An Implementation for Memory Efficient and Data Type Independent Linear and Planar Procrustes Alignment

Karl Skoglund

Informatics and Mathematical Modelling, Technical University of Denmark

September 8, 2003

Abstract

This document describes a method for performing one- and two-dimensional Procrustes alignment in a memory efficient and data type independent manner. The alignment does not change the location, rotation and scale more than necessary, a desirable property in many situations. This also makes it possible to use the same data type for the result as for the original data. The methods is efficient since it only updates a single object vector during the optimization.

1 Introduction

Procrustes analysis [3, 1, 4] is a well known and often used technique for e.g. obtaining the pure shape from a set of objects. Common implementations follow the algorithms in e.g. [2]. These affect the location and scale severely by translating all objects to the origin and normalizing their scale to unit size. In one and two dimensions, there exists a simple expression for aligning a shape onto another (called *ordinary Procrustes alignment*). This can be used to avoid alteration of the data or a copy of the data while calculating the mean. Once the mean is found, the objects can be aligned. This introduces few numerical errors and preserves the overall scale of the objects.

2 Method

Assume that there exists a set of n objects $\mathbf{w}_i$, $1 \leq i \leq n$, of length k in one or two dimensions. In the linear case, the vectors $\mathbf{w}_i$ simply consist of real numbers of some data type. Planar objects are represented by vectors of complex numbers where the real and the imaginary part represent the two variables. Here, the planar case is presented, but the linear case is similar [5]. The transpose of the complex conjugate operator * is simply changed into the transpose operator ^{T}.

The methods presented here require that the data is *centered*, i.e.

$$\mathbf{w}_i^* \mathbf{1}_k = 0, \ \forall \ i \tag{1}$$

This can be fulfilled by subtracting the *centroid*, c_i, from each object.

$$c_i = \mathbf{w}_i^* \mathbf{1}_k / k \tag{2}$$

The mean of all such centroids is denoted $\bar{c}$. This is equal to the centroid of the estimated mean of all objects. The centered version of this mean is denoted $\bar{\mathbf{w}}$,

$$\bar{\mathbf{w}} = \left(\frac{1}{n} \sum_{i=1}^{n} \mathbf{w}_i \right) - \bar{c} \mathbf{1}_k \tag{3}$$

In [2], a simple expression for aligning an object $\mathbf{w}_i$ to the mean $\bar{\mathbf{w}}$ is derived.

$$\tilde{\mathbf{w}}_i = \frac{\mathbf{w}_i^* \bar{\mathbf{w}}}{\mathbf{w}_i^* \mathbf{w}_i} \mathbf{w}_i \tag{4}$$

As mentioned before, this equation requires that all vectors are centered. By use of the centroid of each object, the alignment can be performed without centering the actual data.

$$\tilde{\mathbf{w}}_i = \frac{(\mathbf{w}_i - c_i \mathbf{1}_k)^* \bar{\mathbf{w}}}{(\mathbf{w}_i - c_i \mathbf{1}_k)^* (\mathbf{w}_i - c_i \mathbf{1}_k)} (\mathbf{w}_i - c_i \mathbf{1}_k) + \bar{c} \mathbf{1}_k \tag{5}$$

As always with Procrustes analysis, the problem is that the true mean is unknown and defined in terms of the *aligned* objects. In one and two dimensions, the true mean can be found analytically as the eigenvector corresponding to the largest eigenvalue of the (complex) sum of squares and products matrix,

$$S = \sum_{i=1}^{n} \frac{\mathbf{w}_i \mathbf{w}_i^*}{\mathbf{w}_i^* \mathbf{w}_i} \tag{6}$$

If the number of variables is small, this method can be used to find the mean shape and then simply aligning all objects using equation 5. Usually, the length of each object vector is large. The matrix S has size $(k \times k)$, and performing an eigenanalysis may be computationally infeasible. The true mean must then be found by means of a *generalized Procrustes analysis*. This is an iterative process where the initial mean estimate $\bar{\mathbf{w}}$ is improved until it converges to the true mean. Normally the object vectors have to be standardized and continuously updated during this process. The method presented here manages to avoid this.

The proposed algorithm proceeds as follows:

1. Compute the centroid c_i of each object vector
2. Let $\bar{\mathbf{w}}$ be the first estimate of the mean
3. Align all objects using

$$\tilde{\mathbf{w}}_i = \frac{(\mathbf{w}_i - c_i \mathbf{1}_k)^* \bar{\mathbf{w}}}{(\mathbf{w}_i - c_i \mathbf{1}_k)^* (\mathbf{w}_i - c_i \mathbf{1}_k)} (\mathbf{w}_i - c_i \mathbf{1}_k) \tag{7}$$

   However, the actual objects are not updated. Instead, the results are added to an empty vector. This vector is then divided by the number of objects. This results in a new estimate of the true mean. The expression for this estimate is

$$\tilde{\bar{\mathbf{w}}} = \frac{1}{n} (\tilde{\mathbf{w}}_1 + \ldots + \tilde{\mathbf{w}}_n) \tag{8}$$

4. Repeat step 3, aligning the objects to the current mean estimate, until a stable solution for the mean is found.
5. Align all examples using equation 5 to the true mean.

The algorithm is simple and straightforward. However, a problem occurs. The new estimate of the mean often has a different size compared to the former estimate. This makes the mean drift towards zero size and the algorithm will not converge. The scalings of the object vectors found through equation 7 are optimal for fitting the objects to the current mean estimate, but these have to be adjusted to maintain the scale. The easiest way of doing this is simply by centering the scales, so that the mean scale is 1. The updated algorithm is as follows:

1. Compute the centroid c_i of each object vector
2. Let $\bar{\mathbf{w}}$ be the first estimate of the mean
3. Compute the scalings s_i that align the objects to the current mean estimate,

$$s_i = \frac{(\mathbf{w}_i - c_i \mathbf{1}_k)^* (\bar{\mathbf{w}} - \bar{c} \mathbf{1}_k)}{(\mathbf{w}_i - c_i \mathbf{1}_k)^* (\mathbf{w} - c_i \mathbf{1}_k)}, \ \forall \ i \tag{9}$$

4. Center the scales around unit scale by

$$\tilde{s}_i = s_i - \frac{1}{n} \sum_{j=1}^{n} s_j + 1, \ \forall \ i \tag{10}$$

5. Compute the new mean estimate by

$$\tilde{\bar{\mathbf{w}}} = \frac{1}{n} (\tilde{s}_1 (\mathbf{w}_1 - c_1 \mathbf{1}_k) + \ldots + \tilde{s}_n (\mathbf{w}_n - c_n \mathbf{1}_k)) \tag{11}$$

6. Iterate step 3-5 until the mean is stable
7. Align all examples using equation 5 to the true mean.

2.1 Memory Requirements

The algorithm has low memory requirements, except for the inevitable space taken up by the original object vectors. Two additional object vectors are required for the calculation of the mean. Convergence is declared when the norm of the difference between the the new and old estimates falls below a certain threshold,

$$\| \tilde{\bar{\mathbf{w}}}_{new} - \tilde{\bar{\mathbf{w}}}_{old} \| < \delta \tag{12}$$

Another convergence criterion that only requires the current mean estimate for the calculation would make it possible to use a single object vector for the mean.

The algorithm also keeps a list of the centroids of the objects. The memory requirements for this list is usually tiny, since the number of objects in most cases is small, especially compared to the length of the object vectors.

2.2 Data Type Independence

Since the resulting object vectors will be as close to their original counterpart as possible, the same data type can be preserved. While writing the results back to the object vectors, the values must be clamped to the limits of the data type used to avoid under- and overflow. Of course, if the data type is of integer format, the accuracy will be limited. However, the accuracy is limited already in the original assignment when using integers.

2.3 Convergence

Since the scaling factors are altered they are no longer optimal. This gives slower convergence of this algorithm than for normal Procrustes analysis. Also, the algorithm may not converge in all situations, this remains to be proven. Experiments show that convergence normally can be declared within 3-10 iterations.

3 Applications

Planar Procrustes analysis is often used to align a set of 2-D shapes, which have each been defined by a set of landmarks. This method is especially applicable when preserving the overall scale of the objects is important.

One-dimensional Procrustes alignment can be used aligning brightness and contrast in a set of images. The image intensities of an image are simply put into a single vector. Note that the extent and shape of the images must be similar. This method cannot be used to align a set of images with differing contents. The same algorithm can be used to align color images. A separate alignment is then performed for each color channel. This will align both color balance and brightness/contrast. If these properties differ significantly, a multidimensional Procrustes analysis of all color channels at once may give better results.

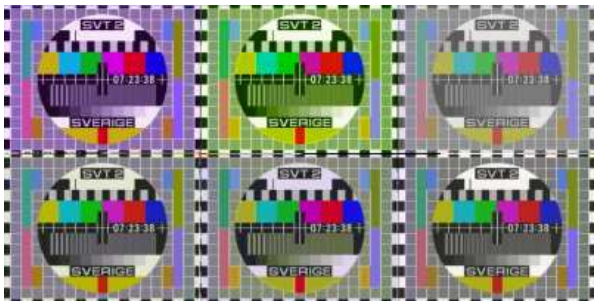

Figure 1: Before (top row) and after (bottom row) comparison.

4 Results

Figure 1 shows six versions of the same image. The top row contains three examples distorted in brightness, contrast and color balance. The bottom row shows the pictures after alignment. The figure shows that brightness and contrast differences are filtered out satisfyingly, while small deviations in color balance still can be seen after alignment. A multidimensional Procrustes analysis might improve this.

References

[1] J. M. F. Ten Berge. Orthogonal Procrustes rotation for two or more matrices. *Psychometrika*, 42:267–276, 1977.

[2] Ian L. Dryden and Kanti V. Mardia. *Statistical Shape Analysis*. John Wiley & Sons, 1999.

[3] C. Goodall. Procrustes methods in the statistical analysis of shape. 53(2):285–339, 1991.

[4] J. C. Gower. Generalized Procrustes analysis. *Psychometrika*, 40:33–50, 1975.

[5] K. Skoglund. Three-dimensional face modelling and analysis. Master's thesis, IMM, Technical University of Denmark, Richard Petersens Plads, Building 321, DK-2800 Kgs. Lyngby, August 2003.