# Detecting karst caverns by pattern recognition

*Fantine Huot and Robert Clapp*

## ABSTRACT

Strong localized heterogeneities in the subsurface, such as karst caverns and sinkholes, cause scattering of seismic waves, thereby degrading the images obtained in conventional processing. We explore the possibility of using pattern recognition techniques for detecting these strong heterogeneities from seismic data. Through synthetic models, we generate training data with significant scattering and demonstrate how to decompose it into elementary shapes fit for machine learning algorithms by applying continuous wavelet transforms. We then provide an overview of support vector machine classification and present how we intend to apply it to our problem.

## INTRODUCTION

The Tengiz carbonate platform in northwestern Kazakhstan is one of the largest producing oil fields in the world. Recently, exploration has targeted karst-like zones with cavernous porosity along the margin of the platform. Lester et al. (2015) showed that these karst caverns appear as localized high-amplitude events on seismic volumes but can also resemble residual noise that may have persisted through processing and imaging. Such localized features induce positional uncertainty in the migrated velocity model and can represent drilling hazards.

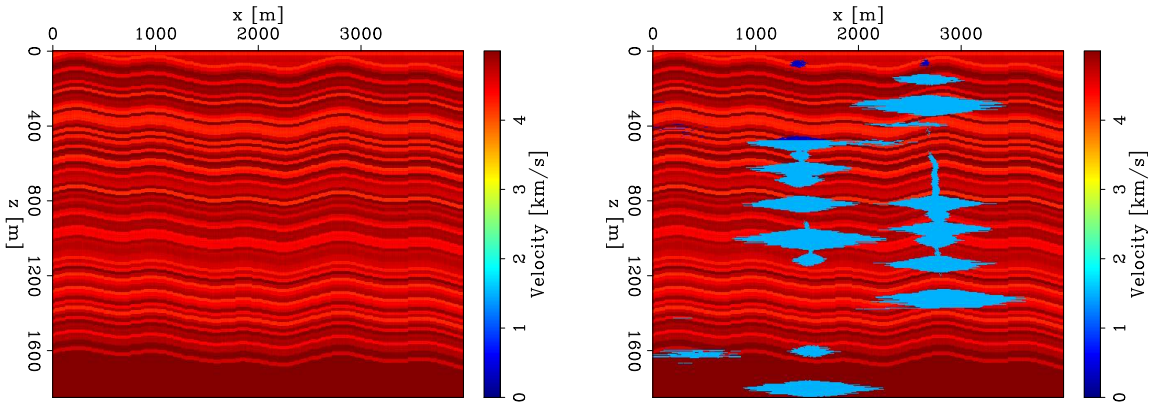
While various methodologies, such as diffraction migration or beam migration (Fomel et al., 2007; Berkovitch et al., 2009; Lester et al., 2015), have been proposed to address the issue of imaging these strong heterogeneities, herein we investigate the potential of techniques commonly used in pattern recognition.

The first algorithm for pattern recognition was introduced 80 years ago (Fisher, 1936). With the advent of computers and the information age, statistical learning has become a very hot field in many scientific areas as well as marketing, finance, and other business disciplines. In recent years, new and improved software packages have significantly eased the implementation burden for many statistical learning methods, providing scientists and practitioners with complete toolkits for training, testing, and deploying models with well-documented examples for all these tasks (Collobert et al., 2002; Pedregosa et al., 2011; James et al., 2013; Jia et al., 2014). With algorithms automatically tracking faces in photographs (Osuna et al., 1997), what would prevent us from training machines to detect specific seismic responses in our data?

We initiated our study with a simple synthetic cavern model from which we generated our training data. We then decomposed it into elementary shapes by applying continuous wavelet transforms, which will help us set up our pattern recognition problem. Among the wide scope of existing machine learning algorithms, we then decided to focus on support vector machine classification, an overview of which is provided in the last section.

## SYNTHETIC DATA GENERATION

The first step of our study is to generate synthetic seismic data which we can use as training data.

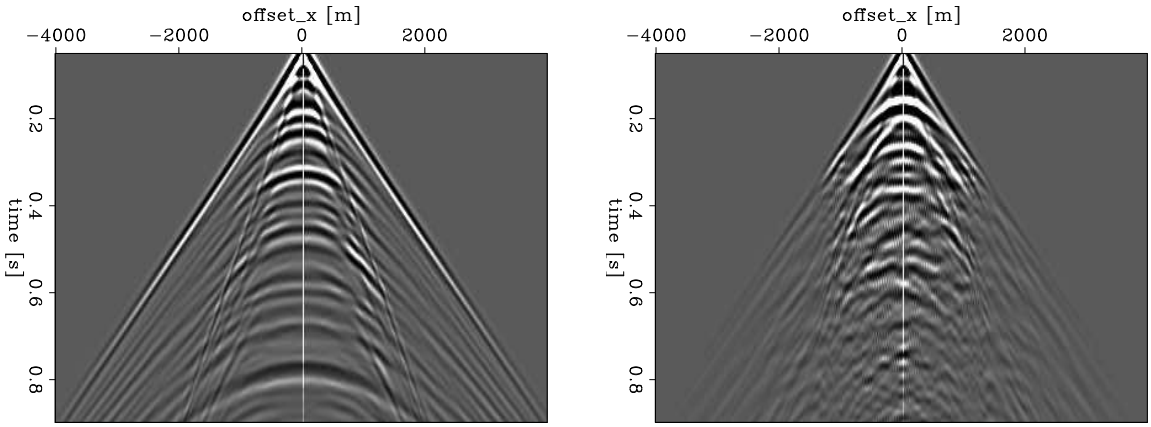



(a) (b)

Figure 1: Synthetic models created for this study. We display 2D slices of the P-wave velocity component. (a) Reference model of layered limestone. (b) Karst cavern model. [**ER**]

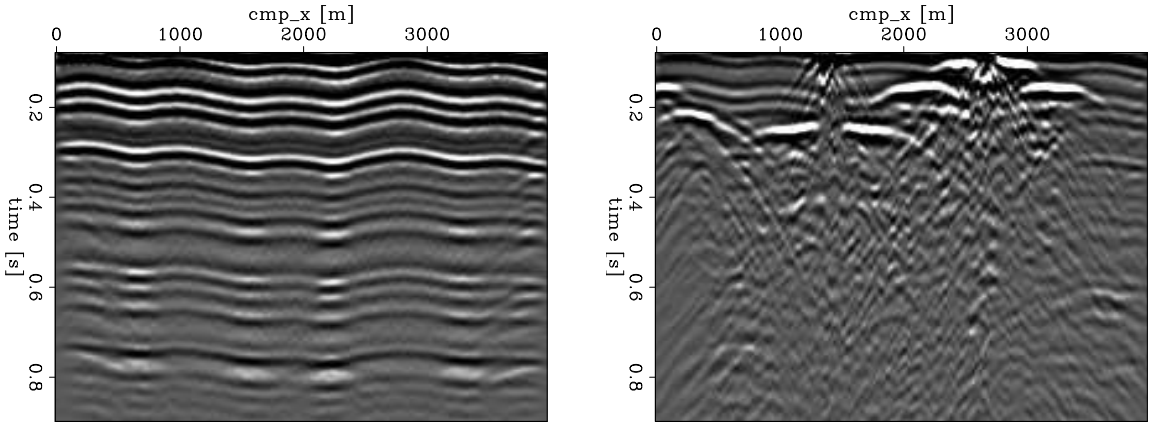
For this purpose, we create a three-dimensional synthetic model of an underground karst channel system in a limestone bedding. We first build a model of layered limestone, as illustrated in Figure 1(a), which we designate as the reference model. Then, caverns of different scales are added, as shown in Figure 1(b), with connected channels partially filled with water. The model is elastic isotropic: the P-wave velocities range from 4000 m/s to 5000 m/s for the limestone, and are equal to 1490 m/s and 330 m/s for the water and air inside the caverns; the S-wave velocities range from 2100 m/s to 2700 m/s for the limestone, and are set near to zero inside the caverns; and the densities range from 2200 kg/m$^3$ to 2700 kg/m$^3$ for the limestone, and are arbitrarily set at water density inside all the caves to avoid significant density contrasts that would generate instability in the wave propagation code. In the following, we consider 2D slices of this model.

The synthetic seismic data is generated using the elastic wave propagation code developed by Alves (2015). This simulation uses a Ricker-type explosive source in the normal stress field, with a peak frequency of 20 Hz. The recordings for a single




(a) (b)

Figure 2: Data recorded for a single shot. (a) Data generated from the reference model. (b) Data generated from the karst cavern model. When comparing the two panels, we clearly see the scattering effect due to the presence of the caverns. [**CR**]




(a) (b)

Figure 3: Zero-offset common midpoint gathers. (a) Data generated from the reference model. (b) Data generated from the karst cavern model. While Panel (a) reflects the aspect of the initial model of layered limestone, the scattering in Panel (b) makes it difficult to tell the location of the caverns. [**CR**]

shot clearly illustrate the scattering effect associated with the presence of the caverns (Figure 2). The zero-offset CMP gathers are produced with 200 shots, with receivers located at the surface. While the CMP gather obtained from the reference model in Figure 3(a) clearly reflects the layers of limestone, we see that the caverns completely scatter the wavefield in Figure 3(b), thereby making it difficult to tell the location of the caverns from only the CMP images.

## CONTINUOUS WAVELET TRANSFORM

Continuous wavelet transforms (CWT) are commonly used in pattern recognition, as they have the ability to decompose complex patterns into elementary forms. They measure the similarity between a signal and an analyzing wavelet by comparing the input signal to shifted and compressed or stretched versions of the wavelet. For an input signal $f$, and a mother wavelet $\psi$, the CWT can be expressed as follows:

$$C(a,\, b\,;f(t),\, \psi(t)) = \int_{-\infty}^{\infty} f(t)\, \frac{1}{\sqrt{a}}\, \psi^* \left( \frac{t-b}{a} \right) dt\ ,$$

where $a$ is the scaling factor, $b$ the time shift, and $*$ denotes the complex conjugate.

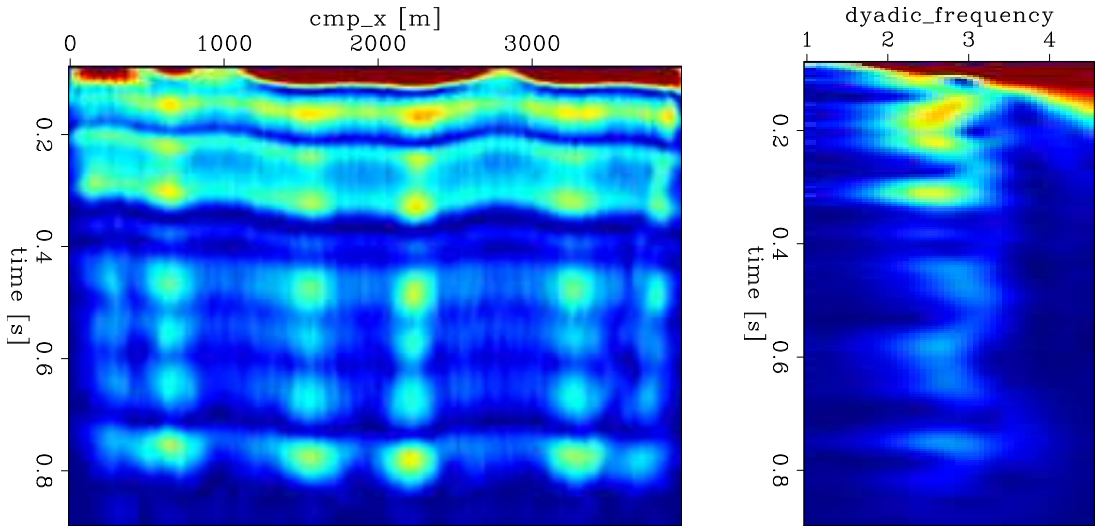
By applying CWT on the zero-offset CMP gathers with a Morlet wavelet as analyzing function, we obtain the results presented in Figures 4 and 5. For each midpoint, we now have a two-dimensional time-dyadic frequency representation of the signal. When we compare the results obtained from the cavern system model in Figure 4(b) with the ones obtained from the reference model in Figure 5(b), we clearly distinguish localized high-amplitude parcels in the time-dyadic frequency space that qualitatively match the location of the caverns in terms of travel-time depths.

As a consequence, we set up our pattern recognition problem in the time-dyadic frequency space. As a basic approach, we configure a sliding window of constant size to vertically scan the time-dyadic frequency panels for all midpoints, and for each windowed image, call a classifier function to determine whether it contains patterns associated with caverns or not. This approach would generate an output of cavern locations as a function of travel-time depth and midpoint.

## SUPPORT VECTOR MACHINES

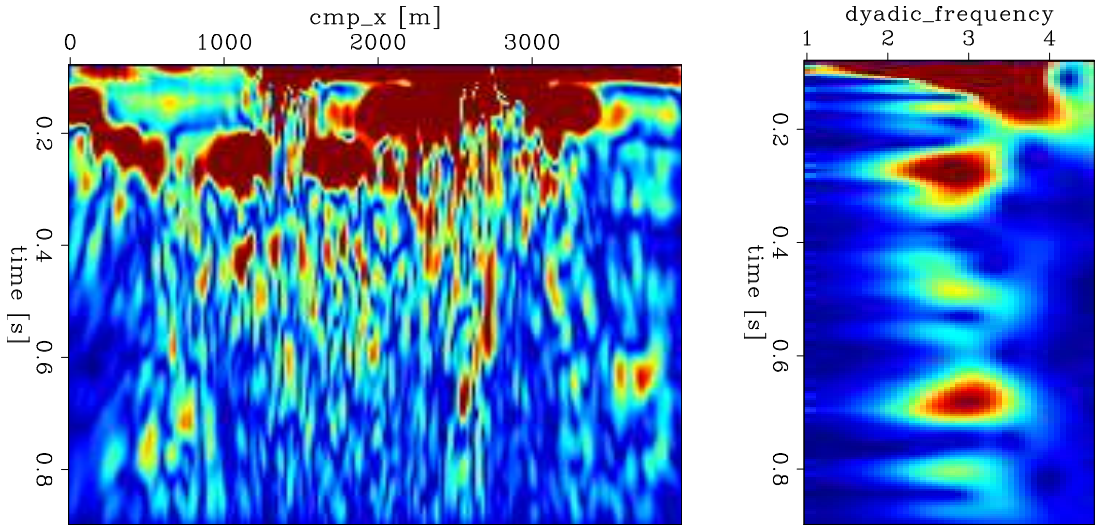
We now have to address the question of how to build our classifier. Image comparisons, especially when the images contain basic shapes, are problems with highly correlated features. Therefore, we focus on support vector machines (SVM), known to be efficient for these type of problems (Hsu et al., 2003; Hastie et al., 2005).

To train the classifier, we build a training dataset of $N$ constant size downsampled grayscale images picked from the time-dyadic frequency space, where half of them correspond to known locations of caverns, while the other half are generated from the reference model. We can express this training data as $N$ pairs




(a) (b)

Figure 4: Continuous wavelet transform applied to the data derived from the reference model. (a) Projection in time-midpoint space at a dyadic frequency equal to 3. (b) Projection in time-dyadic frequency space at a midpoint equal to 2550m. [**CR**]




(a) (b)

Figure 5: Continuous wavelet transform applied to the data derived from the karst cavern model. (a) Projection in time-midpoint space at a dyadic frequency equal to 3. (b) Projection in time-dyadic frequency space at a midpoint equal to 2550m. We clearly distinguish localized high amplitude parcels in Panel (b), which correspond to the specific seismic response associated with the presence of the caverns. [**CR**]

$(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)$, where the $x_i \in \mathbb{R}_p$ represent the features, which are the values associated to the $p$ pixels of each of the training images, and the $y_i \in \{-1, 1\}$ are the binary labels indicating the class to which each feature belongs, namely, whether the image corresponds to a cavern or not.

In the following section, we provide an overview of the theory behind SVM based on formulations by Hastie et al. (2005).

## The linear support vector classifier

We define a hyperplane by the following:

$$\{x : f(x) = x^T\beta + \beta_0 = 0\}, \tag{1}$$

where $\beta$ is a unit vector and $\beta_0$ a constant. The classification rule induced by $f(x)$ is:

$$G(x) = \text{sign}[x^T\beta + \beta_0]. \tag{2}$$

Let us suppose that our classes are completely separable, that is to say, there exists a function $f(x) = x^T\beta + \beta_0$ such that $\forall i,\ y_i f(x_i) > 0$. Therefore, we can find the hyperplane that creates the greatest margin between the training points of each class. The following optimization problem:

$$\min_{\beta,\beta_0} \|\beta\| \quad \text{subject to} \quad y_i({x_i}^T\beta + \beta_0) \geq 1,\ \forall i, \tag{3}$$

captures this concept. This formulation is considered the support vector criterion for separated data.

In practice however, the classes overlap in feature space. Therefore, we re-express the problem to allow some points to be on the wrong side of the margin. We introduce the slack variables $\xi = (\xi_1, \xi_2, \cdots, \xi_N)$, and rewrite Equation (3) as:

$$\min_{\beta,\beta_0} \|\beta\| \quad \text{subject to} \begin{cases} y_i({x_i}^T\beta + \beta_0) \geq 1 - \xi_i\,,\ \forall i \\ \xi_i \geq 0,\ \forall i \quad \text{and} \quad \sum \xi_i \leq K. \end{cases} \tag{4}$$

This way is commonly used to define the support vector classifier for the nonseparable case. By nature of the criterion, we see that points well inside their class boundary do not play a big role in shaping the boundary. The value $\xi_i$ is the proportional amount by which the prediction $f(x_i) = {x_i}^T\beta + \beta_0$ is on the wrong side of its margin. Therefore, by bounding the sum $\sum \xi_i$, we bound the total proportional amount by which predictions fall on the wrong side of their margin. Misclassifications occur when $\xi_i > 1$, so bounding $\sum \xi_i$ at a value $K$ bounds the total number of training misclassifications at $K$.

## Computing the support vector classifier

We now have to address how to implement the support vector classifier in practice. Equation (3) is a quadratic criterion with linear inequality constraints. Therefore, it is a convex optimization problem. Herein, we describe a quadratic programming solution to this problem using Lagrange multipliers. Computationally, it is convenient to re-express Equation (3) as follows:

$$\begin{aligned} &\min_{\beta,\beta_0} \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{N}\xi_i \\ &\text{subject to} \quad \xi_i \geq 0,\ y_i({x_i}^T\beta + \beta_0) \geq 1 - \xi_i\,,\ \forall i, \end{aligned} \tag{5}$$

where the cost parameter $C$ replaces the constant in Equation (3). The separable case corresponds to $C = \infty$.

The Lagrangian primal objective function is expressed as follows:

$$L_P = \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\alpha_i[y_i(x_i^T\beta + \beta_0) - (1-\xi_i)] - \sum_{i=1}^{N}\mu_i\xi_i, \tag{6}$$

which we minimize with respect to $\beta$, $\beta_0$, and $\xi_i$. Setting the respective derivatives to zero, we get the following:

$$\beta = \sum_{i=1}^{N}\alpha_i y_i x_i, \tag{7}$$

$$0 = \sum_{i=1}^{N}\alpha_i y_i, \tag{8}$$

$$\alpha_i = C - \mu_i,\ \forall i, \tag{9}$$

as well as the positivity constraints $\alpha_i,\ \mu_i,\ \xi_i \geq 0,\ \forall i$. By re-injecting these constraints into Equation (6), we obtain the following Lagrangian dual objective function:

$$L_D = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j x_i^T x_j. \tag{10}$$

Because we are using Lagrange multipliers with inequality constraints, we also have to add the following Karush–Kuhn-Tucker conditions to our problem:

$$\alpha_i[y_i(x_i^T\beta + \beta_0) - (1-\xi_i)] = 0, \tag{11}$$

$$\mu_i\xi_i = 0, \tag{12}$$

$$y_i(x_i^T\beta + \beta_0) - (1-\xi_i) \geq 0,\ \forall i. \tag{13}$$

Together, Equation (7) through Equation (13) uniquely characterize the solution to the primal and dual problem.

Maximizing the dual objective function as shown in Equation (10) is a simpler convex quadratic programming problem than the primal shown in Equation (6), and can be solved with standard techniques (Gill et al., 1981).

From (7), we see that the solution for $\beta$ has the following form:

$$\hat{\beta} = \sum_{i=1}^{N} \hat{\alpha}_i y_i x_i, \tag{14}$$

with nonzero coefficients $\hat{\alpha}_i$ only for those observations $i$ for which the constraints in Equation (13) are exactly met. These observations are called the support vectors, because $\hat{\beta}$ is represented in terms of these vectors alone. The solution for $\beta_0$ can be derived from Equation (13) using any support point lying on the edge of the margin ($\xi_i = 0$). Typically, for numerical stability, the average of all the solutions is used.

Given the solutions $\hat{\beta}_0$ and $\hat{\beta}$, the decision function can be written as follows:

$$\hat{G}(x) = \text{sign}[x^T \hat{\beta} + \hat{\beta}_0]. \tag{15}$$

The tuning parameter of this procedure is the cost parameter $C$.

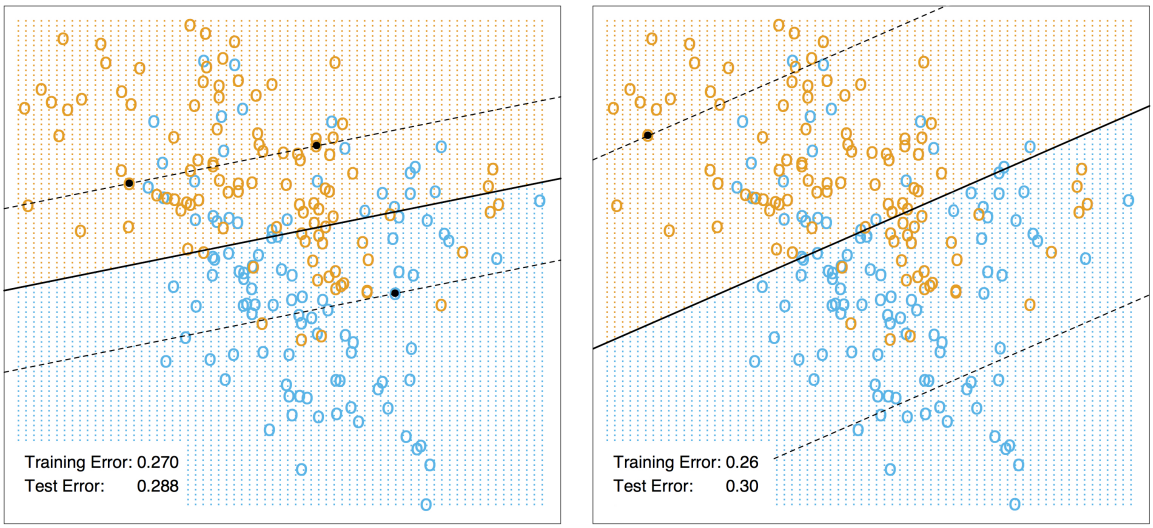



Figure 6: The linear support vector boundary for a dataset with two overlapping classes, for two different values of $C$. A lower value of $C$ allows for more slack and larger margins. The broken lines indicate the margins where $f(x) = \pm 1$. The support points ($\alpha_i > 0$) are all the points on the wrong side of their margin. The black solid dots are those support points falling exactly on the margin ($\xi_i = 0$) (Hastie et al., 2005). [**NR**]

Figure 6 provides a visual example of linear support vector boundaries for a dataset with overlapping classes for two different values of the cost parameter $C$.

## Nonlinear support vector machines and kernels

The support vector classifier subsequently described finds linear boundaries in the input feature space. To achieve better training-class separation, we introduce the idea of enlarging the feature space using basis expansions. Linear boundaries in the enlarged space translate to nonlinear boundaries in the original space, thereby making the problem more flexible. Once the basis functions $h_m(x)$, $m = 1, \cdots, M$ are selected, the procedure is the same as before. We fit the support vector classifier using input features $h(x_i) = (h_1(x_i), h_2(x_i), \cdots, h_M(x_i))$, $i = 1, \cdots, N$, and produce the (nonlinear) function $\hat{f}(x) = h(x)^T\hat{\beta} + \hat{\beta}_0$. The classifier is $\hat{G}(x) = \text{sign}[\hat{f}(x)]$ as before.

The support vector machine classifier is an extension of this idea, where the dimension of the enlarged space is allowed to get very large, infinite in some cases. It might seem that this enlargement would make the computations prohibitive. However, in the following we show we can represent the optimization and its solution in a special way that only involves the input features by way of inner products. We then see that for particular choices of $h$, these inner products can be computed very cheaply.

The Lagrange dual function is re-expressed as follows:

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i\alpha_j y_i y_j \langle h(x_i), h(x_j)\rangle. \tag{16}$$

The solution function $f(x)$ can be rewritten as follows:

$$\begin{aligned} f(x) &= h(x)^T\beta + \beta_0 \\ &= \sum_{i=1}^{N} \alpha_i y_i \langle h(x), h(x_i)\rangle + \beta_0. \end{aligned} \tag{17}$$

Both these equations involve $h(x)$ only through inner products. In fact, we need not specify the transformation $h(x)$ at all, but require only knowledge of the following kernel function:

$$K(x, x') = \langle h(x), h(x')\rangle, \tag{18}$$

that computes the inner products in the transformed space. $K$ should be a symmetric positive definite function.

Three popular choices for $K$ in the SVM literature are shown in the following:

$$d\text{th—Degree polynomial:} \quad K(x, x') = (1 + \langle x, x'\rangle)^d, \tag{19}$$

$$\text{Radial basis:} \quad K(x, x') = \exp(-\gamma\|x - x'\|^2), \tag{20}$$

$$\text{Neural network:} \quad K(x, x') = \tanh(\kappa_1\langle x, x'\rangle + \kappa_2). \tag{21}$$

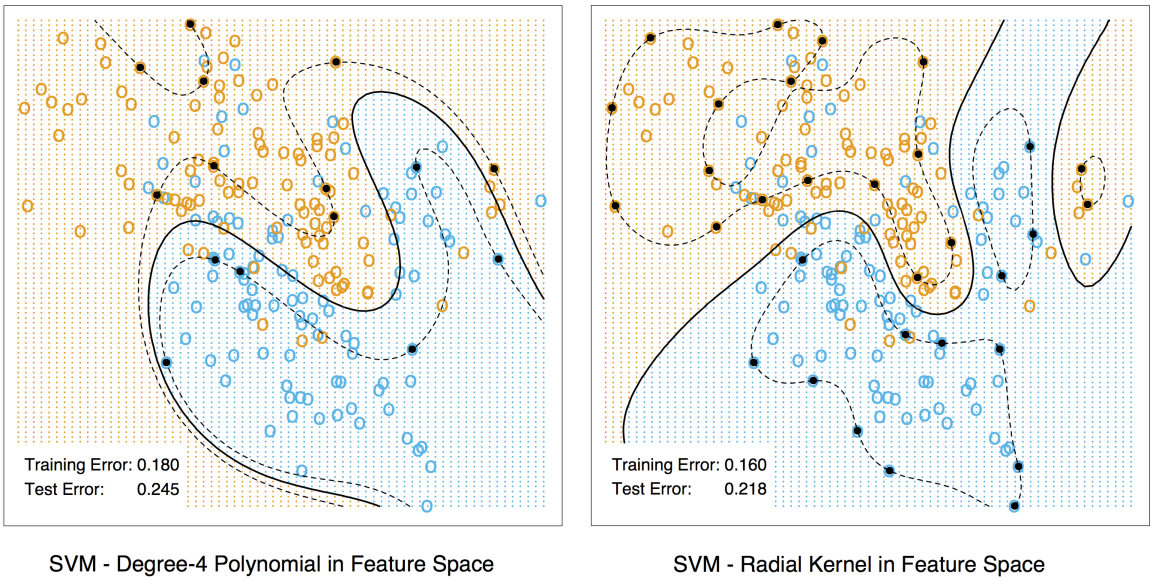



Figure 7: Two nonlinear SVMs for a dataset with two overlapping classes using two different kernels. The left panel uses a 4th degree polynomial kernel, the right one a radial basis kernel (with $\gamma = 1$) (Hastie et al., 2005). [**NR**]

Figure 7 provides a visual example of the impact of using different kernels.

Finally, the solution can be expressed as follows:

$$\hat{f}(x) = \sum_{i=1}^{N} \hat{\alpha}_i y_i K(x, x') + \hat{\beta}_0. \tag{22}$$

By using particular kernels, SVM can thus be implemented in a very cost-effective manner. Once the algorithm is trained, the classification only involves cheap inner products.

## DISCUSSION

As the saying goes, a journey of a thousand miles begins with a single step. We have now set up a basic pattern recognition problem and described the choices we made along the way. We are currently implementing the described methodology, and nothing will guide us better to the next step than our first test results.

However, we can already imagine many more paths to explore. First of all, as we treat each midpoint separately, we do not take full advantage of the potential of our data, for we fail to capitalize on the spatial continuity of the caverns. Therefore, the idea would be to proceed to pattern recognition in higher dimension, such as used for tracking people in videos. Along the same line of thought, we could imagine

combining multicomponent data and evaluating if the combined input of stress field and particle velocity yields better results.

Another idea would be to start with migrated images instead of mere CMP gathers. Indeed, we could run Kirchhoff migration or reverse-time migration and examine the results after applying CWT. This approach would make the starting images closer to the actual geology and facilitate the recognition of the seismic signatures associated with the caverns. Moreover, migration would increase the accuracy of the mapping of the cavern locations in terms of travel-time depth.

Eventually, we could also investigate the concept of detecting other strong heterogeneities, such as salt bodies.

## ACKNOWLEDGMENTS

We would like to thank Gustavo Catao Alves for his help with his elastic modeling code.

## REFERENCES

Alves, G., 2015, Adjoint formulation for the elastic wave equation: Stanford Exploration Project Report, **158**, 133–150.

Berkovitch, A., I. Belfer, Y. Hassin, and E. Landa, 2009, Diffraction imaging by multifocusing: Geophysics, **74**, WCA75–WCA81.

Collobert, R., S. Bengio, and J. Mariéthoz, 2002, Torch: a modular machine learning software library: Technical report, IDIAP.

Fisher, R. A., 1936, The use of multiple measurements in taxonomic problems: Annals of eugenics, **7**, 179–188.

Fomel, S., E. Landa, and M. T. Taner, 2007, Poststack velocity analysis by separation and imaging of seismic diffractions: Geophysics, **72**, U89–U94.

Gill, P. E., W. Murray, and M. H. Wright, 1981, Practical optimization.

Hastie, T., R. Tibshirani, J. Friedman, and J. Franklin, 2005, The elements of statistical learning: data mining, inference and prediction: The Mathematical Intelligencer, **27**, 83–85.

Hsu, C.-W., C.-C. Chang, C.-J. Lin, et al., 2003, A practical guide to support vector classification.

James, G., D. Witten, T. Hastie, and R. Tibshirani, 2013, An introduction to statistical learning, volume **112**: Springer.

Jia, Y., E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, 2014, Caffe: Convolutional architecture for fast feature embedding: Proceedings of the ACM International Conference on Multimedia, 675–678.

Lester, R., E. Liebes, R. Hill, S. Jenkins, A. Makedonov, et al., 2015, Investigating mega-amplitudes in tengiz carbonates through interactive model-building and gaussian beam migration.

Osuna, E., R. Freund, and F. Girosi, 1997, Training support vector machines: an application to face detection: Computer vision and pattern recognition, 1997. Proceedings., 1997 IEEE computer society conference on, 130–136.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al., 2011, Scikit-learn: Machine learning in python: The Journal of Machine Learning Research, **12**, 2825–2830.