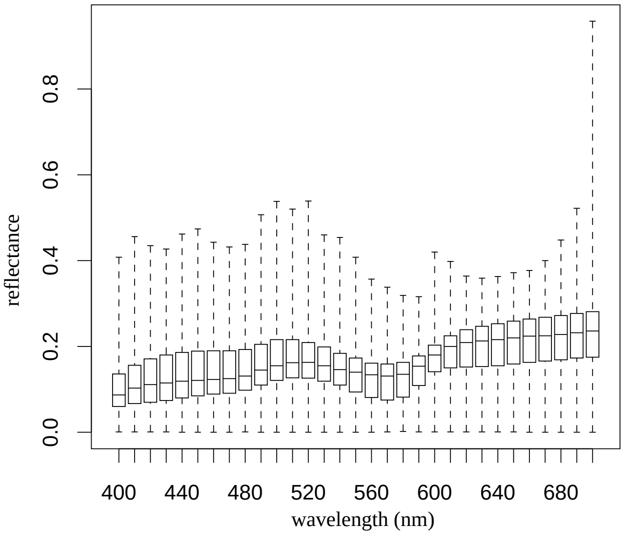

Figure 1: A boxplot of the spectral data. The box shows the median, the first and third quartile of the normalised reflectance spectra, see text. The whiskers extend to the extreme values.

2. Orthogonal decomposition

A standard method to construct representative vectors is to compute the principal components. Figure 2 shows the mean (m) together with the first three principal components (1-3). A disadvantage of the principal components is that they are not representative in the sense that they can not be interpreted as prototypes or typical spectra. We propose a method to find representative spectra, which can be interpreted as prototypes of the dataset. The rest of the paper is organised as follows. In the next section a new algorithm is proposed to find representative vectors or prototypes of spectra. As an illustration, this algorithm is first applied to the solution of a toy problem. Then it is applied to find representative vectors of the SOCS database. In section 4 we project the data and the representative vectors in a two–dimensional subspace to illustrate important features of our solution. Finally, the question of how many representatives are needed is addressed.

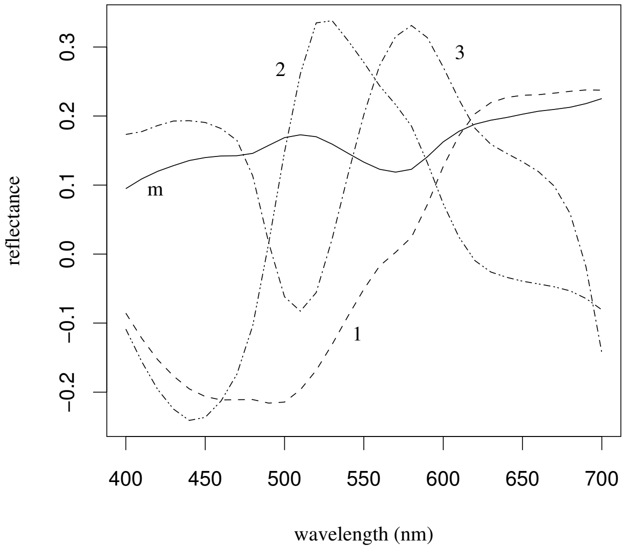

Figure 2: *The mean together with the first three principal components.*

3. Finding representatives

Let us explain how we find representative vectors. Figure 3 shows schematically the situation: The small circles show the preprocessed data samples. The data samples are normalised, thus they sit on a high–dimensional sphere. The idea is to find m representative vectors

$$w_1, ..., w_m, \tag{1}$$

which point in direction of a high concentration or high density of data samples. A cost function is defined, which depends on these m vectors, such that the minimum of the cost function is the desired representation. To define the cost function we proceed as follows: Associated with each representative vector is a subset of the samples, a cluster. We use indicator variables $\lambda_{c,i} \in \{0,1\}$, indicating if sample x_i belongs to cluster c, then $\lambda_{c,i} = 1$, or not, $\lambda_{c,i} = 0$. Each sample belongs maximally to one cluster. In addition we define a margin ρ_c for each cluster, and a hyperplane h_c given by

$$h_c = \{x | w_c \cdot x - \rho_c = 0\}. \tag{2}$$

The hyperplanes are shown in figure 3 by the dashed lines.

The hyperplane for cluster c is estimated such that the samples belonging to cluster c are on the far side of the hyperplane, far from the origin. We allow a fraction ν of samples to be on the side of the origin, which we call the wrong side. Here, $\nu \in [0,1]$ is a parameter. As the distance from the hyperplane to the origin increases, the area of the sphere on the far side of the hyperplane gets smaller. Therefore, if the hyperplane can be chosen far away from the origin, the points in the cluster are highly concentrated. The distance of the hyperplane from the origin in measured by $\rho_c / ||w_c||$. By minimising

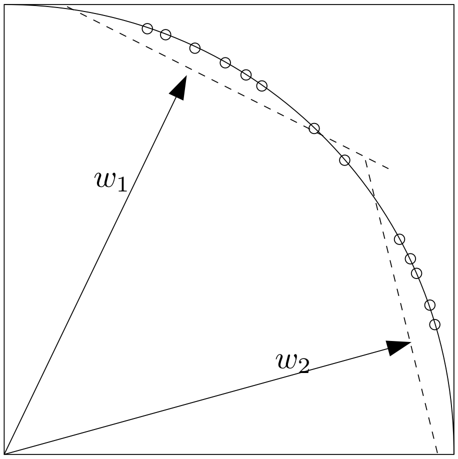

Figure 3: *The idea of how to find representative vectors, see text.*

$$\Omega_c = \frac{1}{2} ||w_c||^2 - \rho_c, \tag{3}$$

we maximise the separation of the hyperplane from the origin, see [9]. An additional term, the empirical risk, is added to control the samples of cluster c sitting on the wrong side of the hyperplane. The empirical risk is

$$R_{emp,c} = \frac{1}{\nu l_c} \sum_i \lambda_{c,i} \big[w_c \cdot x_i - \rho_c \big]_-, \tag{4}$$

where $\big[\cdot\big]_-$ is the function that maps a real argument x to $\max(0, -x)$. Here, l_c is the number of points belonging to cluster c, which is chosen in advance. The empirical risk measures how much the samples x_i belonging to cluster c are sitting on the wrong side of the hyperplane h_c. To have representative vectors, which are as orthogonal as possible, a penalty term is added. The penalty term is

$$\Omega_{div} = \frac{1}{m-1} \sum_{c<d} w_c \cdot w_d, \tag{5}$$

penalising two representative vectors pointing in a similar direction, that is, having a large scalar product. In summary, we consider the constraint optimisation problem

$$\text{minimise} \quad \Omega_{div} + \sum_c (\Omega_c + R_{emp,c}) \tag{6}$$

$$\text{subject to} \quad \lambda_{c,i} \in \{0,1\}, \quad \sum_i \lambda_{c,i} = l_c \tag{7}$$

We solve the optimisation problem by using a stochastic coordinate gradient descent method. In more detail, it is the following two–step procedure. The algorithm proceeds by alternating the two steps, until in has converged or a maximal number of iterations was performed. We recall that the variables are the cluster indicators $\lambda_{c,i}$, the vectors w_c and the margins ρ_c. Our method is similar in spirit to the k–means algorithm [5]. Mathematically speaking, it is a coordinate descent: In the first step, we fix the variables $\lambda_{c,i}$ and ρ_c and update the variables w_c with a stochastic gradient descent. In the second step, we fix the variables w_c and update the variables $\lambda_{c,i}$ and ρ_c. Because the variables $\lambda_{c,i}$ are binary, we do not use a gradient method but rather we will see that we can directly guess a close to optimal point. Next, we present the two optimisation steps. For more details and calculations we refer the reader to [2].

Step one: Let the variables $\lambda_{c,i}$ and ρ_c be fixed. We will compute the partial derivative of the cost function (6) and perform an update

$$w_c \leftarrow w_c - \eta \partial_{w_c}(\Omega_{div} + \Omega_c + R_{emp,c}), \qquad (8)$$

with learning rate η. For efficiency, we do not use all the data to estimate the empirical risk, but rather use only a subset of s examples. We draw these examples randomly from our dataset. Let us denote the set of chosen examples at step t by S_t. A short calculation gives the update rule

$$w_c \leftarrow (1-\eta)w_c - \eta \frac{1}{m-1} \sum_{d \neq c} w_d + \eta \frac{m}{\nu s} \sum_{x_i \in S_t} \theta_{c,i} \lambda_{c,i} x_i. \qquad (9)$$

Here $\theta_{c,i} \in \{0,1\}$ is zero if $w_c \cdot x_i > \rho_c$ and 1 otherwise. The value ν is the relative number of points in each cluster, which are on the wrong side of the margin. The integer s is the number of examples in S_t. The factor $m/\nu s$ in front of the last term in (9) is just one over the expected number of examples in S_t, which belong to cluster c and are on the wrong side of the margin.

Step two: In this step we fix the variables w_c and optimise with respect to the variables $\lambda_{c,i}$ and ρ_c. The algorithm will proceed by first choosing good values for the indicators $\lambda_{c,i}$ and then given these values, determine the optimal margins ρ_c.

To motivate the choice of good values for the indicators, we first look at the optimal value given the binary values $\lambda_{c,i}$. Thus, suppose each point x_i is assigned to one cluster according to (7). Setting the optimal margin ρ_c is now an independent problem for each cluster c. The optimal values of the margins are set as follows. For a cluster c define $a_{c,i} = w_c \cdot x_i$ for all x_i belonging to c and sort the values $a_{c,i}$ in ascending order,

$$a_{c,j_1} \leq a_{c,j_2} \leq ... \leq a_{c,j_{l_c}}. \qquad (10)$$

Then we set

$$\rho_c = a_{c,j_n}, \quad \text{with} \quad n = \lceil \nu l_c \rceil. \qquad (11)$$

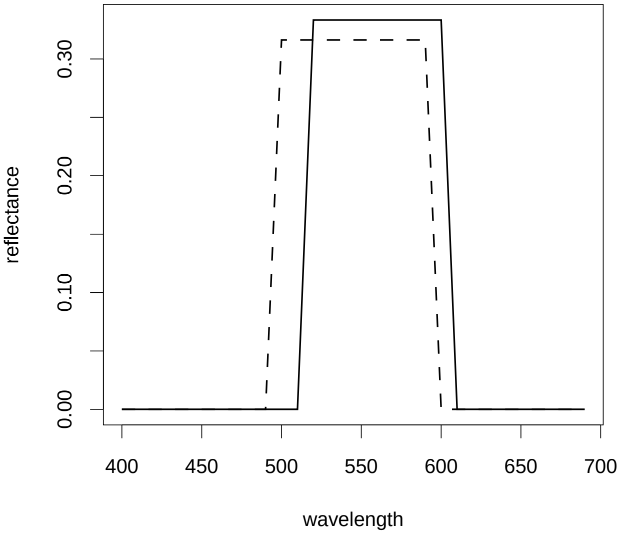

Figure 4: The creation of the toy data. Two randomly generated spectra starting from the same interval. The width of the spectra is randomly chosen. The height is adjusted such that each spectra has unit length.

Hence, ρ_c is sitting at the ν–quantile of the empirical distribution of $\{a_{c,j_1}, ..., a_{c,j_{l_c}}\}$.

From the above we see that suitable values for the indicators $\lambda_{c,i}$ are values that allow large values of ρ_c, because the margins have to be maximised by (3). According to (11), we should choose points x_i to belong to cluster c if their dot product with w_c is large. For this we chose a simple strategy. We iterate over all clusters. For the first cluster we calculate the values $a_{c,i} = w_1 \cdot x_i$ for all points x_i and select l_c points with a maximal value of $a_{c,i}$. For these points we set the indicator $\lambda_{1,i} = 1$. Now suppose the indicators for clusters 1 to $c-1$ are set. Then for cluster c we calculate the values $a_{c,i}$ for all points x_i, which do not already belong to one of the previous clusters. Again, we set the indicators $\lambda_{c,i} = 1$ for the points x_i with maximal values $a_{c,i}$. After the indicators are set for all clusters, the optimal margins ρ_c are determined according to (11) and step two is finished.

From the point of view of efficiency, we can adjust the complexity of the first step by choosing a suitable fraction of the data to estimate the empirical risk term. In step two, all the scalar products of the data with the vectors w_c have to be computed. Therefore, the complexity of one iteration is of the order of ml.

The functional we optimise is a Support Vector type functional, see [10]. As solution we find the representative vectors $w_1, ..., w_m$, the margins $\rho_1, ..., \rho_m$, and the indicators $\lambda_{c,i}$. By construction, the vectors w_i are expressible as a linear combination of data samples with positive coefficients and are therefore positive. As a remark we note that

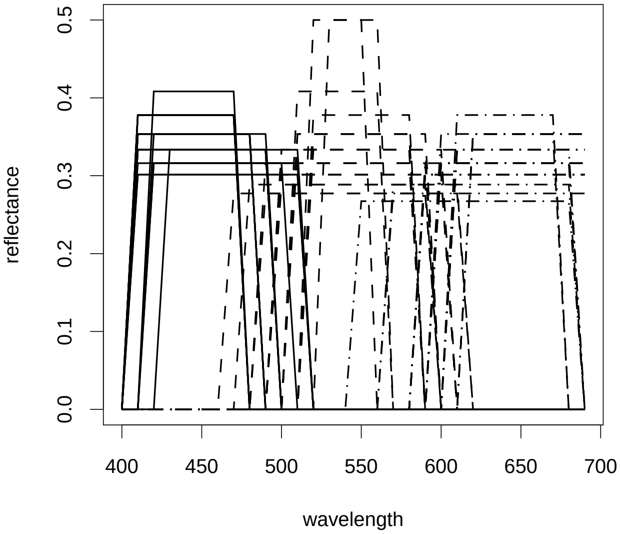

Figure 5: *The toy dataset consisting of* 20 *samples for each of the three starting intervals (solid, dashed, dashed–dotted lines).*

the scalar product in the formulation of our optimisation problem can be replaced by a kernel. Thus, our algorithm can naturally be generalised to a kernel algorithm, [10].

Before we apply the algorithm to the dataset of spectral reflectances in subsection 3.2 a toy problem is discussed in the next subsection.

3.1. Toy example

Let us first apply our algorithm to the following toy problem. We try to find representative vectors in an artificially created dataset. The data consists of a set of artificial spectra, where each spectra f_I is equal to a constant c on the interval I and zero otherwise,

$$f_I(x) = \begin{cases} c & \text{if} \quad x \in I \\ 0 & \text{otherwise.} \end{cases} \tag{12}$$

The constant c is chosen such that $||f_I|| = 1$. Starting with an interval I_1 a new interval I is generated by moving the interval boundaries by random values σ_a, σ_b. In detail, if the starting interval is $I_1 = [a, b]$ then

$$I = [a + \sigma_a, b + \sigma_b] \tag{13}$$

is a randomly generated interval with samples σ_b, σ_b drawn from a Gaussian distribution.

Figure 4 shows two artificial spectra generated randomly starting from the same interval. In this manner, choosing three adjacent intervals I_1, I_2, I_3, twenty randomly generated spectra are created starting with each of the three intervals. Figure 5 shows the whole dataset. Next, let us forget for the moment what we know about the construction of our toy data. We try to recover the structure in the

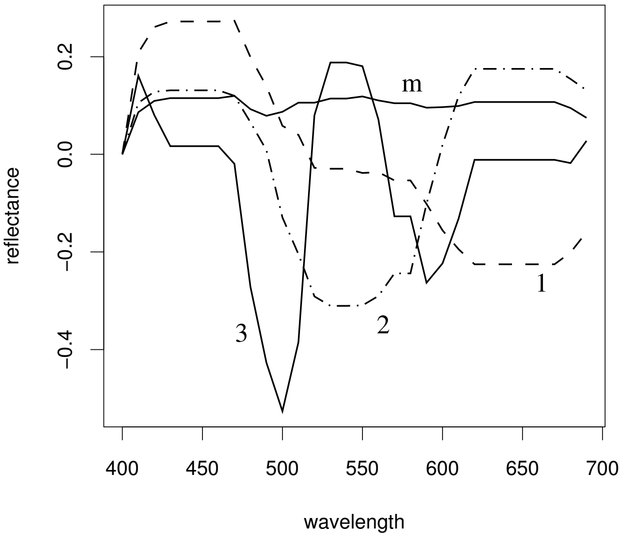

Figure 6: *The mean (m) and the first three principal components (1-3) of the toy data.*

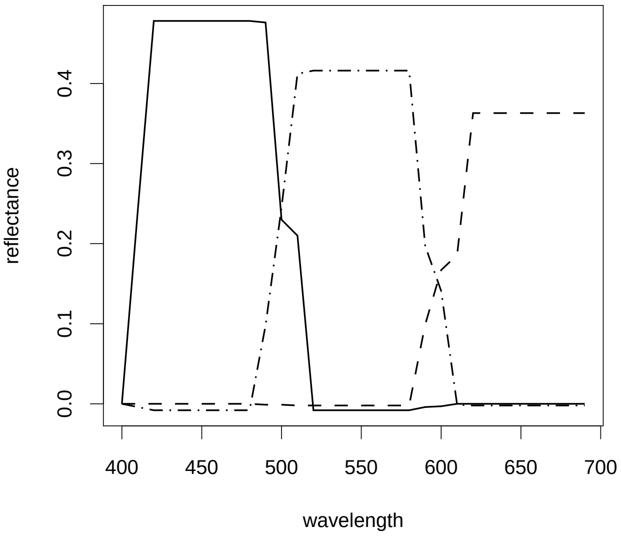

Figure 7: *The three representative vectors of the toy data found by our algorithm (solid, dashed, dashed–dotted lines).*

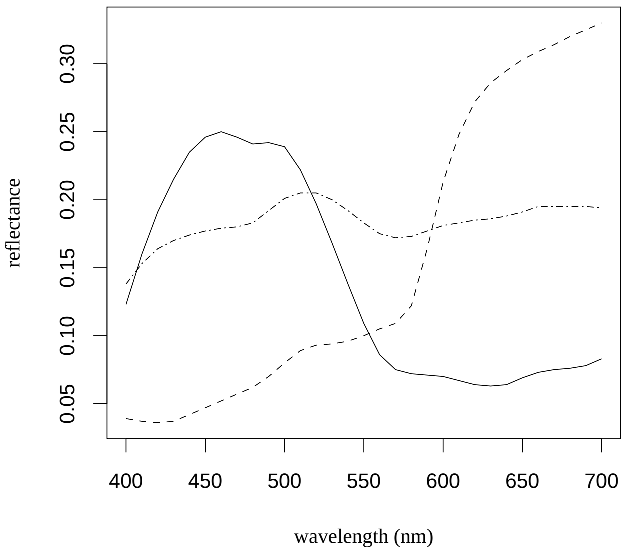

Figure 8: *Solution for $m = 3$: shown are the three representative vectors $w_1, .., w_3$.*

dataset. First, the principal components are computed, second, our algorithm is applied. Figure 6 shows the mean and the first three principal components. It can be seen that the principal components do not visually reveal the structure underlying the dataset. Using our algorithm three representative vectors are computed. They are shown in Figure 7. The representative vectors reveal the structure in the toy data. Each of them represents the set of artificial spectra generated starting with the same interval. By definition of the algorithm, the representative vectors are not constraint to be pairwise orthogonal. In our example, relaxing the orthogonality constraint is important. Nevertheless, the penalty term (5) yields an almost pairwise orthogonal solution.

3.2. Representative Vectors of SOCS data

Figure 8 shows the representative vectors of a solution for $m = 3$. The parameters used are $l_c = 0.6 * l/m$, for all c, where l is the number of samples in the database, and $\nu = 0.2$. We tested how good we can reconstruct the spectra from the projections on three representative vectors. We found a mean squared error of 2.6%, compared to the optimal method, a reconstruction from the projections on the first three principal components, with a mean squared error of 0.9%. We note that for the latter method, one has to keep not only the three principal components, but the mean as a fourth vector, too. Reconstruction using the mean and first two principal components yields a mean squared error of 2.0%.

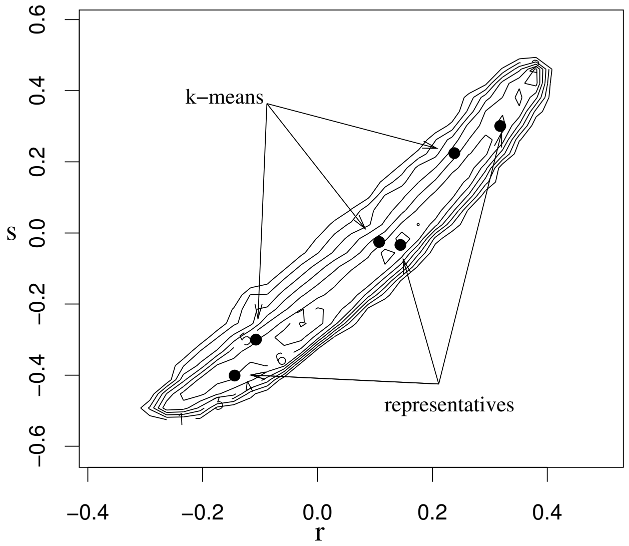

Figure 9: *A density plot of the projections onto three representatives.*

4. Projection on three representatives

In the case of three representatives, we can easily visualise the situation. We normalise the three representatives w_1, w_2, w_3 and proceed as follows: We map a point x into a three dimensional subspace by

$$x \mapsto x' = (x \cdot w_1, x \cdot w_2, x \cdot w_3). \tag{14}$$

As we are interested by the relative amount of the coordinates $(x \cdot w_i)$ only, the vector x' is divided by its norm,

$$x' \mapsto x'' = (x''_1, x''_2, x''_3) = \frac{x'}{||x'||}. \tag{15}$$

The point x'' sits on the surface of a two dimensional sphere, and we project the point from the sphere into the plain by introducing coordinates r, s defined by

$$r = -\sqrt{3}/2x''_2 + \sqrt{3}/2x''_3 \tag{16}$$

$$s = x''_1 - 1/2(x''_2 + x''_3). \tag{17}$$

Figure (9) shows a density plot of the samples in the database. The points show the three representative vectors found. It can be seen how the vectors sit at positions of high density. As a comparison, we plotted three vectors found with a k–means algorithm. Compared to our representatives, these vectors sit at locations of lower density.

5. How many representatives ?

In our algorithm the number of representatives has to be chosen in advance. Naturally, the following question arises:

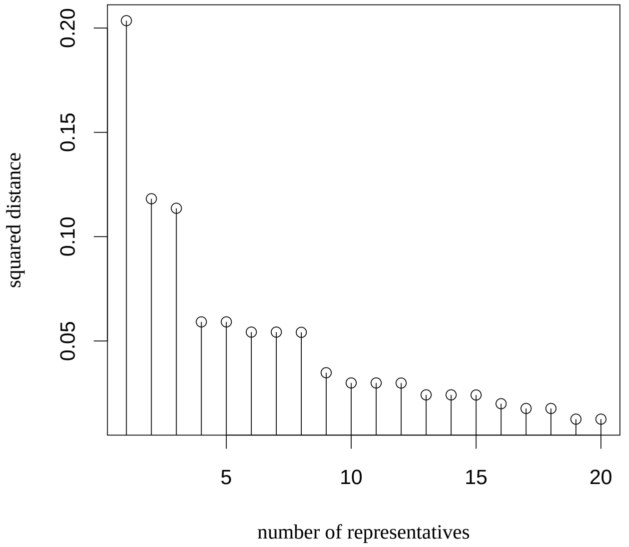

Figure 10: *For* 20 *representatives the plot shows the squared difference to the closest other representative.*

how many representatives have to be chosen ? Many researchers already addressed this question [6, 11, 3, 7, 4] and gave various answers, depending on the criteria used. Our method is different and the dataset larger, therefore our result can only be partially compared with others. We ran the algorithm for $m = 20$, and then we checked how different these 20 representatives are. Difference is measured by a squared L_2–difference. Figure 10 shows the result. Plotted is for each representative w_c the minimal difference,

$$\min_{d \neq c} ||w_c - w_d||^2, \tag{18}$$

to another representative. The representatives are rearranged, in descending order. There are clearly visible jumps in the plot. This can give some indication of how many representatives to choose to well represent the dataset. We can set a threshold , for example at 10%. In other words, we take only those representatives, which differ more than 10% from each other. From the plot we see that we should retain 3 representatives. Setting the threshold at 5% tells us to retain 8 representatives. This is consistent with findings by others [6, 11, 3, 7, 4].

6. Discussion

We proposed a new method to find representative spectra in a large collection of reflectance spectra. We interpreted and compared our result with well-known methods like principal component analysis and k–means. In reconstructing the samples from the projections on the representative vectors, the reconstruction error is larger than it is using the optimal principal component vectors. But unlike the principal components, our representative vectors have a direct interpretation as prototype colours. They are physically realisable and could be used to construct a test target, that is, a small collection of typical colours, like the MacBeth Color Checker [8].

Geometrically, our representatives sit at points of high density in the dataset. Therefore, they are more prototype–like than representatives found with a k–means algorithm.

References

[1] *Standard Object Colour Spectra Database for Colour Reproduction Evaluation (SOCS).* TR X 0012:1998, Information Processing Divisional Council, Japanese Standards Association.

[2] BORER, S. New support vector algorithms for multicategorical data applied to real–time object recognition. *PhD Thesis, Swiss Federal Institute of Technology (EPFL) Lausanne, Switzerland* (2003).

[3] COHEN, J. Dependency of the spectral reflectance curves of the Munsell color chips. *Psychonomic Science 1* (1964), 369–370.

[4] DANNEMILLER, J. Spectral reflectance of natural objects: How many basis functions are necessary? *Journal of the Optical Society of America A 9* (1992), 507–515.

[5] DUDA, R. O., AND HART, P. E. *Pattern Classification and Scene Analysis.* John Wiley & Sons, 1973.

[6] KRINOV, E. Spectral reflectance properties of natural formations. Tech. rep., National Research Council of Canada, 1947. Technical Translation: TT-439.

[7] MALONEY, L. Evaluation of linear models of surface spectral reflectance with small numbers of parameters. *Journal of the Optical Society of America A 3*, 10 (1986), 1673–1683.

[8] MCCANY, C., MARCUS, H., AND DAVIDSON, J. A color-rendition chart. *Journal of Applied Photographic Engineering 2* (1976), 95–99.

[9] SCHÖLKOPF, B., PLATT, J. C., SHAWE-TAYLOR, J., SMOLA, A. J., AND WILLIAMSON, R. C. Estimating the support of a high–dimensional distribution. *Neural Computation 13*, 7 (2001).

[10] SCHÖLKOPF, B., AND SMOLA, A. J. *Learning with Kernels.* MIT Press, 2002.

[11] VRHEL, M., GERSHON, R., AND IWAN, L. Measurement and analysis of object reflectance spectra. *Color Research and Application 19*, 1 (1994), 4–9.