# MODEL BASED PROBABILISTIC PIECEWISE CURVE APPROXIMATION

*Y. Cem Sübakan, Bülent Sankur*

Electrical & Electronics Engineering
Boğaziçi University, 34342 Bebek, Istanbul
{cem.subakan, bulent.sankur}@boun.edu.tr

## ABSTRACT

In this work, we approach the piecewise curve approximation problem with a model-based probabilistic framework. For this purpose, we propose three different models. These models can be used for feature extraction or compression. The first model is a variant of the Bayesian regression model where we can parametrically alter the design matrix. The second model approaches the piecewise curve approximation as a clustering problem. The third model adds temporal connectivity to the second model and combines Hidden Markov models with linear regression. We run the first and the third models on a curve which is used to rank existing algorithms and show that our approaches outperforms its rivals. We also run our models on several real-life curves to show their capabilities.

***Index Terms*—** Piecewise curve approximation; Bayesian modeling; Curve segment clustering; Hidden Markov Models

## 1. INTRODUCTION

Curve modeling or curve approximation is a frequently emerging topic in image processing. It finds various applications in fields like medical imaging, geographical information systems, and computer vision. The goals in curve approximation could be to approximate parsimoniously in order to decrease the the storage size, to extract useful descriptors to be used for detection, classification and information retrieval purposes, or both to compress and describe the curve. For example, in our previous work [1], we used curve approximation approach to extract some useful features to analyze the trajectories of the radio-controlled model helicopters. There exists a plethora of approaches to piecewise curve approximation in the literature, ranging from dynamic programming [2] to tree based coding [3], from Bézier curves [4] to wavelets [5] and graph theory [6].

Bayesian modeling using graphical models is very popular in machine learning due to the modeling flexibility and the availability of various inference and learning algorithms within the framework. There are several works on piecewise curve approximation which employ a Bayesian modeling approach [7, 8]. In these works the problem is viewed as a changepoint detection problem, where changepoints define the region of fitted piecewise polynomials.

In this work, we propose three different probabilistic models: i) Bayesian Piecewise Regression Model, ii) Curve Segment Clustering Model, iii) Hidden Markov Model for Curve Segment Clustering. The first model also approaches the problem as a changepoint detection problem. It is basically an adaptation of the Bayesian linear regression model [9] which uses linear spline functions as basis functions. The positions of the splines are controlled by a latent variable. We use Metropolis-Hastings Markov Chain Monte Carlo (MCMC) algorithm to do inference on this model. The second model approaches the problem as a clustering problem. Given a curve data, the inference algorithm simultaneously clusters the data and performs regression. In this sense, it is a combination of linear regression with the Gaussian mixture models. The third model adds temporal connectivity to the second model to impose spatial sequentiality in learning. In both second and third model we use Expectation Maximization Algorithm for learning.

The paper is organized as follows: In Section 2, we introduce the probabilistic models using the generative model convention and the corresponding directed acyclic graphs. Sample curves generated from the models are given to illustrate these models. The experimental results and comparisons of the models are provided in Section 3.

## 2. METHODOLOGY

Before proceeding with the details of these models, we introduce the Bayesian linear regression notation used in the sequel: Given an input vector $\mathbf{x}$, the output vector $\mathbf{y}$ is generated as follows:

$$\mathbf{y} = \mathbf{\Phi}(\mathbf{x})\mathbf{w} + \epsilon \tag{1}$$

where, $\mathbf{\Phi}(\mathbf{x})$ is our design matrix whose columns are the suitable basis functions for the problem in hand: If we wish to fit polynomials to the data, we may choose the columns as the powers of $\mathbf{x}$. The coefficient vector which enables us to take

the linear combination of the basis functions is given by $\mathbf{w}$ Finally, $\epsilon \sim \mathcal{N}(0, \sigma_y^2 \mathbf{I})$ is the isotropic observation noise.

### 2.1. Model 1 - Bayesian Piecewise Regression

Basically, this model is a variant of the standard linear regression model [9] where we can also parametrically alte the design matrix. The generative model of the Bayesiai piecewise regression model is defined as follows:

$$p(\mathbf{w}_i) = \mathcal{N}(\mathbf{w}_i; 0, \sigma_w^2 \mathbf{I}) \quad (2)$$

$$p(\mathbf{y}_t|\mathbf{W}, \mathbf{p}) = \mathcal{N}(\mathbf{y}_t; \phi_{\mathbf{p}}(t)\mathbf{W}, \sigma_y^2 \mathbf{I}) \quad (3)$$

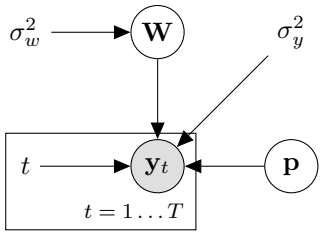



**Fig. 1**. Directed Graph of the Bayesian piecewise regression model

The directed graph of the model is given in Figure 1. The columns of the design matrix $\mathbf{\Phi_p}$ are first order spline functions. The $\mathbf{p}$ vector determines the position of the peaks of the spline functions which in turn determines the breaking points of the generated curve. The construction of the design matrix ensures that we always have a continuous curve. Notice the modifications in $\mathbf{p}$ are coupled between all basis functions in that the peak position of a spline is also the onset position of the next spline, the offset of the previous spline, and as a consequence the design matrix rows sum to a constant. In Figure 2 we show an example design matrix and the resulting generated curve. The combination coefficient vectors $\mathbf{w}_i$ form a matrix, each column for an individual dimension. For example, in the case of planar curves one has simply $\mathbf{W} = [\mathbf{w}_1 \quad \mathbf{w}_2] = [\mathbf{w}_x \quad \mathbf{w}_y]$. The $i$ index is used to index the x,y coordinates of the data and $t$ is the discrete time index where $T$ is the total number of observations and $K$ is the number of segments to be used in the generated curve. The observation noise and the prior coefficient variance are respectively controlled by parameters $\sigma_y^2$ and $\sigma_w^2$.

#### *2.1.1. Inference & Parameter Learning*

It is not possible to find a analytical solution for the posterior distribution of the partition vector $\mathbf{p}$. So we resort to MCMC sampling $\mathbf{p}$. Specifically, we use Metropolis-Hastings sampling. In each iteration, we propose a new partition vector $\mathbf{p}^*$ according to some proposal density $q(\mathbf{p}^*|\mathbf{p}^\tau)$, and then find the corresponding curve to this particular partition vector. We

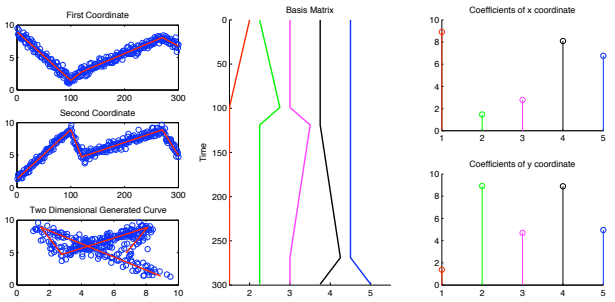



**Fig. 2**. Left column: from top to bottom, the x and y coordinates of the data, and the noisy data (circles) generated according to Eq. 4, mean $\mathbf{\Phi_p(t)W}$ is shown with solid line; Middle column: The design matrix $\mathbf{\Phi_p}$. $\mathbf{p}$ is chosen as [100 120 270], which determine the peaks of the triangles in columns 2, 3, 4; Right column: Linear combination coefficients $\mathbf{w}_i$, of the corresponding basis functions.

accept the partition vector proposed in iteration $\tau$, using the following acceptance probability:

$$A(\mathbf{p}^*, \mathbf{p}^\tau) = \min\left(1, \frac{p(\mathbf{y}_t, \mathbf{W}^*, \mathbf{p}^*) q(\mathbf{p}^\tau|\mathbf{p}^*)}{p(\mathbf{y}_t, \mathbf{W}^\tau, \mathbf{p}^\tau) q(\mathbf{p}^*|\mathbf{p}^\tau)}\right) \quad (4)$$

where, $\mathbf{W}^*$ is the coefficient matrix corresponding to $\mathbf{p}^*$. We find it using,

$$\mathbf{W}^* = \left(\frac{\mathbf{I}}{\sigma_w^2} + \frac{\mathbf{\Phi}_{\mathbf{p}^*}^T \mathbf{\Phi}_{\mathbf{p}^*}}{\sigma_y^2}\right)^{-1} \left(\frac{\mathbf{\Phi}_{\mathbf{p}^*}^T \mathbf{y}}{\sigma_y^2}\right) \quad (5)$$

This way, it is observed that the algorithm converges to a stationary distribution quickly with a random initialization. Generally, choosing an uniform initial $\mathbf{p}$ vector is a good strategy. After having sampled from the posterior distribution of the partition vector, we estimate it via averaging the samples after the burn-in period of the Markov Chain. Finally, note that the construction of the basis matrix makes this model applicable only to data having ordering information, hence it is not amenable to analyze unordered point clouds.

### 2.2. Model 2 - Curve Segment Clustering

The main motivation of this model is to combine the Bayesian linear regression model with Gaussian mixture model. The generative model is defined as follows:

$$p(\mathbf{r}_t) = \prod_{k=1}^{K} \pi_k^{r_{tk}} \quad (6)$$

$$p(\mathbf{w}_k) = \mathcal{N}(\mathbf{w}_k; 0, \sigma_w^2 \mathbf{I}) \quad (7)$$

$$p(y_t|\mathbf{w}_k, \mathbf{r}_t) = \prod_{k=1}^{K} \mathcal{N}(y_t; \phi(x_t)\mathbf{w}_k, \sigma_{y,k}^2)^{r_{tk}} \quad (8)$$

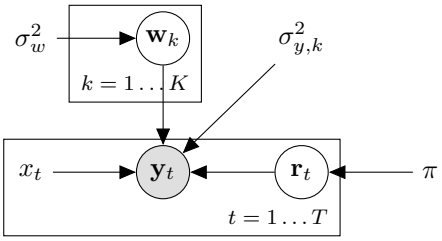



**Fig. 3**. Directed Graph of the line clustering model

The corresponding directed graph is given in Figure 3. Different from the first model, in this model we have $K$ linear combination vectors denoted as $\mathbf{w}_k$. Also, the construction of the design matrix is different. In this model, we take one of the coordinates of the two dimensional curve (e.g., x-coordinate) as the input to our design matrix. This gives us the freedom to treat curve data without ordering information, i.e., point clouds as in the case of edge maps in images to avoid the need for the ordering information of the data, so as to be able to model a point cloud. From the data generating perspective, first each data item indexed by $t$ is assigned to one of the $K$ clusters using the indicator variable $\mathbf{r}_t$ which has only one of its $K$ components set to 1 and the rest to zero. Then, each cluster (curve segment) is generated by multiplying the design matrix $\mathbf{\Phi}$ with the corresponding coefficient vector $\mathbf{w}_k$. The parameters of the model are $\sigma_w^2$, $\sigma_{y,k}^2$ and $\pi$, which respectively denote the prior variance of the coefficient vectors, the observation noise, and the prior cluster assignment probabilities. Notice that, because of the construction of the design matrix, this model is restricted to two dimensional curves. An example generated data is given in Figure 4 .

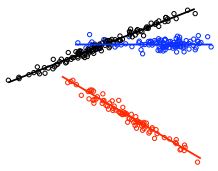


**Fig. 4**. An example data generated from the Curve Segment Clustering Model. The "mean lines" of each cluster, $\mathbf{\Phi}(\mathbf{x}_k)\mathbf{w}_k$ are shown with solid lines and the generated noisy data $\mathbf{y}$ are shown as circles.

### *2.2.1. Parameter Learning*

In order to learn the parameters of the model we use an Expectation-Maximization Algorithm. The latent variable is $\mathbf{r}_t$ and the parameters to be optimized are $\sigma_{y,k}^2$, $\pi_k$ and $\mathbf{w}_k$.

- **E-Step:**
  The posterior distribution of the indicator variable $\mathbf{r}_t$ is as follows:

$$\mathbb{E}_{p(\mathbf{r}_{1:T}|\mathbf{y}_{1:T},\theta)}[r_{tk}] = \frac{\pi_k \mathcal{N}(y_t; \phi(x_t)\mathbf{w}_k, \sigma_{y,k}^2)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(y_t; \phi(x_t)\mathbf{w}_j, \sigma_{y,j}^2)} \tag{9}$$

  where $\theta$ is used as a shorthand for the parameters $\sigma_w^2$, $\sigma_{y,k}^2$ and $\pi$. Note that the posterior distribution of all of the indicator variables $\mathbf{r}_{1:T}$ factorizes over $t$ because of the conditional independence properties of the model which can be seen easily using Figure 3.

- **M-Step:**
  Update equations for the parameters to be optimized are as follows:

$$\mathbf{w}_k^{\text{new}} = \left( \frac{\mathbf{I}}{\sigma_w^2} + \frac{\sum_{t=1}^{T} \mathbb{E}[r_{kt}] \phi^T(x_t)\phi(x_t)}{\sigma_{y,k}^2} \right)^{-1} \times \left( \frac{\sum_{t=1}^{T} \mathbb{E}[r_{tk}] \phi^T(x_t) y_t}{\sigma_{y,k}^2} \right) \tag{10}$$

$$\pi_k^{\text{new}} = \frac{1}{N} \sum_{t=1}^{T} \mathbb{E}[r_{tk}] \tag{11}$$

$$(\sigma_{y,k}^2)^{\text{new}} = \frac{\sum_{t=1}^{T} \mathbb{E}[r_{tk}](y_t - \phi(x_t)\mathbf{w}_k)^2}{\sum_{t=1}^{T} \mathbb{E}[r_{tk}]} \tag{12}$$

## 2.3. Model 3 - Hidden Markov Model for Curve Segment Clustering

One shortcoming of the curve segment clustering model in section 2.2 was that it could not handle non-convex shapes or self-crossing curves. Typical example are trajectories of hand gestures, of handwriting, of acrobatic maneuvers etc. To compensate for this problem, we now add temporal connectivity between the successive indicator variables $\mathbf{r}_t$ so that they form a first order Markov chain. One would conjecture that with the imposition of a chain behavior, the curve model could resolve better sequences where sequentiality is to be preserved. The generative model is specified as follows:

$$p(\mathbf{r}_t) = \prod_{k=1}^{K} \pi_k^{r_{tk}} \tag{13}$$

$$p(\mathbf{r}_t|\mathbf{r}_{t-1}) = \prod_{j=1}^{K} \prod_{k=1}^{K} A_{jk}^{r_{(t-1),j} r_{t,k}} \tag{14}$$

$$p(\mathbf{w}_{ki}) = \mathcal{N}(\mathbf{w}_{ki}; 0, \sigma_w^2) \tag{15}$$

$$p(\mathbf{y}_t|\mathbf{W}_k, \mathbf{r}_t) = \prod_{k=1}^{K} \mathcal{N}(\mathbf{y}_t; \phi(t)\mathbf{W}_k, \sigma_{y,k}^2)^{r_{tk}} \tag{16}$$

The corresponding directed graph is given in Figure 5. $\mathbf{A}$ is the state transition matrix, $\pi$ is the prior distribution for the

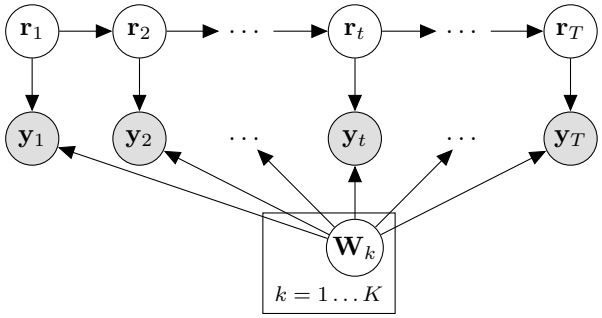



**Fig. 5**. Directed graph of the Hidden Markov Model. Note that the main difference of this model from the second model is the Markovian relationship between the indicator variables $\mathbf{r}_t$.

first state. As in the second model $\sigma^2_{y,k}$ ve $\sigma^2_w$ are the observation noise and the prior coefficient variance respectively. Note that, the columns of the design matrix are formed using the discrete index of the data. Therefore, we use a coefficient matrix as in the first model, but this time we have in total $K$ $\mathbf{W}_k$ matrices for each cluster $k$. Like in the first model $i$ index is for the coordinates of the data. An example data generated from the data is given in Figure 6.

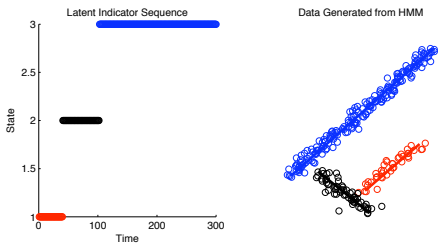



**Fig. 6**. An example data generated from the Hidden Markov Model. Left column:The hidden indicator variable sequence; Right column: The corresponding generated data.

### *2.3.1. Parameter Learning*

We again use Expectation Maximization Algorithm for parameter learning. The latent variable is $\mathbf{r}_{1:T}$. The parameters to be optimized are $\sigma^2_{y,k}$, $A_{jk}$, $\pi_k$ and $\mathbf{W}_k$.

- **E-Step:**
  The sufficient statistics which should be computed in the E-step are as follows:

$$\mathbb{E}_{p(\mathbf{r}|\mathbf{y},\theta)}[r_{tk}] = \gamma(r_{tk}) = p(r_{tk}|\mathbf{y}) \tag{17}$$

$$\mathbb{E}_{p(\mathbf{r}|\mathbf{y},\theta)}[r_{(t-1),j}r_{t,k}] = \xi(r_{(t-1),j}r_{t,k}) = p(r_{(t-1),j}r_{t,k}|\mathbf{y}) \tag{18}$$

  Note that $\gamma$ and $\xi$ are widely used in HMM literature, so we stick to this convention. $\gamma(\mathbf{r}_t)$ ve $\xi(\mathbf{r}_{t-1}, \mathbf{r}_t)$, can be computed efficiently using the forward-backward message passing algorithm. [9].

- **M-Step:**
  The update equations to optimize the parameters are as follows:

$$\mathbf{w}_{ki}^{\text{new}} = \left( \frac{\mathbf{I}}{\sigma_w^2} + \frac{\sum_{t=1}^T \gamma(r_{tk})\phi^T(t)\phi(t)}{\sigma_{y,k}^2} \right)^{-1} \times \left( \frac{\sum_{t=1}^T \gamma(r_{tk})\phi^T(t)y_{ti}}{\sigma_{y,k}^2} \right) \tag{19}$$

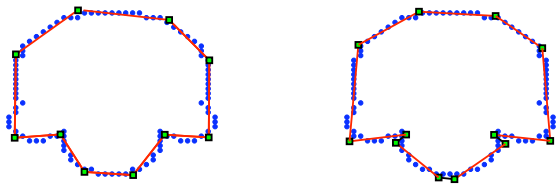
$$A_{jk}^{\text{new}} = \frac{\sum_{t=2}^T \xi(r_{(t-1),j}r_{t,k})}{\sum_{l=1}^K \sum_{t=2}^T \xi(r_{(t-1),j}r_{t,l})} \tag{20}$$

$$\pi_k^{\text{new}} = \frac{\gamma(r_{1k})}{\sum_{j=1}^K \gamma(z_{1k})} \tag{21}$$

$$(\sigma_{y,k}^2)^{\text{new}} = \frac{\sum_{t=1}^T \gamma(r_{tk})(\mathbf{y}_t - \phi(t)\mathbf{W}_k)^T(\mathbf{y}_t - \phi(t)\mathbf{W}_k)}{\sum_{t=1}^T \gamma(r_{tk})} \tag{22}$$

## 3. EXPERIMENTAL RESULTS

In this section, we try our algorithms on the task of approximating the curves with linear segments. First, we run the algorithms on a test curve, which was also used in [10, 11], where the relative performance of 23 different algorithms were compared. We use the same metric as in [11], i.e., the combination of integrated squared error (ISE) and the number of approximating points M. Secondly, we run the algorithms on three different real life curves which are used in [6]. In Figure 7, we show our results using Model 1 and Model 3 on the curve from [10]. We don't show the results obtained with Model 2, because of space constraints and the similarity between Model 2 and 3. It is also observed that Model 3 gives better results than Model 2. We choose the number of lines

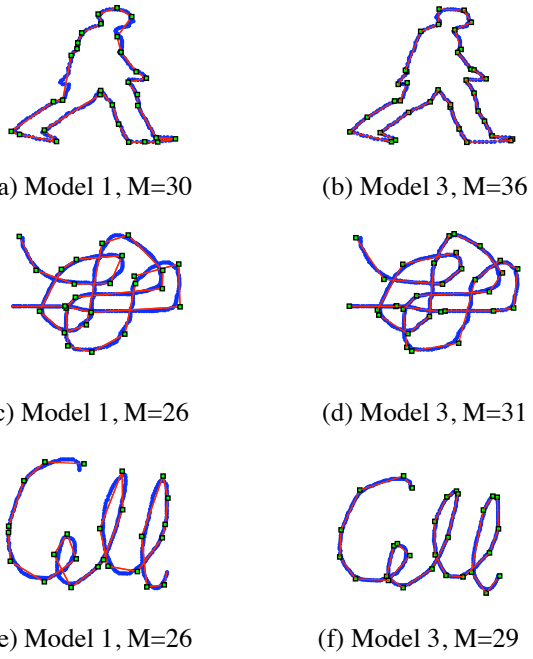


(a) Model 1, ISE:28.86, M=10 (b) Model 3, ISE:24.73, M=12

**Fig. 7**. The results obtained on the curve given in [cite]. The green squares are the points used for approximation.

to be fitted $K$, by running the models within a range of $K$, and then by choosing the best $K$ with respect to the performance metric in [11]. In order to be able to report the number

of points used in Model 3, we resort to a heuristic pruning procedure: The model does not have a continuity constraint between the lines. So, generally, we have to report twice the number of lines as M. However, in most cases, the lines end up to be continuous. What we do is to find the intersection of the consecutive lines and accept it as a representative point if it is nearer to the end points of the consecutive segments than a certain threshold. We depict points that do not qualify as such with a black line between them in Figure 7 and 8. We see that both the first and the third model performs better than the other 23 competitors and even better than the reference algorithm [2], which has reference ISE values 38.9 and 25.9 for M=10,12 respectively. To obtain these results, we arbitrarily initialize the models. In Figure 3, we show the results of Model 1 and Model 3 on the curves used in [6] . We see that we are able to express these curves with a few number of points without losing the general shape information if losing some details. We've seen in these results that, the first model is more parsimonious in terms of points whereas the third model is more loyal to the original curve.



(a) Model 1, M=30 (b) Model 3, M=36

(c) Model 1, M=26 (d) Model 3, M=31

(e) Model 1, M=26 (f) Model 3, M=29

**Fig. 8**. The results obtained on the curves used in [6]. The green squares are the points used for approximation.

## 4. CONCLUSIONS

In this work we propose three different probabilistic models for curve compression using Bayesian modeling framework. We've tried our algorithms on several different curves and see that, our models are competitive with the existing curve analysis algorithms. It may be possible to extend our models to more sophisticated ones, where it is possible to impose prior knowledge on spatial properties of the approximating curves as a future work. Also, for model selection, reversible jump MCMC algorithm can be employed on the first model and different sparsity constraints can be considered on third model to enrich the design matrix, which may enable us to handle complicated shapes with a fewer number of parameters.

## 5. REFERENCES


[1] Y.C. Subakan, O. Agcaoglu, and B. Sankur, "An automatic maneuver assessment system for r/c aerobatic model helicopter competitions," in *Signal Processing and Communications Applications (SIU), 2011 IEEE 19th Conference on*, april 2011, pp. 650 –653.

[2] J.C. Perez and E. Vidal, "Optimum polygonal approximation of digitized curves," *Pattern Recognition Letters*, vol. 15, pp. 743–750, 1994.

[3] X. Huo and J. Chen.;, "Jbeam: multiscale curve coding via beamlets," *IEEE Transactions on Image Processing*, vol. 14, no. 11, pp. 1665–1677, Nov. 2005.

[4] H. Chang and H. Yan, "Vectorization of hand-drawn image using piecewise cubic bézier curves fitting," *Pattern Recognition*, vol. 31, no. 11, pp. 1747–1755, 11 1998.

[5] S. L. Wang, Y. P.; Lee and K. Toraichi, "Multiscale curvature-based shape representation using b-spline wavelets.," *IEEE Transactions on Image Processing*, vol. 8, no. 11, pp. 1586–1592, 1999.

[6] E. Hadju and I. Pitas, "Piecewise linear digital curve representation and compression using graph theory and a line segment alphabet.," *IEEE Transactions on Image Processing*, vol. 17, no. 2, pp. 126–133, 2008.

[7] D. G. T. Denison; B. K. Mallick and A. F. M Smith, "Automatic bayesian curve fitting," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 60, no. 2, pp. 333–350, 1998.

[8] E. Punskaya; C. Andrieu; A. Doucet and W.J. Fitzgerald, "Bayesian curve fitting using mcmc with applications to signal segmentation," *IEEE Transactions on Signal Processing*, vol. 50, no. 3, pp. 747–758, 2002.

[9] C. M. Bishop, *Pattern Recognition and Machine Learning*, vol. 4, Springer, 2006.

[10] C.H. Teh and R.T. Chin, "On the detection of dominant points in digital curves," *Trans. Pattern Analysis and Machine Intelligence*, vol. 11, pp. 859–872, 1989.

[11] P.L. Rosin, "Techniques for assessing polygonal approximations of curves," *IEEE Transactions of Pattern Analysis and Machine Intelligence*, vol. 19, no. 6, pp. 659–666, 1997.