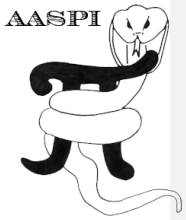
VOLUMETRIC CLASSIFICATION USING GAUSSIAN MIXTURE MODELS – Program gmm3d



Overview

Visual examination of seismic facies on large 3-D seismic data sets where there is little a priori geologic information can be time consuming and inaccurate. The process can be more automated and improved through the use of machine learning. By teaching a computer how to recognize patterns, features can automatically be picked and/or statistically interpreted. This has the obvious benefit of quicker interpretations, but moreover it can highlight features that might otherwise go unnoticed. The Gaussian mixture model provides a flexible framework by which to accomplish this.

A Gaussian mixture model is a statistical construct that attempts to model the probability density of a set of data and can also act as a clustering technique to generate unsupervised seismic facies. The **gmm3d** module is mostly concerned with the latter. Gaussian mixture models can be considered a probabilistic formulation of the *k-means* algorithm where the means of each cluster are also given a covariance. In fact, the **kmeans3d** algorithm is internal to **gmm3d** where the centroids from **kmeans3d** are used as initial estimates for the means of each Gaussian component. Gaussian mixture models are based on probability theory, and by construction provide a posterior probability that any particular voxel belongs to a given cluster (also called mixture decomposition). Moreover, Gaussian mixture models provides a more formal way of finding the number of clusters within a dataset such as the Bayesian information criterion.

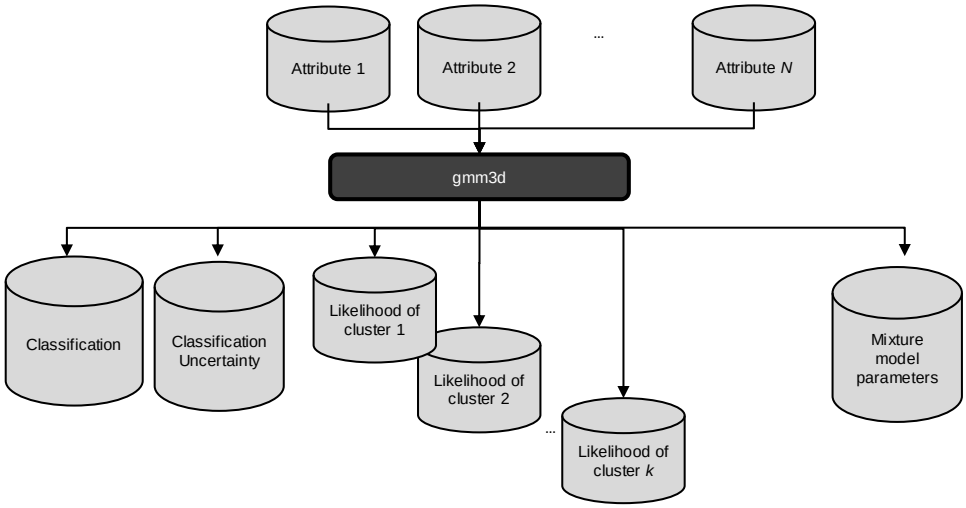
Model based classification comes with a couple assumptions. First, is that the dataset comes from a mixture of multivariate Gaussian distributions. In other words, the clusters are assumed to be shaped like a multivariate Gaussian distribution. Secondly, the classification approach implies that a voxel is produced by a single cluster, or Gaussian distribution.

A Gaussian mixture model, like *k*-means, has no explicit relationship between the cluster numbering and the proximity of one cluster to another. This lack of organization can result in similar facies appearing in totally different colors, and confusing the interpretation if the color bar is not given careful consideration.

Computation Flow Chart

The input to program **gmm3d** provides a means to assign voxels at each vector representing *N* attributes into *k* user-defined (or auto-defined) clusters.

Commented [K1]: SEG style. All scalars are italic. All vectors and matrices are bold, no italic.



The mixture model parameters are in the form of comma-delimited .csv file called gmm_parms.csv and contains the means, covariances, and weights of the mixture model. The classification volume consists of integer numbers that correspond to the cluster number, and for each classification there is a corresponding uncertainty in the uncertainty volume. The likelihood of class 1…k are seismic attribute volumes that contain the mixture decomposition of the model and describe how likely voxels belong to each individual cluster.

Commented [K2]: Capture images of your GUI and describe each parameter. Use program dip3d as a template.

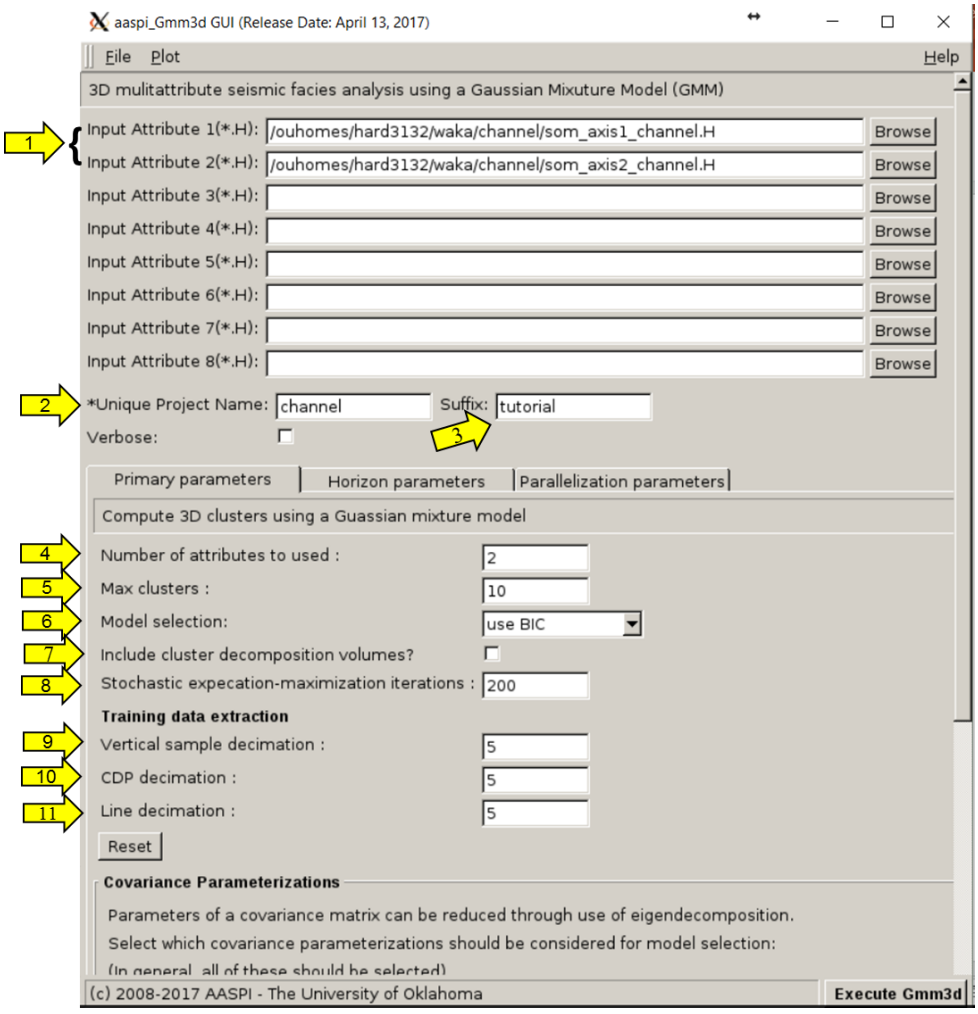
Commented [K3R2]:

Volumetric Classification: Program **gmm3d**

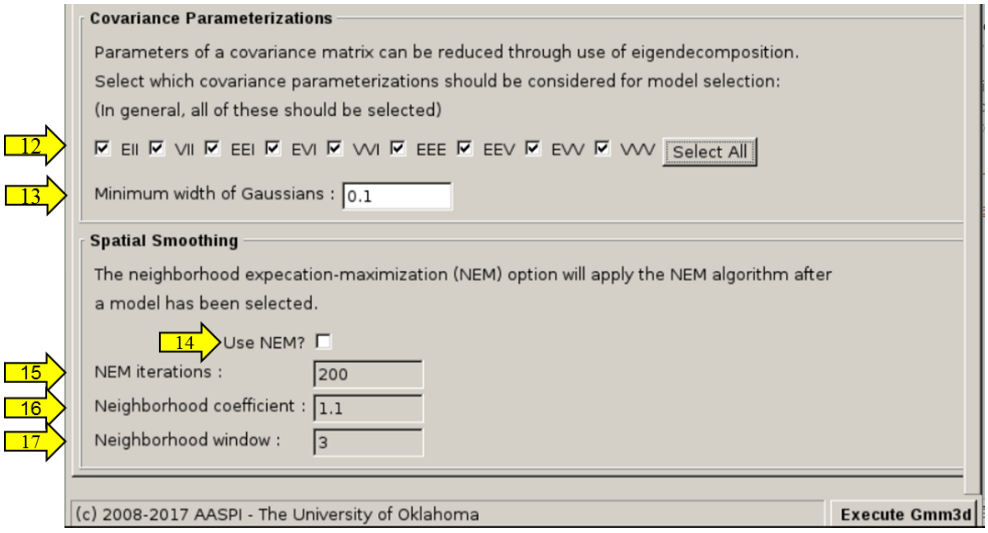
Parameter Description

To begin, at the top of **aaspi_util** GUI, click the 'Volumetric Classification' tab, located to the right of the 'File' tab. A down drop menu will appear containing all of the AASPI volumetric classification programs – **pca3d, kmeans3d, som3d, gtm3d, psvm3d, gmm3d,** and **define_training_data**. If using many attributes with **gmm3d,** good practice is to project the data into a latent space using one of the other AASPI programs. Two axes produced by **som3d** are used as an example of a latent space.

The following is the **gmm3d** GUI:



And scrolling down the GUI provides a few more options:



Click (1) *Browse,* and select *som_axis1_channel.H* as your first input seismic data set. Click (1) *Browse,* and select *som_axi2_channel.H* as your second input seismic data set. You also need to choose a (2) *unique project name*, which will be tacked on to an attribute descriptor to organize the output. We purposely used the project name previously used in the conversion from segy to AASPI format, but this is not necessary. Then (3) select *Suffix*. If you wish to run the program using the same data but under different parameters, then it will be useful to distinguish the between runs of **gmm3d** using different *Suffix* names such as *Suffix = 1* or *Suffix=VVV*.

Under the *Primary parameters* tab, the (4) *Number of attributes to use* textbox should be updated automatically when selecting a file using the *Browse* button. The (5) *Max clusters* textbox indicates the maximum number of clusters to search for. The program, **gmm3d,** searches for models with the number of components ranging from 1 to *Max clusters*. The (6) *Model selection* down drop menu offers two options: *use BIC* and *use max clusters*. Selecting *use BIC* will make the program calculate the *Bayesian Information Criterion* (BIC) after a model has been found; the optimal, or selected, model is then the model with the highest BIC value. Selecting *use max clusters* makes the program only consider models with a number of components equal to *Max clusters*. If the (7) *Include cluster decomposition volumes* box is checked, then the program outputs the posterior probability of each component as a different seismic volume; this option doesn't increase runtime, but will increase the amount of disk space. By default, the program outputs a single classification volume. For example, if we set *Max clusters* to be 10, select *use max clusters,* and check *Include cluster decomposition volumes*, then the output will contain a classification volume and 10 cluster decomposition volumes. The stochastic expectation-maximization is used in conjunction with the classification expectation-maximization

algorithm to optimize each model. The convergence criteria for the classification expectation maximization is when the partition doesn't change. The stochastic expectation-maximization runs for a specific number of iterations indicated by the (8) *stochastic expectation-maximization iterations* textbox. The (9, 10 and 11) *Training data extraction* textboxes allows the decimation rate to be specified.

Following Celeux and Govaert (1996), the size, shape, and orientation of each covariance matrix associated with each Gaussian component can be controlled through use of eigendecomposition. By controlling the size, shape and/or orientation of the covariance matrix, the number of parameters needed to estimate a Gaussian mixture model can be reduced and more parsimonious models can be generated. The *Bayesian Information Criterion* provides a way to compare models of differing parsimony by having a penalty term that increases in magnitude with the number of estimated parameters. The **gmm3d** GUI offers nine different covariance parameterizations in (12), with the more parsimonious models on the left and the more complex models on the right (VVV is the unconstrained covariance matrix). By default, all covariance parameterizations are checked, and models will be found for each parameterization that is checked. The (13) *Minimum width of Gaussian* is directly related to the covariance matrix and must be non-zero to prevent covariance matrices from becoming singular.

The classification of voxels using conventional Gaussian mixture models does not use spatial information, and the resulting classification may seem disjointed. The neighborhood expectation-maximization (NEM) algorithm allows for spatial information to be included in the optimization process. This is done by adding a spatial term to the conventional expectation-maximization objective function. By checking the (14) *use NEM* checkbox, the program will run the NEM algorithm after a model has been selected to try and smooth the classification in the spatial domain. The NEM algorithm can get computationally expensive, so the (15) *maximum number of iterations* can be set in case convergence doesn't occur quickly enough. The (16) *neighborhood coefficient* is a scalar value that gives weight to the spatial information; setting the *neighborhood coefficient* to zero will result in the conventional expectation-maximization algorithm. The (17) *neighborhood window* defines the furthest training voxel that will influence the classification. This means that the actual window size is the user defined *neighborhood window* value multiplied by the decimation rates that construct the training data set.

Gaussian mixture model theory

A multivariate Gaussian distribution can be defined as follows:

$$\varphi(\mathbf{x}|\mu, \mathbf{C}) = \frac{1}{(2\pi)^{\frac{d}{2}}|\mathbf{C}|^{\frac{1}{2}}} e^{-\frac{1}{2}(\mathbf{x}-\mu)'\mathbf{C}^{-1}(\mathbf{x}-\mu)}$$

where d is the number of input attributes, $\mathbf{C}$ is a covariance matrix, and $\boldsymbol{\mu}$ is the mean. In general, a Gaussian mixture model approximates a probability density function by adding together weighted Gaussian distributions (Figure 2.1). A Gaussian mixture model can also be used for clustering and classification, where a cluster of a Gaussian mixture model can be described by its weighted Gaussian distribution. A Gaussian mixture density

with K clusters for a data vector $\mathbf{x}_i$, where $i = 1 \ldots N$, is given by

$$p(\mathbf{x}_i|\psi) = \sum_{j=1}^{K} \pi_j \varphi(\mathbf{x}_i|\mu_j, \mathbf{C}_j)$$

where π_j is the weight of the j^{th} cluster such that

$$\sum_{j=1}^{K} \pi_j = 1$$

and $\pi_j \geq 0$. The posterior probability, $w_{i,j}$, of a data vector, $\mathbf{x}_i$, belonging to a certain cluster, j, is given by

$$w_{i,j} = \frac{\pi_j \varphi(\mathbf{x}_i|\mu_j, \mathbf{C}_j)}{\sum_{j=1}^{K} \pi_j \varphi(\mathbf{x}_i|\mu_j, \mathbf{C}_j)} = \frac{\pi_j \varphi(\mathbf{x}_i|\mu_j, \mathbf{C}_j)}{p(\mathbf{x}_i|\psi)}.$$

A data vector is then classified to the cluster with the greatest posterior probability, $\max_j(w_{i,j})$. The uncertainty in the classification for a data vector, $\mathbf{x}_i$, is then (Bensmail et al., 1996)

$$U_i = 1 - \max_j(w_{i,j}).$$

The general objective is to generate models with a range of differing clusters and covariance parameterizations, and then select the best model using the Bayesian Information Criterion (BIC). The selected model can then be used to classify data vectors as being from a certain cluster to generate unsupervised seismic facies. Under this classification scheme, data vectors are assumed to be from a single Gaussian distribution.

Initialization and Optimization

The means of each Gaussian component are initialized using the centroids from the **kmeans** output. The weights are initialized by taking the number of attribute vectors assigned to each centroid from the **kmeans** output and dividing by the total number of attribute vectors. The covariance matrices are initialized by calculating the within-cluster covariance matrices (i.e. the covariance with respect to each centroid using their respective data attribute vectors from the **kmeans** output).

Learning of the mixture parameters, $\{\pi_j, \boldsymbol{\mu}_j, \mathbf{C}_j,\}$, is often carried out with the Expectation-Maximization (EM) algorithm, but the inherent nature of seismic data requires that alternative EM algorithms to be used. The objective is to use a Gaussian mixture model to classify voxels as being from a certain Gaussian distribution. The Classification Expectation-Maximization (CEM) is implemented with the Stochastic Expectation-Maximization (SEM) algorithms to deal with this (Celeux and Govaert, 1996). The SEM algorithm is iterated for a user-defined number of times, and then the CEM algorithm is ran afterwards until there is no change in the classification of voxels.

The CEM and SEM algorithms are like the conventional EM algorithm but have an additional intermediate step called the C-step and S-step respectively. The C- and S-step of both algorithms define a partition of the input data, and the mixture model parameters are updated using the respective partitions. The CEM algorithm is briefly outlined:

- *E-step*: Accumulate the N x K responsibility matrix, $\mathbf{W}$ (also called the posterior probability). Each element in the matrix is given by

$$w_{i,j} = \frac{\pi_j \varphi(\mathbf{x}_i|\mu_j, \mathbf{C}_j)}{p(\mathbf{x}_i|\psi)}$$

- *C-step*: Create K-partitions by assigning each $\mathbf{x}_i$ to the cluster that provides the highest posterior probability according to the responsibility matrix.

- *M-step*: For each cluster, update the mixture parameters, $\{\pi_j, \boldsymbol{\mu}_j, \mathbf{C}_j,\}$, with the respective partition defined in the previous step using maximum likelihood estimates.

The SEM algorithm is identical, except the *C*-step is replaced by the *S*-step:

- *S-step*: Create *K*-partitions by randomly assigning each $\mathbf{x}_i$ to a cluster using the posterior probabilities in the responsibility matrix.

The SEM algorithm helps to avoid sub-optimal solutions provided by the CEM. However, since the S-step is random, the partition it creates is also random. To deal with this, the resulting SEM model is used to initialize the CEM algorithm which then gives a final partition.

Covariance parameterizations

The size, shape, and orientation of each covariance matrix associated with each Gaussian component can be controlled through use of eigendecomposition. By controlling the size, shape and/or orientation of the covariance matrix, the number of parameters needed to estimate a Gaussian mixture model can be reduced and more parsimonious models can be generated. The nine closed form solutions from Celeux and Govaert (1993) are considered:

Module symbol	Covariance parameterization	Description
EII	$\lambda\mathbf{I}$	Spherical. Equal volume.
VII	$\lambda_j\mathbf{I}$	Spherical. Varying volume.
EEI	$\lambda\mathbf{B}$	Diagonal. Equal volume. Equal shape.
EVI	$\lambda\mathbf{B}_j$	Diagonal. Equal volume. Varying shape.
VVI	$\lambda_j\mathbf{B}_j$	Diagonal. Varying volume. Varying shape.
EEE	$\lambda\mathbf{D}\mathbf{A}\mathbf{D}^{\mathrm{T}}$	Ellipsoidal. Equal volume. Equal shape. Equal orientation.
EEV	$\lambda\mathbf{D}_j\mathbf{A}\mathbf{D}_j^{\mathrm{T}}$	Ellipsoidal. Equal volume. Equal shape. Varying orientation.
EVV	$\lambda\mathbf{D}_j\mathbf{A}_j\mathbf{D}_j^{\mathrm{T}}$	Ellipsoidal. Equal volume. Varying shape. Varying orientation.
VVV	$\lambda_j\mathbf{D}_j\mathbf{A}_j\mathbf{D}_j^{\mathrm{T}}$	Ellipsoidal. Varying volume. Varying shape. Varying orientation.

Model selection

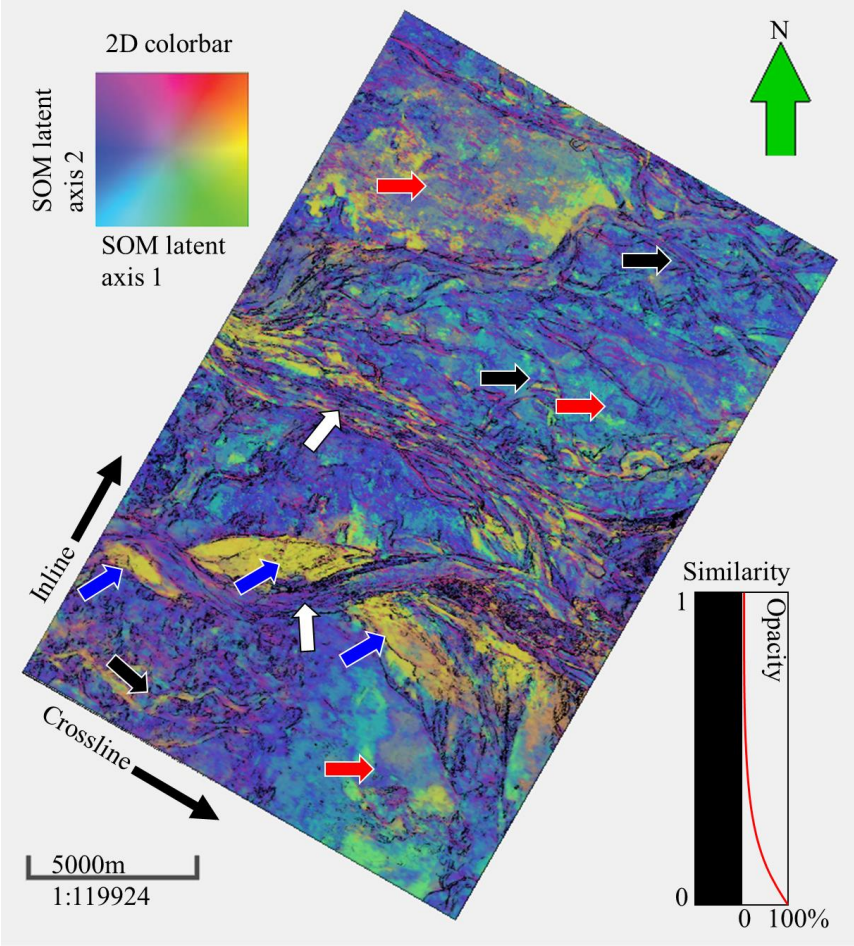
Likelihood is an effective way to compare models of the same complexity, but is not sufficient to compare models of differing complexity (i.e. differing number of clusters and/or covariance parameterization). The Bayesian Information Criterion (BIC) proposed by Schwarz (1978) can be implemented to automatically choose the best model. The BIC is defined as follows:

$$BIC = \log\big(L(\psi)\big) - \frac{1}{2}m\,\log(n)$$

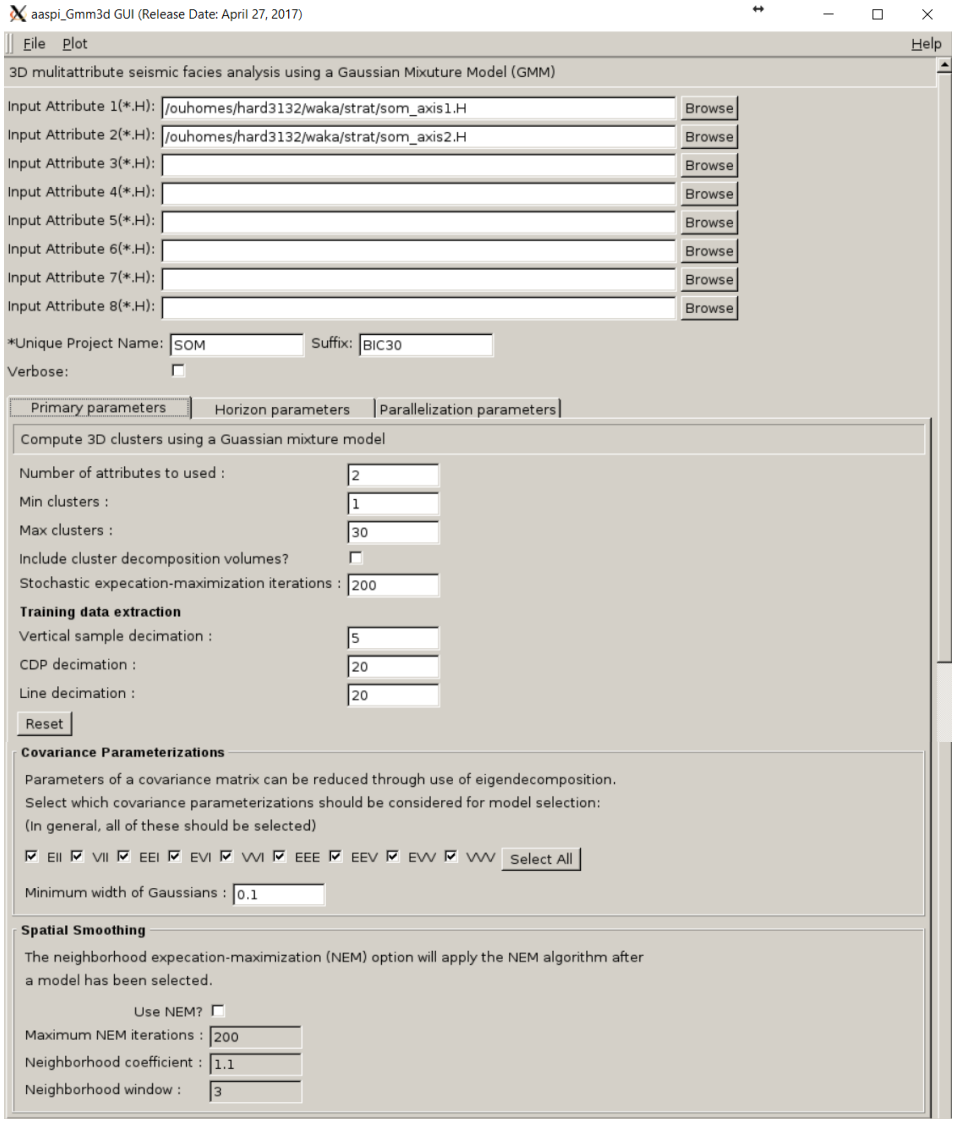
where $L(\psi)$ is the loglikelihood of the model, m is the number of estimated parameters, and n is the number of training voxels. The BIC penalizes extraneous parameters and large sample sizes to choose the best model in terms of parsimony and likelihood.

Visualization of the result

To view the resulted facies map, a user can either use aaspi_plot, or import the facies volume into another commercial seismic interpretation software. If using a commercial interpretation software, remember to use a discrete colorbar, or turn off the value interpolation, because the values in a ***gmm3d*** generated facies map are discrete values. In this example, facies are generated along a horizon in a turbidite system in Canterbury basin, New Zealand. The inputs to ***gmm3d*** are two ***som3d*** latent axes generated from peak spectral frequency, peak spectral magnitude, curvedness, and coherent energy (Zhao et al. 2016). The two latent axes can be correndered to produce a single facies map along the horizon, and similarity is overlaid to give a sense of structure:

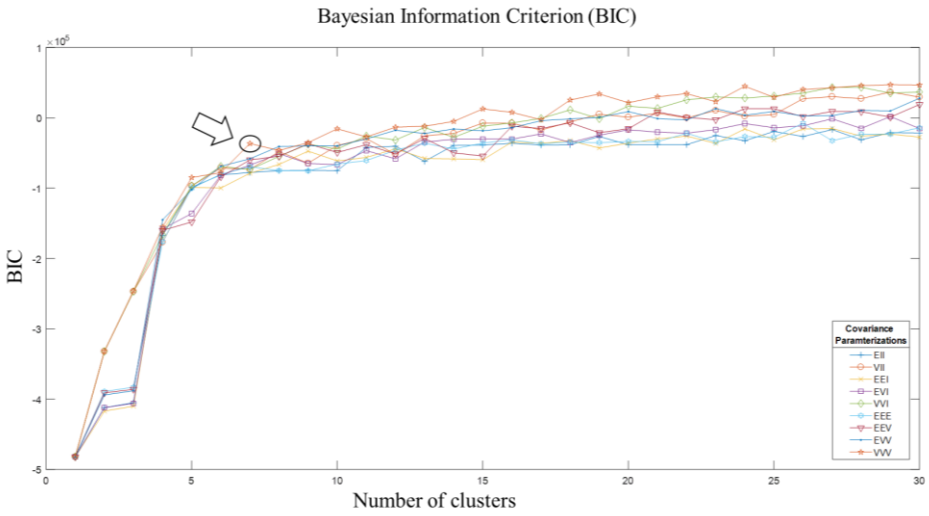


Initially, the minimum and maximum number of clusters are set to be 1 and 30 respectively because we want to search through a range of clusters to find the best one. The number of stochastic expectation-maximization iterations needs to be set due to lack of convergence properties; the default of 200 should be fine for most applications. All covariance parameterizations are selected and given a minimum width of 0.1, and no spatial smoothing options are considered.



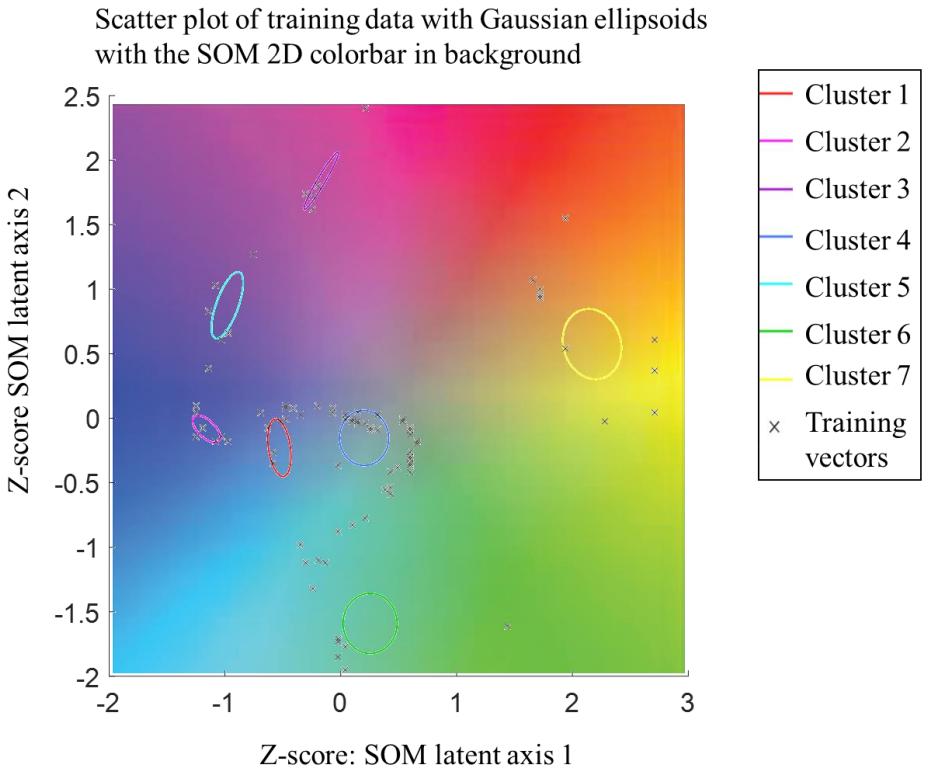
Volumetric Classification: Program **gmm3d**

The program ***gmm3d*** automatically selects the model with the highest BIC value, which in this case would be 30 clusters with a covariance parametrization of *VVV*. Executing ***gmm3d*** under these parameters produces two comma-delimited files called BIC.csv and training_vectors.csv, a classification attribute volume, and an uncertainty attribute volume. The BIC values for each covariance parametrization are plotted against the number of clusters to see how the BIC function looks.

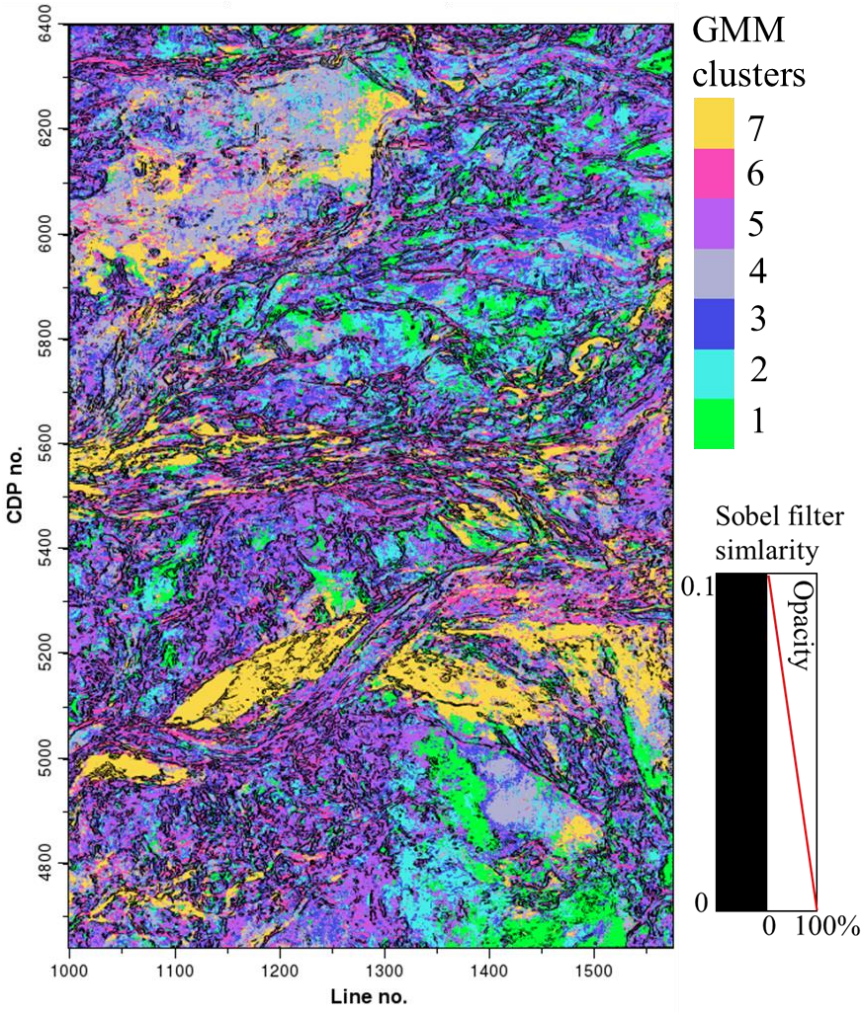


Generally, the BIC has a maximum and the model with the maximum BIC value is considered the best model (in this case that would be 30 clusters with covariance parametrization *VVV*). In this case however, the BIC seems to increase as the number of components get larger. This may be due to cluster geometries in the latent space not fitting a Gaussian distribution well, creating the need for multiple Gaussian distributions to fit the cluster. In consideration of this, the mixture model with 7 clusters and a covariance parameterization of *VVV* is selected due to the location being a local maximum and due to the BIC not increasing significantly. The program ***gmm3d*** is re-run under the same parameters, except the number of clusters is set to be 7 and the only covariance parameterization selected is *VVV*. This means that only one model will be considered and applied to the data, and BIC.csv will also be overwritten.

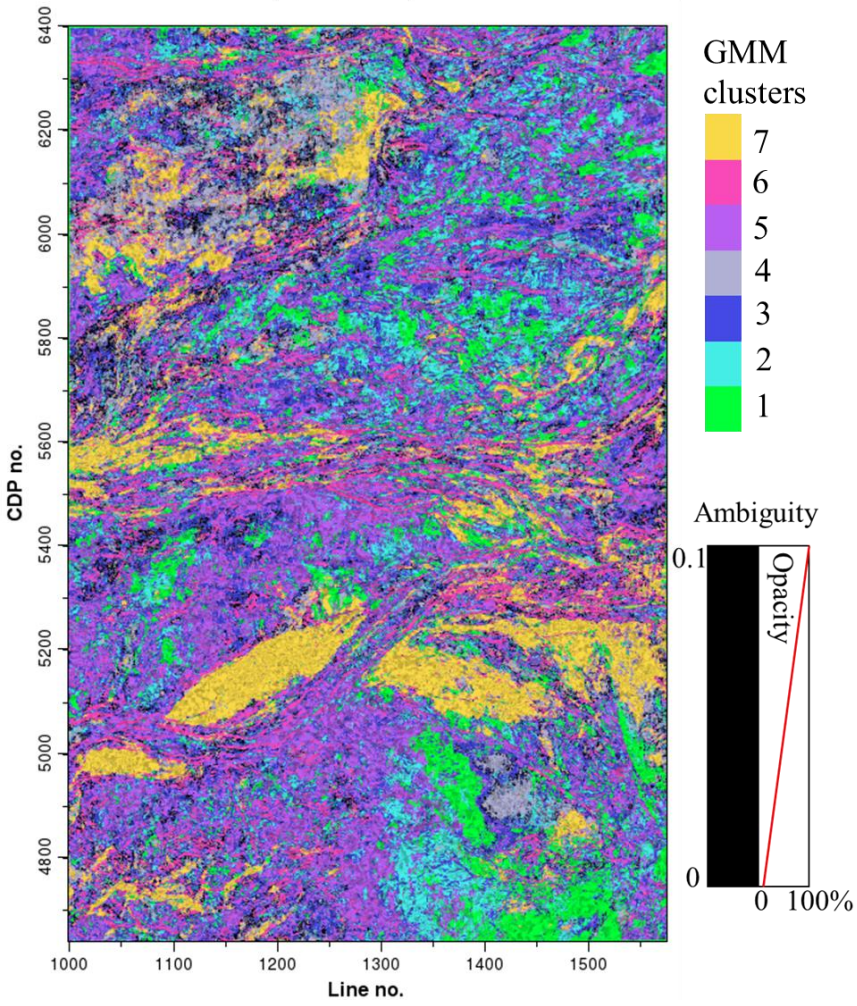
After the model is selected and applied to the data, the user must select a good colorbar. For the sake of comparison, each cluster is assigned a color close to where the clusters' means would be on the 2D colorbar from the SOM facies map. Here we show the training data with Gaussian ellipsoids corresponding to each cluster of the Gaussian mixture model with the 2D colorbar in the background:



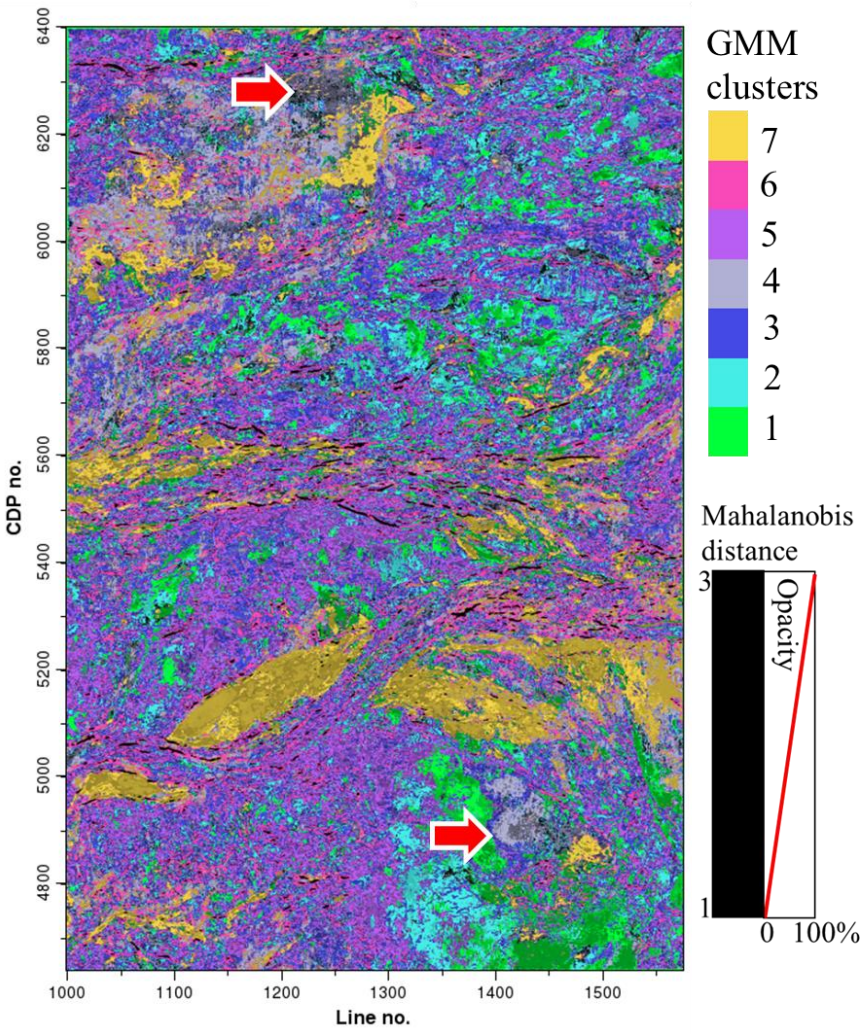
Each cluster is assigned a color close to where the centers of the ellipsoids are in the cross plot above. The figure below is a facies map from ***gmm3d*** along a horizon in a turbidite system in Canterbury basin, New Zealand with similarity overlaid to give a sense of structure. We interpret the white arrows as multistoried channels, black arrows as sinuous channel complexes, blue arrows as a sand filled channel, and red arrows as slope fans.



The classification is overlaid by the ambiguity to hide areas where the mixture model is not so confident in the classification.



Moreover, anomalous voxels that do not fit the GMM well can be highlighted by co-rendering the classification with the Mahalanobis distance between the cluster center and each voxel. Red arrows highlighting anomalous areas within possible slope fan deposits.



References

Ambroise, C., M. Dang, and G. Govaert, 1997, Clustering of spatial data by the EM algorithm: geoENV I – Geostatistics for Environmental Applications, **9**, 493-504

Celeux, G. and G. Govaert, 1992, A Classification EM algorithm for clustering and two stochastic versions: Computation Statistics & Data Analysis, **14**, 315-332.

Celeux, G. and G. Govaert, 1993, Gaussian Parsimonious clustering models. Pattern Recognition, 28, 781-793.

Zhao, T., J. Zhang, F. Li, K. J. Marfurt, 2016, Characterizing a turbidite system in Canterbury Basin, New Zealand using seismic attributes and distance-preserving self-organizing maps: Interpretation, **4**, SB79-SB89.