Mixtures of Self-Modeling Regressions

Rhonda VanDyke¹, Kert Viele², Robin Cooper³

(1) Division of Biostatistics and Epidemiology, Cincinnati Children’s Hospital
(2) Department of Statistics, University of Kentucky
(3) Department of Biology, University of Kentucky

Abstract

A shape invariant model for functions \( f_1, \ldots, f_n \) specifies that each individual function \( f_i \) can be related to a common shape function \( g \) through the relation \( f_i(x) = a_i g(c_i x + d_i) + b_i \).

We consider a mixture model that allows multiple shape functions \( g_1, \ldots, g_K \), where each \( f_i \) is a shape invariant transformation of one of those \( g_k \). We derive an MCMC algorithm for fitting the model using Bayesian Adaptive Regression Splines (BARS) and discuss some of the computational difficulties that arise. The method is illustrated using synaptic transmission data, where the groups of functions may indicate different active zones in a synapse.

1 Introduction

Self-Modeling regressions (Lawton et. al 1972, Kneip and Gasser 1988) form a class of models for functional data observed for many individuals. We observe data \( Y_{ij} = f_i(x_{ij}) + \epsilon_{ij} \), where the subject specific functions \( f_i(x) \) have the form \( f_i(x) = h(g, \theta_i)(x) \) where \( g \) is a base function and \( \theta_i \) is a subject specific parameter which specifies a transformation connecting \( g \) and the functions \( f_i \). A variety of choices is available for the transformation class which unites the subject specific functions \( f_i \) to \( g \). One of first examples in the literature, which we will address here, is the shape invariant model (SIM, Lawton et. al 1972), where \( \theta_i = (a_i, b_i, c_i, d_i) \) are called self-modeling coefficients and

\[
f_i(x) = a_i g(c_i x + d_i) + b_i
\]

The self-modeling coefficients result in \( f_i \) being an affine transformation of \( g \) in the x and y axes. Other types of transformations may be found in Telesca and Inoue (2008) who use a Bayesian warping function method. Altman and Villarreal (2004) have developed the SIM model where interest centers on variation in the self-modeling coefficients, using nonlinear mixed effects (Lindstrom and Bates 1990) and p-splines to model the \( \theta_i \) parameters.

Our interest here is

1. We wish to consider mixtures of shape invariant models (MSIMs) where there is more than one underlying shape function \( g \). Unknown to the investigator, suppose the subject specific functions may be grouped so that one group is defined by shape invariant transformations of an underlying \( g_1 \), another group is defined by shape invariant transformation of an underlying \( g_2 \), and so on. A motivating example follows below.
2. Utilize Bayesian Adaptive Regression Splines (BARS, DiMatteo et. al 2001, Behseta et al. 2005a, 2005b) to estimate the underlying shape functions $g_k$. BARS has been shown to provide parsimonious fits (e.g. fewer knots) of complicated functions. Several computational challenges arise in this endeavor which are described in section 3.

The data in Figure 1 provide our motivation for this problem. Figure 1 shows data obtained from a recording of synaptic transmission at a neuromuscular junction (NMJ) from a crayfish. The motor nerve is repeatedly stimulated and the synaptic response is monitored as described in Viele, et. al. (2006). With low frequency stimulation of this low output NMJ, occasionally an evoked response is observed. In the sample shown 65 events occurred in 1000 stimulations. After some data cleaning (see Lancaster et. al 2007) our data of interest is the 65 functions describing voltage over time.

These functions vary in terms of their heights, widths, and peak locations, but in fact do all have similar shapes. Figure 2 shows the “aligned curves” which result from fitting a SIM model to all the functions. The details of this fit, which simply assumed known knots for a spline fit to $g$, is described in Viele et. al (2006). This fit accounts for over 96% of the variation in the dataset.

Biologically, the variation between the functions is assumed to occur because of aspects related to the physiology and structure of the synapses at the NMJ. The electrical signal in the neuron is translated into a vesicle fusing with the presynaptic membrane at active zones to release neurotransmitters. The postsynaptic receptors on the muscle detect the transmitter which results in a depolarization of the muscle fiber. There are various possibilities for the variation in quantal responses. Some of those possibilities involve the location where neurotransmitter is released and such variation would result in two more groups of functions appearing in the data. Thus, finding groups of functions in the data may indicate multiple sites active in the synapse.

In searching for different sites, we wish to find groups of functions within the 65 functions in Figure 1. This is our motivation for fitting a mixture of shape invariant functions, where the different underlying shape functions may indicate different active zones. Previous work in this area has uncovered groupings based on functionals of the traces, for example by fitting a normal mixture model on the peak amplitudes (Viele et. al 2003). Of course, individual functionals only utilize a small amount of the information available in the entire voltage function. It would also be possible to fit a single shape function $g$ and then fit a mixture model over the parameters $(a, b, c, d)$, which we do not pursue here.

A brief outline of this paper is as follows. In section 2, we introduce a mixture of shape invariant models and describe our computational techniques, including necessary adjustments to the BARS algorithm. In section 4 we describe methods for evaluating the results (this is nontrivial due to identifiability issues present in shape invariant models in general) and simulations used to test the methods. In section 5 we discuss the results of the simulations and the analysis of the data in Figure 1. Finally, in section 6 we provide a discussion of the results.

2 Mixtures of Shape Invariant Models

2.1 Prior and Likelihood Structure

Suppose we observe $I$ individuals and for each individual we observe data pairs $(x_{ij}, y_{ij})$ for $i = 1, \ldots, I$ and $j = 1, \ldots, J_i$. We also assume there exist underlying shape functions $g_1, \ldots, g_K$ such that the $i^{th}$ individual follows shape $g_k$ with probability $p_k$, where $p = (p_1, \ldots, p_K)$ is a vector of
probabilities. Let \( z_i \) be a group indicator where \( z_i = k \) indicates the \( i^{th} \) individual follows the \( k^{th} \) function and \( \theta_i = (a_i, b_i, c_i, d_i) \) be the self-modeling coefficients for the \( i^{th} \) individual. Finally \( \sigma^2 \) is the error variance. Where possible, we place conditionally conjugate priors on the parameters to ensure an easier MCMC implementation. In what follows \( g \) refers to the collection of \((g_1, \ldots, g_k)\), \( \theta \) refers to the collection of \( \theta_1, \ldots, \theta_I \), and \( N(\mu, \sigma^2)(x) \) refers to a normal density with mean \( \mu \) and variance \( \sigma^2 \) evaluated at \( x \), with similar notation for other densities. The general prior and likelihood structure is below. We will describe the prior on the \( \theta_i \) and \( g_k \) separately.

\[
\pi(\theta, g, \sigma^2, p) = \left[ \prod_i \pi(\theta_i) \right] \left[ \prod_k \text{BARS}(g_k) \right] \pi(\sigma^2) \pi(p)
\]

\[
\pi(\sigma^2) = \text{InvGamma}(a_\sigma, b_\sigma)(\sigma^2)
\]

\[
\pi(p) = \text{Dir}(a_1, \ldots, a_K)(p_1, \ldots, p_K)
\]

\[
z_i|\theta, g, \sigma^2, p \sim \text{Multinomial}(1, p_1, \ldots, p_K)
\]

\[
f_i(x) = a_ig_z_i(c_ix + d_i) + b_i
\]

\[
Y_{ij}|x_{ij}, z_i, \theta, g, \sigma^2, p \sim N(f_i(x_{ij}), \sigma^2)
\]

Our inferential goals are to estimate the underlying shape functions \( g_1, \ldots, g_K \), the self-modeling coefficients \( (a_i, b_i, c_i, d_i) = \theta_i \) for each trace \( f_i \), and the posterior probabilities each individual trace belongs to each group.

2.1.1 Priors on \( g_k \)

We utilize Bayesian Adaptive Regression Splines (DiMatteo et. al 2001, Behseta et al. 2005a, 2005b) to estimate each of the \( g_k \) functions. Each \( g_k \) has a B-spline basis expansion

\[
g_k(x) = \sum_{r=1}^{R_k+3} \beta_kb_r(x)
\]

where \( b_1, \ldots, b_{R_k+3} \) form a B-spline basis with knots \( \xi_1, \ldots, \xi_{R_k} \) and spline coefficient vector \( \beta \). BARS utilizes a reversible jump MCMC algorithm which samples over the knots in a B-spline used to estimate the function of interest. Without loss of generality, assume the function of interest (for us one of the \( g_k \) functions) is supported over the unit interval \((0, 1)\). A B-spline basis is formed using a vector of knots \( \xi \). The prior on this vector assumes the number of knots follows a \( \text{Poi}(\lambda) \) distribution (here we use \( \lambda = 20 \)) and that the individual knots are uniformly distributed throughout \((0, 1)\). BARS proceeds by iteratively adding, removing, and relocating knots. These changes in the knot structure are accepted or rejected according to BIC (BIC uses the maximum likelihood estimates of the spline coefficients). As BIC is used, this is very close asymptotically to using a unit-information prior (Kass and Wasserman 1995) on the spline coefficients \( \beta \). This structure is what is intended by the notation \( \text{BARS}(g_k) \) used above.
2.1.2 Priors on the self-modeling coefficients

Finally, for estimating the self-modeling coefficients \( \theta_i = (a_i, b_i, c_i, d_i) \), we place a vague bivariate normal prior on \( a_i \) and \( b_i \) (these parameters act as regression coefficients). In general, there is no conditionally conjugate prior for \( c_i \) and \( d_i \). For functions with a single dominant peak, a transformation of \( c_i \) and \( d_i \) to a location-scale family is useful.

Suppose for example the underlying shape function \( g \) has a peak at \( x_g^* \). The individual function \( f_i \) would have that peak at the \( x \) where \( c_i x + d_i = x_g^* \), which is \( x = (x_g^* - d_i)/c_i \). Note that changes in either \( c_i \) or \( d_i \) result in changes to the peak location of \( f_i \). As proper alignment of the peak is very important to achieving a high likelihood, this induces a correlation in the posterior distribution of \( c_i \) and \( d_i \). We have found better results (in terms of the mixing properties of the MCMC chain) by transforming to \( c_i \) (which controls the “spread” of the curve) and \( m_i = (x_g^* - d_i)/c_i \), which represents the location of the peak. Thus, instead of \( f_i(x) = a_i g(c_i x + d_i) + b_i \), we use the transformation \( f_i(x) = a_i g(c_i (x - m_i) + x_g^*) + b_i \).

To see the benefit of this transformation by example, let \( g \) be a triangular “tent function” where \( g(x) = 0 \) outside \((0.2, 0.6)\), \( g(x) \) increases linearly from 0 to 1 over \((0.2, 0.4)\), and \( g(x) \) decreases linearly from 1 to 0 over \((0.4, 0.6)\). We generated data using \( f(x) = g(0.5x - 0.25) \) for \( x = (0, 0.01, 0.02, \ldots, 2) \) and \( N(0, 0.1^2) \) error. The peak of \( g \) occurs at 0.4 and the corresponding peak of \( f \) occurs at \( m = 1.3 \). Keeping everything fixed at the true value except \( c \) and \( d \), the left pane of Figure 3 shows the loglikelihood over \((c, d)\). As can be seen, there is a distinct correlation between \( c \) and \( d \) which limits the mixing of the MCMC chain. In contrast, the right pane shows the loglikelihood as a function of \( c \) and \( m \), which provides a much better structure for MCMC.

Of course, this transformation is limited to \( g \) with a single peak, which is not always true. Thus, this transformation will not produce an improvement in general, or other transformations may be useful in other contexts. For our synaptic transmission data, we have found this transformation to be quite useful.

Thus, for data with a dominant peak

\[
\pi(a_i, b_i, c_i, m_i) = [N_2(\mu_{a,b}, \Sigma_{a,b})(a_i, b_i)] \left[ \text{Gamma}(\alpha_c, \beta_c)(c_i) \right] \left[ \text{Uni}(0, 1)(m_i) \right]
\]

2.2 Identifiability

Identifiability is a pervasive issue in self-modeling regressions (see Altman and Villarreal 2004, Kneip and Gasser 1988). Suppose for example we have a particular underlying shape function \( g(x) \) and a set of coefficients \((a, b, c, d)\) producing \( f(x) = ag(cx + d) + b \). One could achieve the same \( f \) by taking \( g'(x) = 2g(x) \) and taking \( a' = a/2 \). Generally, one can take any self-modeling transformation of \( g \) and then adjust \((a, b, c, d)\) appropriately to produce the same \( f \).

In the original Lawton et. al (1972) paper, this problem was addressed by forcing \( g \) to have particular properties (such as a maximum of 1 and minimum of 0). However, in the Bayesian formulation such constraints would require any estimation method to be significantly altered. BARS, which we utilize here, is not obviously adjustable to handle constraints on the function.

An alternative method of forcing identifiability is to pick an individual trace \( f_i \) and force its corresponding \((a, b, c, d)\) to be \((1, 0, 1, 0)\). This method is utilized in Viele et. al (2006). This forces the scaling on \( g \) to follow the chosen trace but allows BARS to run unconstrained. The central theme here is that in the self-modeling paradigm you are really estimating relative shifts and scalings between the traces.
Unfortunately, in the mixture setup this is not feasible, as there are multiple shape functions. Since we do not know apriori which traces belong to which group (e.g. which are self-modeling versions of $g_1$, which are self-modeling versions of $g_2$, etc.) we cannot select traces to use as “anchors” in each group. Thus the simplest option seems to be to leave a nonidentified model. Altman and Villarreal (2004) also use an unconstrained formulation without difficulty, though from a frequentist standpoint.

Note, however, that our central inferential goals are the assignment of traces to groups, the estimation of the individual traces, and the underlying shape of the $g_1$ and $g_2$ functions. The first two are identifiable, and $g_1$ and $g_2$ are identifiable up to self modeling versions. As with any mixture model, there is a secondary identifiability problem in that the component labels may be switched, but component label switching has not been an issue in the analyses we performed.

3 Implementation

We implemented the model in Section 2 using an MCMC scheme. Some parameters are of course easier to update than others.

3.1 Sampling Scheme

3.1.1 $\sigma^2$ and $p$

Both $\sigma^2$ and $p$ have straightforward conjugate priors conditional on the rest of the parameters.

$$\pi(p|\text{rest}) = \text{Dir} \left( a_1 + (\#Z_i = 1), \ldots, a_k + (\#Z_i = K) \right) (p_1, \ldots, p_K)$$

$$\pi(\sigma^2|\text{rest}) = \text{IGamma} \left( \alpha_{\sigma} + \frac{N}{2}, \beta_{\sigma} + \sum_{i,j} e_{ij}^2 / 2 \right)$$

where

$$e_{ij}^2 = (y_{ij} - \hat{y}_{ij})^2 \quad N = \sum_i J_i$$

3.1.2 Shape functions $g_1, \ldots, g_k$

Conditional on the remaining variables, each $g_k$ is conditionally independent, with each $g_k$ being updated separately using only those data functions with $z_i = k$.

Each $g_k$ is defined by its corresponding knot set $\xi_k$ and spline coefficients $\beta_k$. In the original BARS algorithm from DiMatteo et al (2001), updating was done entirely on $\xi$ using BIC with the maximum likelihood estimate of $\beta$. While we update $\xi$ using the BARS updating, we also draw $\beta|\xi$. Thus, we draw $\xi, \beta|\text{rest}$ via $\xi|\text{rest}$ and $\beta|\xi,\text{rest}$. Our reason for drawing $\beta$ values is that, unlike DiMatteo et al (2001), we have additional parameters (specifically $z_i$ and $\theta_i$) that are easier to simulate if $\beta$ is fixed.
3.1.3 Updating \( Z_i \)

Updating each \( z_i \) also requires moving functions between groups, as there are subject specific parameters \( \theta_i \) for each variable. In the MCMC scheme one must take into account that \((a,b,c,d)\) have different interpretations depending on which group the function belongs to (e.g. the value of \( z_i \)). If \( g_1 \) and \( g_2 \) have different peaks, for example, then a perfect fit for \( g_1 \) would require a much different value of \( a \) than a perfect fit for \( g_2 \).

Ideally, we would like to avoid this problem and integrate the self-modeling coefficients out to find

\[
Pr(Z_i = k|g_1, \ldots, g_K, p, \sigma^2) = \int Pr(Z_i = k|g_1, \ldots, g_k, p, \sigma^2, \theta_i)\pi(\theta_i)d\theta_i
\]

Unfortunately, it is not computationally efficient to find this integral nor is it efficient to approximate it (for example finding the MLE of \( \theta_i \) would require numerical maximization). Thus in our sampler we utilize \( \theta_{ik} \) (different \( \theta_i \) for each \( k \)). To draw a new \( Z_i \), note

\[
Pr(Z_i = k|g_1, \ldots, g_K, p, \sigma^2, \theta) = \frac{p_k \prod_j N(f_{ik}(x_{ij}), \sigma^2)(y_{ij})}{\sum_{w=1}^K p_w \prod_j N(f_{iw}(x_{ij}), \sigma^2)(y_{ij})}
\]

where

\[
f_{ik}(x_{ij}) = a_{ik}g_k(c_{ik}x_{ij} + d_{ik}) + b_{ik}
\]

3.1.4 Updating the Self-Modeling coefficients

Finally, the self-modeling coefficients \((a,b)\) (recall in the sampler we have separate self-modeling coefficients \( \theta_{ik} \) for each \( i \) and \( k \)) are updated straightforwardly, as they are regression parameters with conjugate priors. The \((c,m)\) coefficients are updated using a Metropolis-Hastings random walk scheme.

3.2 Singularities in the BARS updating

In the original BARS implementation, proposed knot sets which resulted in singular design matrices (loosely, too many knots in an area with too few data points), these proposals were discarded, which may be viewed simply as a data dependent prior. In the self-modeling setup, there is an additional complication. In one iteration we may achieve a nonsingular design matrix, but as the self-modeling coefficients \( \theta_i \) are adjusted, this moves the \( x_{ij}^* \) and may result in singularity in the design matrix. We additionally reject these moves out of hand in the sampler. At present, this has been a limited problem, but seems an unavoidable problem as long as BIC is used in the BARS steps.

3.3 Starting Values

To initialize the chain, all traces were first aligned to the largest trace (the trace with the largest difference between the maximum and minimum values) using a Procrustes registration (Ramsay and Li 1998). We then computed the mean square error between each aligned trace and the largest trace. The half of the traces with the smallest MSE values were placed in one group with the half of the traces with the largest MSE values were placed in the second group. Essentially, this means the
half of the traces closest in shape (not necessarily scale) to the largest trace were placed together in a group. This initialized the $Z_i$ values.

To initialize $g_1$ and $g_2$, we fit a spline to the traces in each group (we simply assume 9 equally spaced knots initially, which is then adjusted by the sampler). The initial spline coefficients are then computed by aligning each individual trace to its corresponding $g_z$, and finally the error variance is computed by taking the MSE of the entire fit.

4 Assessing the results

At the end of the MCMC run we have a set of $M$ iterations. For each $m = 1, \ldots, M$, the $m^{th}$ iteration contains functions $g_1^{(m)}, \ldots, g_k^{(m)}$ (expressed in a spline basis), mixing proportions $p^{(m)}$, an error standard deviation $\sigma^{(m)}$, and component assignments $z_1^{(m)}, \ldots, z_I^{(m)}$ for each individual function. Finally, for each $i$ we have self-modeling coefficients $b_i^{(m)}$.

We assess the results in terms of the identifiable pieces of the sampler.

1. We directly have the mixing proportions $p$ between the different functions $g_k$. We also compute the estimated posterior probability each individual trace belongs to each group by finding the observed proportion of times $z_i^{(m)} = k$ for each $k$. Thus, we acquire overall proportions of functions in each group and the estimated posterior probabilities each individual function belongs to each individual $g_k$.

2. To assess the fit of the individual functions, we computed the estimated $f_i(t)$ by constructing for each iteration and each trace

$$ \hat{f}_i^{(m)}(t) = a_i^{(m)} g_i z_i^{(m)} \left( c_i^{(m)} t + d_i^{(m)} \right) + b_i^{(m)} $$

$$ \bar{f}_i(t) = \frac{1}{M} \sum_{m=1}^{M} \hat{f}_i^{(m)}(t) $$

In the plots that are constructed below, we plot a selection of the $\hat{f}_i^{(m)}(t)$ functions (iterations $m = 100, 200, \ldots$) and the overall average $\bar{f}_i(t)$.

3. To estimate each $g_k$ we must remember that each $g_k$ is allowed to drift in the sampler, and thus the resulting $g_k^{(m)}$ iterations must be aligned in shape before producing an estimate. Thus, we begin by taking the functions $g_k^{(m)}$ across the iterations and aligning them to each other. This was done via a Procrustes registration as in Viele et. al (2006). The base shape function of this alignment was used as the estimate for $g_k$.

5 Results

5.1 Simulated Data

We simulated data to identify how well the algorithm discriminated two separate, but similar functions. In each sample dataset we used
These functions are similar to some parametric models used in the biology literature for EPSPs, and are quite similar in shape. Figure 4 shows the two underlying functions. The two functions are similar in shape, but they are clearly not identical. The function $g_2$ is not differentiable at $x = 0$ (note self-modeling versions with a slight shift are quite similar, so this cusp could appear at a different $x$ in the estimate, see the discussion of identifiability), which creates a point where multiple knots are desirable. This is a situation where adaptive knot splines are more effective than using known knots.

Each sample dataset consisted of 100 functions, 50 generated from self-modeling versions of $g_1$ and 50 from self-modeling versions of $g_2$. The self-modeling coefficients $(a, b, c, d)$ were randomly chosen for each function in each simulation to provide a range of overlapping functions.

Forty simulated datasets of 100 functions were generated, 20 of which had an error standard deviation of $\sigma = 0.15$ while the remaining 20 had an error standard deviation of $\sigma = 0.05$. One sample dataset (for $\sigma = 0.05$) is shown in Figure 5. Figure 6 shows the same dataset color coded by the appropriate $g$ function. It is certainly not obvious from the data alone which $g$ is appropriate.

The MCMC algorithm from section 3 was run on each simulated dataset. The sampler was run for 80,000 iterations with the first 20,000 removed as burnin (this is perhaps excessive, but burn-in is long in this context and we were quite conservative). For each simulated dataset we computed the assessable quantities described in section 4 ($g_1$ and $g_2$ estimated up to shape invariance, the individual functions $f_i$, and the probabilities of correct classification).

Figure 7 shows the estimated $g_1$ and $g_2$ plotted against the group assignments (for plotting, each function was assigned to the group that functions spent the most time assigned to in the MCMC run) for the simulated dataset from Figure 5. The diagonal panes in Figure 7 show the functions (aligned) assigned to group 1 plotted against the fitted $g_1$ and the functions assigned to group 2 (aligned) plotted against the fitted $g_2$. The off-diagonal panes show each $g_k$ plotted against the functions which were NOT assigned to that group.

Overall, correct classification rates for the functions were quite high across the simulations. Of the 20 datasets generated with $\sigma = 0.05$, all but 2 had all 100 functions classified correctly. For the other 2 datasets, one had 99 functions classified correctly and the other had 94 functions classified correctly. For the 20 datasets with $\sigma = 0.15$, there was more variability because of the increased noise but there were still high correct classification rates, ranging from a low of 76 classified correctly to a high of 99. Of the 20 datasets with $\sigma = 0.15$, 15 classified 90 or more functions correctly.

5.2 Synaptic Transmission Data

The 61 firings from Figure 1 were analyzed using the algorithm from section 3, using two groups. The sampler was run for 80,000 iterations, with the first 20,000 being removed as burn-in (this is perhaps excessive, but we find burn-in rates to be rather long in this context).

Of the 61 firings, 31 were classified to group 1 over 50% of the time while the other 30 were classified to group 2 over 50% of the time (to make sure the algorithm simply didn’t split the 61
functions in half arbitrarily, we also fit subsets of the functions and found consistent assignments). Figure 8 is identical to Figure 1 except the functions are color coded into group 1 (red) and group 2 (blue).

There is no obvious pattern to the assignments in Figure 8 (e.g. group 1 traces are not generally higher, wider, etc.). When we construct the plot analogous to the aligned data in Figure 2, we do see a pattern emerge. In Figure 9, the central difference in the groups appears near the aligned peaks. Group 1 (in red) shows a quick increase, followed by a slower decrease after the peak. In contrast, group 2 (in blue) has a slower increase to the peak, but a quicker decrease from the peak.

Figure 10 shows the individual estimates of $g_1$ and $g_2$ plotted against the functions assigned (here meaning greater than 50% posterior probability) to each group. Thus the diagonal panels in this figure correspond have good fits (e.g. the functions assigned to group 1 with $g_1$ overlaid, and the functions assigned to group 2 with $g_2$ overlaid), with the off diagonal panels show the differences between the groups (e.g. the function assigned to group 1 with $g_2$ overlaid, and the functions assigned to group 2 with $g_1$ overlaid). Again, the two different shapes are apparent in the graph.

6 Discussion

Fitting a mixture of shape invariant models allows for multiple shapes to be present in a sample of functions. In substantive contexts like the synaptic transmission data, finding these groups has a direct impact on questions like the number of active zones in a synapse. The algorithm proposed in this paper behaves well in simulations and provides results for the synaptic transmission data consistent with biological expectations.

One outstanding issue still to be addressed is inferentially determining the number of groups present in the data. Substantively, the finding of groups is expected, as such a finding can be directly correlated to the underlying structural entities which are responsible for efficacy in chemical synaptic transmission in general (Cooper et al. 1995a,b, Viele 2003). However, if one fits a model with two groups, then two groups will appear in the results, thus in the future it will be important to develop an inferential procedure. Fortunately for the synaptic transmission data, simpler mixture models have also detected two groups (Viele 2003).

In univariate Gaussian mixture models, there are variety of model selection criteria that could be supplied (Steele and Raftery 2009, Charnigo and Sun 2004) or one could use reversible jump samplers to move between models with differing numbers of components, similar to Richardson and Green (1997). In our context it is unclear how to implement a model selection criteria which requires the number of parameters in the model, as the spline formulation includes the number of knots as one of the unknown parameters. Even simplistic criteria such as investigating the mean square errors between functions, which should be similar to $\chi^2$ or noncentral $\chi^2$ distributions, is complicated by the differing scaling produced by the self-modeling coefficients for each function.

Acknowledgements

This research was supported by NCRR(NIH) Grant P20 RR16481.
7 References


Figure 1: A set of 65 evoked excitatory post-synaptic potentials (EPSPs). These data describe the evoked voltage responses observed at the crayfish neuromuscular junction from 1000 stimulation trials. While the functions vary in height, width, and peak locations, all the functions have a similar underlying shape.
Figure 2: The Aligned Firings. These are the same data from Figure 1 aligned to each other
Figure 3: The likelihood structure of $(c,d)$ (shift-scale) compared to the $(c,m)$ (location-scale) transformation. For a function with a single dominant peak, the location-scale transformation has a much lower correlation between $c$ and $m$ and thus the corresponding MCMC algorithm has superior mixing properties.
Figure 4: Two shapes used for simulations. These two functions are aligned versions of $g_1(t) = 0.2 \exp\{-20(x - 0.2)^2\}$ and $g_2(t) = \exp(-7x) - \exp(-12x)$. They are similar, but certainly NOT identical, in shape. The simulations consisted of generating 40 datasets, each with 100 functions, 50 of which are self-modeling versions of $g_1$ with error and 50 of which are self-modeling versions of $g_2$ with error. Note also $g_2$ is not differentiable at the initial rise which creates a situation where multiple knots are desirable in that region.
Figure 5: A sample of 100 simulated functions. There are 50 self-modeling versions of each of $g_1$ and $g_2$ mixed into the dataset. The inferential problem is to separate the traces and simultaneously estimate the underlying shape functions.
Figure 6: The same dataset from Figure 5, but colored according to the underlying groups (unknown in practice). The different groups are simulated so that they do not separate in groups based on height, width, or other functionals, but only by shape.
Figure 7: Results for simulated dataset. The four panels show the functions assigned to each group versus the fitted curves for $g_1$ and $g_2$. The diagonal panes show the functions assigned the corresponding $g$ (e.g. the functions assigned to group 1 with $g_1$ overlaid, and the functions assigned to group 2 with $g_2$ overlaid). In contrast, the off diagonal panes correspond to the functions overlaid with the “other” $g$ (so the functions assigned to group 1 with $g_2$ overlaid, and the functions assigned to group 2 with $g_1$ overlaid).
Figure 8: The synaptic transmission data from Figure 1, color coded into the groups identified by the Mixture of Self-Modeling regressions algorithm (a function was classified in a group if more than 50% of the MCMC iterations placed it in that group). These groups do not appear to separate by peak amplitude, latency, or other commonly used functionals.
Figure 9: The synaptic transmission data from Figure 2, color coded into the groups identified by the Mixture of Self-Modeling regressions model. The distinction between the two groups appears to be in the rate of increase in the rise and the rate of decrease in the descent. Functions in group 1 (red) have a faster rise and slower descent, while functions in group 2 (blue) have a slower rise and faster descent.
Figure 10: Four panels showing the functions assigned to each group versus the fitted curves for $g_1$ and $g_2$. The diagonal panes show the functions assigned the corresponding $g$ (e.g. the functions assigned to group 1 with $g_1$ overlaid, and the functions assigned to group 2 with $g_2$ overlaid). In contrast, the off diagonal panes correspond to the functions overlaid with the “other” $g$ (so the functions assigned to group 1 with $g_2$ overlaid, and the functions assigned to group 2 with $g_1$ overlaid). This illustrates the differences in structure between $g_1$ and $g_2$. 