Segmentation of clusters by template rotation
expectation maximization

Carl-Magnus Svensson¹ and Marc Thilo Figge¹,²

1. Leibniz Institute for Natural Product Research and Infection Biology, Hans-Knöll Institute, Jena, Germany
2. Friedrich Schiller University Jena, Germany

Abstract. To solve the task of segmenting clusters of nearly identical objects we here present the template rotation expectation maximization (TREM) approach which is based on a generative model. We explore both a non-linear optimization approach for maximizing the log-likelihood and a modification of the standard expectation maximization (EM) algorithm. The non-linear approach is strict template matching, while the linear TREM allows for a more deformable model. As benchmarking we compare TREM with standard EM for a two dimensional Gaussian mixture model (GMM) as well as direct maximization of the log-likelihood using non-linear optimization. We find that the linear algorithms, TREM and standard GMM, are faster than the non-linear algorithms without any loss of segmentation accuracy. In the comparison between the linear models we find an edge for the linear TREM on our data as it better approximates the orientation and shape of the objects than the standard GMM.

Keywords: Segmentation, generative model, expectation maximization, template matching

1 Introduction

Template matching is a general term for matching an object with known features or shapes to a more complicated scene where the object may be present. Different variants of template matching are widely used in optical character recognition, medical image segmentation, face detection, and action recognition [7, 5]. For a template to be matched in the scene there is normally a large number of possible transformations to be considered such as object scaling, rotation and translations. This means that the computational cost of the matching normally is high and to match numerous objects in different configuration fast leads to a combinatorial explosion [17, 9].

This paper will demonstrate the use of the template rotation expectation maximization (TREM) approach for segmentation of individual objects in clusters. By assigning single pixels to terms of a two-dimensional Gaussian mixture model (GMM) when we have prior information about how the objects look, we will separate clusters of objects with a known shape. While TREM is generally applicable to different areas of computer vision it is of special interest for
automated image analysis of microscopy images, where the occurrence of many identical objects that needs to be segmented occurs regularly [12]. These objects are not seldom tightly clustered to an extent that standard segmentation techniques, such as thresholding or watershed, becomes extremely challenging. For template matching in general, an approximate match often is sufficient [7] but is often not useful in microscopy images as we require high resolution, especially when tracking the segmented object. It is also the case that template matching normally breaks down when objects are touching [27]. For object segmentation in microscopy images expectation maximization (EM) [1] has been combined with template matching in semi-automated algorithms to extract features from high-resolution images of zebrafish [20] and for segmentation of cell nuclei using with utilizing supervised learning [2]. For object clusters that efficiently can be skeletonized the combination of EM and template matching the method for untangling C. elegans clusters by Riklin Raviv et al. [22] is an attractive method. Unsupervised method that works directly on shapes in binary or grey scale images are constrained to either strongly rod-like [13, 26] or almost perfectly circular/spherical cells [25]. TREM is applicable to both those extremes but also to the whole spectrum of oval and rectangular objects. For non-biological systems these shape restrictions might seem strongly limiting but in biology there are several bacteria [4, 24], fungal spores [14] and algae [21] that are fulfilling the criteria of shapes that can be well described by a two dimensional Gaussian function. Because of its probabilistic formulation TREM is robust to small and medium object deformations and moderate affine transformations other than translation and rotation. This paper evaluates two different approaches in fitting a Gaussian mixture model where a fixed template for the shape of the objects is imposed through the definition of a fixed covariance matrix, \( \Sigma_0 \). We consider the case of pure rotation of the template in a hybrid algorithm of classical EM for location and non-linear optimization for orientation, called non-linear TREM. We also consider a linear TREM formulation that restricts the deformation of the covariance matrix so that the determinant is not changed while applying EM parameter fitting. To evaluate the performance of TREM we compare it to an GMM without any restrictions on the covariance matrix. On the other extreme we are also comparing it to the direct log-likelihood maximization (DLLM), where all parameters of the log-likelihood are estimated using non-linear optimization. Although this paper only deals with synthetic data the method could be readily applied to for example clusters of Escherichia coli [24].

2 Generative Model

In principle, TREM can be applied to higher dimensions than two, for the case of image analysis the case of three dimensions is of particular interest, but in this paper we will consider the case of two dimensions for clarity. We assume that the pixels belonging to an foreground object are a realization of the probability function \( p(x_n) \), where \( x_n \) is the pixels that are part of the object. We formulate
the log-likelihood as
\[ L(\theta) = \sum_{n=1}^{N} \log(p(x_n|\theta)). \]  

(1)

As in other maximum likelihood formulations, we seek the parameters, \( \theta \), that maximises the log-likelihood given the data \([1]\). The generative model of a cluster containing \( K \) objects express the probability that a pixel at position \( x_n \) is part of object \( k \). We assume that this probability is described by the GMM

\[ p(x_n|\theta) = \sum_{k=1}^{K} p(x_n|k, \theta)p(k) \]

\[ p(x_n|k, \theta) = \mathcal{N}(x_n, \mu_k, \Sigma_k), \quad p(k) = 1/K. \]

In this model we assume that the prior distribution, \( p(k) \), is discretely uniform based on that we are considering near identical objects where the intensities, at least within a cluster, are approximately equal. The noise model, \( p(x_n|k, \theta) \), is a two dimensional Gaussian governed by the mean, \( \mu_k \in \mathbb{R}^2 \), and covariance matrix \( \Sigma_k \in \mathbb{R}^{2 \times 2} \). For template rotation, the template we want to match is a constant diagonal matrix, \( \Sigma_0 = \text{diag}([\sigma_1, \sigma_2]) \), that can be rotated by a matrix \( R(\phi) \in \mathbb{R}^{2 \times 2} \). The constant part of the covariance matrix \( \Sigma_0 \) describes the unrotated shape of the objects we want to segment. The covariance matrix is in non-linear TREM formed by rotating the template \([19]\) according to

\[ \Sigma_k = R(\phi_k) \Sigma_0 R(\phi_k)^T, \quad R(\phi_k) = \begin{bmatrix} \cos(\phi_k) - \sin(\phi_k) \\ \sin(\phi_k) \cos(\phi_k) \end{bmatrix}, \]

(3)

where \( \phi_k \in [-\pi, \pi] \) is the rotation of object \( k \). We follow the standard steps of EM \([3, 16]\) by formulating the variational free energy as

\[ F(\theta, q) = \sum_{n=1}^{N} \left[ \sum_{k=1}^{K} q(k) \log(p(x_n|k, \theta)) + \log(p(k)) \right] + H(q) \leq L(\theta). \]

(4)

Here, \( q(k) \) is an arbitrary probability function and the Shannon entropy of \( q(k) \) is defined as \( H(q) = -\sum_{n=1}^{N} \sum_{k=1}^{K} q(k) \log(q(k)) \). The EM algorithm then alternately increases \( F(\theta, \theta') \) by finding \( q \) while fixing the parameters \( \theta \) (E-step) and then updating the parameters \( \theta \) while considering \( q \) to be constant (M-step). In practice, given a set of parameters \( \theta' \) we perform the E-step by setting \( q(k|\theta') = p(k|x_n, \theta') \). The variational free energy can now be formulated as a function of variable and fixed parameters, \( F(\theta, \theta') \). During the M step we take the derivative of \( F(\theta, \theta') \) with regard to the parameter of interest and set it to zero. For the mean of the Gaussians we get the usual update rule for GMMs

\[ \mu_k = \frac{\sum_{n=1}^{N} q(k|\theta') x_n}{\sum_{n=1}^{N} q(k|\theta')}. \]

(5)
Unfortunately the formulation of $\Sigma_k$ and the non-linearity of the rotation matrix does not allow us to find a similar update rule for $\phi_k$ which is the only remaining free parameter. We can however use non-linear optimization to find

$$\arg\max_{\phi_k} \mathcal{L}(\phi_k|\mu_k).$$

(6)

For the maximization of Eq. (6) we are using the Nelder-Mead algorithm [15] as this is a stable algorithm for high dimensional problems and is therefore suitable as the dimension is governed by the number of objects in a cluster.

As an alternative we can rescale the template according to

$$\Sigma_k = \Sigma_0^{1/2} C_k \Sigma_0^{1/2}. \quad (7)$$

This is a different approach than the pure rotation we have seen in non-linear TREM as we are considering a rescaling of the template [8]. To limit the deformation of the template we solve $\frac{\partial \mathcal{R}(\theta, \theta')}{\partial C_k} = 0$ with the constraint that $\det(C_k) = 1$ using Lagrange multipliers [1]. This constraint will ensure that the scaled covariance matrix is retaining the same determinant as $\Sigma_0$ and while the main axes are allowed to change their length they can not scale independently. The update rule for the scaling matrix is given by

$$C_k = \frac{\sum_{n=1}^{N} q(k|\theta') \left[ (\Sigma_0^{-\frac{1}{2}}(x_n - \mu_k))(x_n - \mu_k)^T \Sigma_0^{-\frac{1}{2}} \right]}{\det \left( \sum_{n=1}^{N} q(k|\theta') \left[ (\Sigma_0^{-\frac{1}{2}}(x_n - \mu_k))(x_n - \mu_k)^T \Sigma_0^{-\frac{1}{2}} \right] \right)^{1/2}}. \quad (8)$$

This linear TREM algorithm then fits the Gaussians by Eq.s (5) and (8) using standard EM iterations.

We compare TREM with the use of non-linear optimization directly on Eq. (1) and find

$$\arg\max_{\phi_k, \mu_k} \mathcal{L}(\phi_k, \mu_k),$$

the direct log-likelihood maximization (DLLM) described by (9) will be solved using Nelder-Mead optimization just as in the case of non-linear TREM.

We also compare it with traditional EM for GMMs, e.g. where the unconstrained covariance matrix is updated according to

$$\Sigma_k = \frac{\sum_{n=1}^{N} q(k|\theta')(x_n - \mu_k)(x_n - \mu_k)^T}{\sum_{n=1}^{N} q(k|\theta')} \quad (10)$$

In total we consider four ways to fit Gaussian functions to the relevant data. The normal un-constrained GMM where $\mu_k$ is updated according to (5) and the covariance matrix by (10). In linear TREM the mean is updated as in the GMM, the scaling matrix $C_k$ is updated following (8) and the covariance matrix is calculated using (7). The non-linear TREM is updating the means using (5) but the rotation of $\Sigma_0$ is found by non-linear optimization. Finally we directly maximize the log-likelihood by non-linear optimization of (9) in the DLLM algorithm. We will evaluate these four approaches by looking at their ability to
find the position and orientation of rectangular objects in clusters, segmentation accuracy and in terms of the computation time required for each algorithm.

All algorithms are initiated at four different initial positions, two are placing the \( k \) Guassians at an equally spaced in the angular direction at distance \( R = 20 \pm 3 \) pixels away from the cluster center. Each cluster is rotated with \( \phi_k \) chosen from a uniform \( U(-\pi, \pi) \) distribution. The third initialization are the centres found after K-means clustering of the pixels in the cluster and the fourth initialization is a \( \pi/2 \)-rotation of the K-means centres around the center of the cluster. The use of just K-means for initialization normally causes linear TREM and standard GMM to converge to the same solution, therefore some variability in initialization is important to find different configurations. All algorithms are stable in convergence for different initializations as long as they are reasonably close to the cluster, e.g. the nearest pixel in the cluster is not more than ten or so standard deviations away from the centre of the Guassian. For GMM, non-linear TREM and DLLM we then pick the solution that has the highest log-likelihood and for the linear TREM we chose the solution that has deformed the template the least. This is done by calculating,

\[
T_{\text{init}} = \sum_{k=1}^{K} |\text{tr}(\Sigma_k) - \text{tr}(\Sigma_0)|
\]

and then determine which initial condition that produced the smallest \( T_{\text{init}} \). The reason for this condition is that for a pure rotation of the template its trace is unchanged. In theory this condition can be applied to traditional GMM, but in practice the unconstrained determinant of the covariance matrix in this case means that there is no guarantee that the matrix with the traces most like the template is most shaped like the template. All code implementation was made in Python and the code is available from the authors upon request.

3 Synthetic data

We created synthetic data by placing rectangular objects with length \( L \) and width \( B \) in an image at uniformly random positions and individually rotating each object by an angle that is drawn from a uniform distribution. The intensity of pixel \( i,j \) given \( P \) objects with centres \([i_p, j_p]^T\) and rotation \( \phi_p \) is given by

\[
I_{i,j} = \sum_{p=1}^{P} f_L \left( [i,j]^T - [i_p, j_p]^T \right) R(\phi_p) \cdot f_B \left( [i,j]^T - [i_p, j_p]^T \right) R(\phi_p) + \xi
\]

\[
f([x,y]^T)_L = \begin{cases} 
1 & \text{if } x < L/2 \\
\frac{1}{e^{-|x|-L/2}} & \text{else}
\end{cases}
\]

\[
f([x,y]^T)_B = \begin{cases} 
1 & \text{if } y < B/2 \\
\frac{1}{e^{-|y|-B/2}} & \text{else}
\end{cases}
\]

(12)
The rotation matrix is defined as in Eq. (3) and $\xi$ is $\mathcal{N}(\xi, 0, 0.025)$-distributed noise. The intensities are thresholded at $I = 0.95$ to acquire binary images. In Fig. 1 we present an example of an intensity image (left panel) and the corresponding binary image (middle panel). Rightmost in Fig. 1 we have plotted Gaussians with the same centre and orientation as the original objects. In our synthetic data the size of the rectangles are $L = 20$ px and $B = 4$ px and the un-rotated covariance matrices are $\Sigma_0 = \text{diag}(25, 1)$, thereby conserving the 5:1 ratio of major to minor axis. For both flavours of TREM as well as for DLLM this is also the template. We are generating 10 images, each with 200x200 px, and place in total 193 objects in them. After thresholding, 97 of these objects were found to be so close to each other that they form a cluster of two or more objects.

**Fig. 1.** From left to right we see an intensity image generated by Eq. (12), the binary version and finally Gaussians with the same centre and rotation as the rectangular objects.

## 4 Results

When evaluating the algorithms we will only look at clusters where two or more objects are merged in the binary images. In the case of isolated object it is not to be expected that the algorithms differs, other than in run-time, and the use of a generative model is anyway not necessary to identify the position and orientation of these objects. In Fig. 1 we have three clusters containing in total eight objects (two cluster with two objects and one with four objects). The goodness of fit will be evaluated by comparing the centres of the segmented regions from the binary images with their distances to the centres of the original objects. The distance between the centres of a ground truth object, $\mu_{GT}$, and a fitted Gaussian, $\mu_F$, are calculated as $||\mu_{GT} - \mu_F||$, where $||\cdot||$ denotes the Euclidian norm. To ensure that we compare the fitted Gaussian with the most appropriate ground truth object without comparing multiple $\mu_F$’s with the same $\mu_{GT}$, we assign the pairs using the Hungarian algorithm [10] implemented in the scikit-learn-library [18]. This algorithm find the parings that globally minimises the overall distance between ground truth and fitted Gaussians. In the left panel of Fig. 2 we see the
distributions of the distances as box plots over the 97 Gaussians of interest. All algorithms perform similar with respect to the median and the range between the first and third quartile. The non-linear TREM and the DLLM have considerably more outliers than the EM-based algorithms, inducing the means of the former being higher compared with the latter.

To include the shape and orientation differences of the fitted Gaussians we will also calculate the Kullback-Leibler (KL) divergences [11] between the fitted Gaussians and the ground truth Gaussians that are displayed in Fig. 1. The KL divergence between two $d$-dimensional normal distributions $p(x)$ and $q(x)$ with respective means (variance) $\mu_p (\Sigma_p)$ and $\mu_q (\Sigma_q)$ is given by [23]:

$$D_{KL}(p||q) = \frac{1}{2} \left[ \log \left( \frac{\det(\Sigma_q)}{\det(\Sigma_p)} \right) - d + [\mu_p - \mu_q]^T \Sigma_q^{-1} [\mu_p - \mu_q] + \text{Tr}(\Sigma_q^{-1} \Sigma_p) \right].$$

We assign the pairs of fitted and ground truth Gaussian using the Hungarian algorithm just as in the centre distance evaluation. For the KL divergence between a fitted and a ground truth Gaussian we assign $\mu_p = \mu_F$ and $\mu_q = \mu_{GT}$, the covariance matrices are assigned accordingly. In the middle panel of Fig. 2 the KL divergence is plotted and the GMM shows the highest divergence compared with ground truth. For the other three algorithms the performance is comparable, but it is worth noticing that again the non-linear approaches have many outliers pulling up their means while not affecting the median. The purpose of

![Fig. 2. Box plots summarising the goodness of fit and segmentation accuracy results of the four different algorithms. In all plots the box edges show the lower to upper quartile of the data with the median as a red line. Notches show the 95% confidence intervals for the median and whiskers indicate 1.5 times the interquartile range. Outliers are depicted as blue crosses and may extend beyond the range shown. Red stars indicate the mean of each distribution. Left: The Euclidian distance between fitted and ground truth Gaussians. Middle: The KL divergence between fitted and ground truth Gaussians. Right: The Jaccard index for segmentation using the different algorithms.](image)

the Gaussian fitting is that it should be used for segmentation and object identification within clusters. Therefore, we also evaluate the segmentation where each pixel is assigned to an object based on the Gaussians. To quantify the segmenta-
tion we calculate the Jaccard index [6] for the segmentation by considering the overlap between each object obtained from ground truth and the best matching object obtained. The Jaccard index is defined as

\[
J_i(A, B) = \frac{A \cup B}{A \cap B},
\]  

(13)

where \(A\) is the set of pixels in each ground truth object and \(B\) the pixel from the best matching object segmented by each algorithm. The matching of segments is again made using the Hungarian algorithm. In the right panel of Fig. 2 we see the Jaccard index for the four algorithms and we see that all the values are within the confidence interval of each other so no certain conclusions can be drawn from this measure.

Besides accuracy the run-time of the algorithm is of importance and we are therefore also presenting the convergence time per image for the four algorithms. Simulations were performed on a 8 core Intel\textsuperscript{TM} Core\textsuperscript{TM} i7-3770 @ 3.40 GHz with 16 Gb RAM running a Linux 3.13.0-53-generic(x86_64) kernel. Both EM-based algorithms, GMM and linear TREM, needs approximately 200 seconds on average to complete the fitting of an image, GMM needs on average 183 s (94 s, 308 s) per image and linear TREM 208 s (111 s, 348 s). The numbers in parenthesis gives the bootstrapped 95% confidence intervals of the mean values presented. DLLM needs on average 3033 s (1786 s, 4792 s) while non-linear TREM needs 5225 s (2693 s, 8586 s). As expected, both non-linear algorithms are at least one order of magnitude slower than the linear ones with non-linear TREM as the slowest. This is not surprising as for the non-linear TREM a
non-linear optimization, albeit in a lower dimensional space than DLLM, is run multiple times as the positions of the Gaussians are updated using Eq. (5).

In Fig. 3 we see an example of an image segmented by the four algorithms as well as the intensity image and the ground truth segmentation. The ground truth segmentation is constructed by placing Gaussians at the object centres and rotating their base covariance matrix, $\Sigma_0$, by the same angle as the original object. The greatest gain of applying TREM or DLLM is demonstrated in the top cluster where two objects are laying almost parallel. The GMM is splitting this cluster in transversally while the template based algorithms succeed in the correct segmentation. Linear TREM is not immune to these type of errors, as can be seen in the faulty split of the cluster at the bottom edge of the image. The non-linear algorithms are able to correctly segment also this cluster, while in other cases the constraint of pure rotation may lead to worse segmentation than the more flexible linear algorithms. For the cluster with four object close to the left edge of the image the non-linear TREM is performing poorly while the other algorithms are able to well capture the true structure. The many outliers seen in Fig. 2 for the non-linear algorithms can be explained by these failures. To investigate if there is a systematic difference between GMM and linear TREM in segmenting clusters with two almost parallel objects we create such pairs. The position and rotation of each object is determined by the parameters $(x_k, y_k, \phi_k)$ where $k \in \{1, 2\}$ indicate the object. We set the rotation of one object to $\mathcal{N}(\phi_1, 0, 0.05)$, e.g. close to zero, and the rotation of the other object is drawn from to $\mathcal{N}(\phi_2, \nu, 0.05)$. We chose $\nu \in \{\pi/6, \pi/8, \pi/10, \pi/12, 0\}$ and draw 30 pairs for each $\nu$, see Fig. 4 for examples of pairs. The centres of the objects are drawn from ($\mathcal{N}(x_1, 20, 2), \mathcal{N}(y_1, 15, 2)$) and ($\mathcal{N}(x_2, 20, 2), \mathcal{N}(y_2, 25, 2)$). In Fig. 4 we have one example cluster for each $\nu$ and how the the two linear algorithms segmented them. We did not consider non-linear TREM or DLLM for this experiment as the results for the synthetic images suggests, see for example Fig.

![Fig. 4. Left: Examples of segmentation of nearly parallel objects by GMM and linear TREM. Middle: Jaccard index for the segmentation of 30 pairs for each orientation difference ($\nu$) by GMM. Middle: Jaccard index for the segmentation of 30 pairs for each orientation difference ($\nu$) by linear TREM.](image-url)
3, that they would likely perform very well but would naturally be one order of magnitude slower. This experiment is designed to test if linear TREM retain the desired property of handling this segmentation task at the same time as its computational efficiency is on the same scale as standard GMM. As a performance measure we consider the Jaccard index and this is shown in the middle (GMM) and right (linear TREM) panels. For $\nu = \pi/6$ both algorithms perform well and approximately equal while for smaller $\nu$ GMM starts failing to obtain the correct segmentation. The object pairs should be split longitudinally, as both algorithms do in the case of $\nu = \pi/6$ and $\nu = \pi/8$ in the segmentation examples of Fig. 4. For smaller angle differences than $\nu = \pi/6$ the GMM is often splitting the cluster transversally, see GMM segmentation for $\nu > \pi/8$ in Fig. 4. For $\nu = 0$ GMM actually splits more clusters transversally than longitudinally as can be deducted from the median Jaccard index that is 0.36, comparable to the Jaccard index of 0.33 expected if splitting the clusters transversally instead of longitudinally. From Fig. 4 it is clear that linear TREM has the property of finding segmentations that agree with the original template at a speed comparable to standard GMM fitting by EM.

5 Discussion

Here we have investigated four different approaches to the problem of finding the individual objects in a cluster of $K$ such objects using generative models and template matching. It is assumed that we know the approximate shape and size of the individual objects, which are not rotationally invariant. The synthetic data is of a similar shape that can be found in real biology, for example *Escherichia coli* [24]. It is naturally of importance that the algorithms are further tested on real data so that the findings about applicability of, in particular, linear TREM can be confirmed. This study has demonstrated that linear TREM is better than traditional GMM in resolving clusters of objects with known shape without any significant loss of efficiency. The EM fitting of a two dimensional GMM with standard update rules regularly take on forms that we know are not present in the images we are analysing. To alleviate this problem we seed the segmentation process with a known shape that we represent by a fixed covariance matrix $\Sigma_0$. By realizing this in three different algorithms, linear and non-linear TREM as well as direct non-linear maximization of the log-likelihood (DLLM), we have examined the effectiveness of the algorithms both in terms of accuracy and speed. The non-linear algorithms had many outliers with poor performance which probably is due to unreliability when confronted with larger clusters and clusters where the binary outline did not match the expected summation of $K$ objects. The latter case is not uncommon in fluorescence microscopy where the signal from multiple object close to each other often superimpose to create a halo around the cluster. For large clusters the dimensionality can become an issue, DLLM has to solve a $3K$-dimensional problem where $K$ is the number of objects in the cluster. The issues with convergence and escape from local minima can possibly be alleviated by simulated annealing but this will further
slow down algorithms that are already one order of magnitude slower than GMM and linear TREM. In the case of applying TREM to 3D data this problem would be attenuated while the linear TREM can easily be adapted. The formulation of linear TREM also means that we would not have to worry about any ambiguities when rotating in three dimensions. In conclusion the linear TREM is a strong candidate for this type of segmentation and cluster analysis as it keeps the flexibility and speed of the GMM while imposing a shape prior of objects.

In this paper we used synthetic data where we know the position and orientation of each object and we can therefore determine the correct number of objects in each cluster. However, as the algorithm is only applicable to approximately rigid objects with a known shape the number of objects in experimental data can normally be well approximated by measuring the cluster area and comparing it to the known area of the individual objects. In this study the focus has been on resolution of binary clusters while the initial segmentation process has been reduced to a simple thresholding. For implementation on real data there would be additional challenges, for example the need to approximate the number of objects in a cluster, but also further cues that could be used. Incorporation of edges or texture information can serve as further feature to either perform a better initial segmentation or for choosing initial values for the Gaussians.

Acknowledgments. This research was funded within the project DropCode supported by Thüringer Aufbaubank.

References