Semi-supervised clustering of seismic data

Tue Boesen[a,b,*,1], Eldad Haber[a,b,2] and G. Michael Hoversten[c,3]

[a]University of British Columbia, Vancouver, Canada.
[b]Computational Geoscience Inc., Vancouver, Canada
[c]Chevron Energy Technology Company, San Ramon CA, USA

ARTICLE INFO

Keywords:
Semi-supervised clustering
Oil prospectivity
Graph Laplacian

ABSTRACT

We present a new graph-Laplacian based semi-supervised clustering method. This new approach can be viewed as an extension/improvement of previously published work, both in terms of areas of applicability and computational speed. Our clustering method is capable of handling very large datasets with millions of data points using very limited amounts of labeled data. In this work, we apply our clustering method to 3D oil prospectivity, based on amplitude-versus-angle inversion parameters and borehole information. We cluster the synthetic Life of Field dataset, which has a fault-block constrained central oil reservoir, where we also perform a cross-validation check of the predictive power of our method. Furthermore, we cluster a field dataset, which is characterized by a stratigraphic trapped channelling system. In both cases we find appealing results.

1. Introduction

Semi-supervised learning (SSL) has been rapidly improving in the last few years, as evidenced by the Cifar 10 benchmark test [8], which in the last 5 years have had at least 15 different SSL state-of-the-art methods that have taken the accuracy from 79.6 % [13] to an impressive 97.3 % [18]. These rapid improvements, combined with its broad domain of applicability, makes SSL particularly attractive for many problems in science. While the usage of machine learning is growing rapidly, there are still some issues holding it back; one thing is its notorious reputation for being a black box, from which little insight is gained [1]. Secondly, machine learning is often associated with a heavy computational burden, which prohibits a lot of people from using it [7].

In the following, we approach semi-supervised learning from a minimalistic point of view. We create an objective function, which we minimize in order to develop a semi-supervised clustering method based on a graph Laplacian [17]. In this way, we are capable of creating a clustering algorithm with a closed form expression that requires very limited computational power. Our objective function can be thought of as an extension/improvement of the objective developed in [20]. The key difference between our objective function and theirs, is that ours includes a constraint. Without this constraint, the clustering can predict the probable order of classes (i.e. which classes are more/less likely than others), but not their actual probabilities. Furthermore, while the constraint does add extra steps into the calculations, the overall result is a boost in computational speed, due to the fact that the constraint can replace one of the linear systems that would normally need to be solved. This is particularly relevant in our 2 class examples, where the clustering time is nearly halved. Finally, the constraint enables us to handle problems where the known labels contain a mix of different classes, as used in our examples.

[5] used the unconstrained objective given in [20] and combined it with a deep neural network in a pseudo-labelling approach to perform image classification. Their approach achieved state-of-the-art results, and shows how a clustering approach can be combined with a deep neural network. While their results are impressive, their approach does have one issue, since their objective is unconstrained, their clustering return pseudo-probabilities, which should not be used as entropy weights. Our objective function have no such limitations, and could be used in a similar fashion.

Instead of combining our clustering method with a deep neural network and classifying CIFAR10, we chose to highlight the broad applicability of our method. Thus, we show how our method can be applied to predict oil based on a seismic survey. Predicting oil is a highly sought target in petrophysics, and is traditionally regarded as being a notoriously hard problem, due to the convoluted relationship between the parameters

*Corresponding author
tue.boesen@gmail.com (T. Boesen); EldadHaber@gmail.com (E. Haber); hovg@chevron.com (G.M. Hoversten)
ORCID(s):

[1]Main writer of the article and developer of code.
[2]Conceptualized the method and theorized how to apply the method to seismic data.
[3]Supplied the seismic datasets, as well as expert knowledge on the data and results.

typically derived from a seismic survey and oil. Previous somewhat similar approaches include: [19] who predict oil reservoirs using a fusing of genetic algorithms, simulated annealing and neural networks. [6] who predict oil bearing sand using seismic inversions combined with 3D geologic modelling and petrophysical relations. [12] who utilize a Markov chain Monte Carlo method and combine it with a naive Bayesian classifier, which enables them to bin the well production and produce an oil prediction map.

A standard inversion of a seismic survey gives a set of parameters discretized in a 3D volume. In the framework of machine learning, each point in this 3D volume can be thought of as an unlabelled data point, with the parameters generating the features of each data point. Boreholes intersecting the data volume can be utilized to generate labelled data. Well-logs from boreholes are prohibitively expensive to get in comparison to seismic data, so typically the amount of labelled data will be severely limited in comparison to the amount of unlabelled data, which is exactly the domain of semi-supervised learning.

1.1. Our contribution

We create a simple yet effective semi-supervised clustering method, with a new improved objective function that both speeds up the computational time, and increases accuracy. We furthermore apply this approach to oil prospectivity, which is not typically a field such approaches have been applied to.

2. Preliminaries

This section formulates the semi-supervised learning problem, and introduces how a graph Laplacian is created and can be used as a regularizer.

2.1. Problem setup

Assume that we are given a dataset $\mathbf{X} \in \mathbb{R}^{n \times l}$, where n is the number of data points, and l is the number of features. Hence, each row of the matrix represents a data point. Let $\mathbf{X}_k$, be a known labelled subset of $\mathbf{X}$, with prior associated label probabilities $\mathbf{U}_k^{obs} \in \mathbb{R}^{n_k \times n_c}$, where n_c is the number of classes in the data, and n_k is the number of known labelled points. The goal in semi-supervised learning is to find a probability matrix $\mathbf{U} \in \mathbb{R}^{n \times n_c}$, which matches $\mathbf{U}_k^{obs}$ on the known subset and gives a reasonable estimate of the probabilities on all the remaining data points.

2.2. Graph-Laplacian

Graph Laplacians are often used in both unsupervised learning [15] and semi-supervised learning [5] to give an approximate similarity measure between various datapoints.

For a given dataset $\mathbf{X} \in \mathbb{R}^{n \times l}$, suppose we are given two associated vectors $\mathbf{y}_i, \mathbf{y}_j$ residing in a latent feature space $\mathbf{Y} \in \mathbb{R}^{n \times d}$, and we have a distance measure $\Psi : \mathbb{R}^d \to \mathbb{R}$. We can then define an adjacency matrix $\mathbf{A}$ as:

$$\mathbf{A}_{ij} = \begin{cases} 1, \text{ if } \Psi(\mathbf{y}_i, \mathbf{y}_j) \text{ is among the } m \\ \quad \text{smallest distances } \forall j, \text{ given } i. \\ 0, \text{ otherwise.} \end{cases} \tag{1}$$

Where m is the number of nearest neighbours. Thus the adjacency matrix contains the connection information between each point and its m nearest neighbours. Note that in general the adjacency matrix is directional, and hence not symmetric, which is not desirable in our case. For this reason, we create and use the symmetrical adjacency matrix, $\mathbf{A}_{\text{sym}}$, defined as:

$$\mathbf{A}_{\text{sym}} = \text{sign}(\mathbf{A} + \mathbf{A}^\top). \tag{2}$$

Based on the adjacency matrix, we define a weight matrix $\hat{\mathbf{W}}$ as:

$$\hat{\mathbf{W}}_{ij}(\mathbf{y}_i, \mathbf{y}_j) = \mathbf{A}_{\text{sym}} \exp\left(-\frac{\Psi(\mathbf{y}_i, \mathbf{y}_j)}{\epsilon}\right), \tag{3}$$

where ϵ is the median distance of all connections defined in the adjacency matrix. It should be noted that the weight matrix can be defined in many other ways, as shown in [5, 17].

Using $\hat{\mathbf{W}}$, we can define the symmetrical normalized graph Laplacian [16]:

$$\mathbf{L} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}} \hat{\mathbf{W}} \mathbf{D}^{-\frac{1}{2}}, \tag{4}$$

where $\mathbf{D}$ is a diagonal matrix with the sum of the rows of $\hat{\mathbf{W}}$ on its diagonal, and $\mathbf{I}$ is the identity matrix.

The graph Laplacian can be used as a regularizer, on a parameter $\mathbf{f} \in \mathbb{R}^n$ by using the fact that [17]:

$$\mathbf{f}^\top \mathbf{L} \mathbf{f} = \frac{1}{2} \sum_{i,j}^{n} \hat{\mathbf{W}}_{ij} \left(\frac{f_i}{\sqrt{d_i}} - \frac{f_j}{\sqrt{d_j}} \right)^2, \tag{5}$$

where d_i is the i'th diagonal element of $\mathbf{D}$. Thus a regularization penalty is incurred if f_i and f_j are different and $\hat{\mathbf{W}}_{ij}$ is not negligible. In other words, the regularization penalizes points that are close in our chosen metric and have different values in $\mathbf{f}$.

3. Method

3.1. Creating an objective function

Our objective function is given as:

$$\phi(\mathbf{Y}) = \frac{\alpha}{2n} \text{Tr}\left[\mathbf{Y}^\top \mathbf{L} \mathbf{Y}\right] + \frac{1}{2} ||\mathbf{Y} - \mathbf{Y}^{obs}||_W^2$$

$$\text{st. } \mathbf{Ye} = 0, \tag{6}$$

with α being a scalar hyper-parameter, determining the relative strength between the two terms. $\mathbf{L}$ is the symmetric-normalized graph Laplacian [17], $\mathbf{e}$ is a vector of ones, and $\mathbf{Y}$ is a pseudo-probability matrix, connected with the normal probability $\mathbf{U}$ through a softmax normalization:

$$\mathbf{U} = \exp(\mathbf{Y}) \oslash \left(\exp(\mathbf{Y})\mathbf{e}\mathbf{e}^\top\right). \tag{7}$$

The idea is that a minimization of (6) will lead to reasonable label pseudo-probabilities, since the second term penalizes known data points that do not match their label pseudo-probabilities, while the first term connects all points with their most similar points and encourage them to have similar label probabilities, as shown in (5). The constraint ensures that each probability corresponds to exactly one pseudo-probability, hence there exist a bijective mapping between the two spaces, which specifically enables us to find $\mathbf{Y}^{obs}$ based on $\mathbf{U}^{obs}$ as:

$$\mathbf{Y}^{obs} = \log(\mathbf{U}^{obs}) - \frac{\log(\mathbf{U}^{obs})\mathbf{e}}{\mathbf{e}^\top\mathbf{e}} \otimes \mathbf{e}^\top. \tag{8}$$

Had we not included the constraint, then the transformation of $\mathbf{U}^{obs}$ into $\mathbf{Y}^{obs}$ would only be defined up to a constant, which should be chosen in a consistent way with the solution returned when solving the objective function. But without the constraint the solution is free to return any constant. Thus without the constraint, we would have a system where the input depends on information from the output, and since we do not have that our solution would be inconsistent.

3.2. Solving the objective function

We seek to find the $\mathbf{Y}$ that minimize the objective function given in (6). In order to ensure that the constraint is fulfilled we can explicitly build the constraint into our solution variable. Let $\mathbf{V}$ be the inherently unconstrained solution to (6). From this we create the inherently constrained solution, $\mathbf{Y}$:

$$\mathbf{Y} = \mathbf{VC}, \tag{9}$$

where $\mathbf{C}$ is known as a centering matrix, and is given as:

$$\mathbf{C} = \mathbf{I} - \frac{\mathbf{e}\mathbf{e}^\top}{\mathbf{e}^\top\mathbf{e}}. \tag{10}$$

From $\mathbf{C}$'s construction it follows that:

$$\mathbf{Ce} = 0, \mathbf{C}^2 = \mathbf{C}. \tag{11}$$

By applying (11) to (9) we get:

$$\mathbf{Ye} = 0, \mathbf{YC} = \mathbf{Y}. \tag{12}$$

We minimize the objective function given in (6) by finding its derivative:

$$\begin{aligned}\frac{\partial\phi(\mathbf{Y})}{\partial\mathbf{V}} &= \left(\frac{\alpha}{n}\mathbf{LY} + \mathbf{W}(\mathbf{Y} - \mathbf{Y}^{obs})\right)\frac{\partial\mathbf{Y}}{\partial\mathbf{V}} \\ &= (\frac{\alpha}{n}\mathbf{L} + \mathbf{W})\mathbf{Y} - \mathbf{WY}^{obs}\mathbf{C} \\ &= 0 \\ \Rightarrow \mathbf{Y} &= \left(\frac{\alpha}{n}\mathbf{L} + \mathbf{W}\right)^{-1}\mathbf{WY}^{obs}\mathbf{C},\end{aligned}$$

where $\mathbf{W}$ is the diagonal normalized weight matrix, which fulfils the normalization, $\text{Tr}(\mathbf{W}) = 1$, and has zeros on the diagonal for all unknown points. Note that this weight matrix has no connection to the one defined in (3), which is the weight matrix used to create the graph-Laplacian. This system is positive semidefinite, but unfortunately often ill-conditioned and thus needs to be stabilized. Stabilization is achieved by adding a small identity to the system, enough to stabilize the system, but not enough to disturb the solution. With the added identity our final solution looks like:

$$\mathbf{Y} = \left(\frac{\alpha}{n}\mathbf{L} + \mathbf{W} + \beta\mathbf{I}\right)^{-1}\mathbf{WY}^{obs}\mathbf{C}, \tag{13}$$

where β is a system dependent scalar, depending on the size and conditioning of the system. For the seismic problems we have explored thus far, a β somewhere between $10^{-4} - 10^{-8}$ lead to stable solutions. We solve this system using preconditioned conjugate gradient, where we use a LU splitting as preconditioner [14]. Note that the system is decoupled over the classes, so each class can be solved independently. Furthermore, since the solution by construction fulfils the constraint $\mathbf{Ye} = 0$, we only have to solve (13) for $n_c - 1$ classes.

So, the solution strategy is to solve (13) for $n_c - 1$ classes, use the constraint $\mathbf{Ye} = 0$ to get the last class pseudo-probability, and then convert the pseudo-probabilities to probabilities using the softmax normalization given in (7).

3.3. Simple example

Before we move on to more complicated examples, we start with a simple example that intuitively highlights how the method works. Consider Figure 1, which shows a noisy clustering problem with 3000 data points and 3 classes. Each data point is characterized by 2 features, hence $\mathbf{X} \in \mathbb{R}^{3000\times2}$, while our 3 observed datapoints are given as:

$$\mathbf{U}_k^{obs} = \begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{bmatrix}. \tag{14}$$

$\mathbf{U}_k^{obs}$ is extended into $\mathbf{U}^{obs} \in \mathbb{R}^{3000\times 3}$, by inserting the labelled datapoints on their associated rows in the full dataset, and setting all other rows to 0 (actually all the other rows can be set to anything, since they will be insignificant due to the diagonal weight matrix, which only gives a non-zero weight to the labelled datapoints). With $\mathbf{U}^{obs}$ created, we can create the observed pseudo-probabilities $\mathbf{Y}^{obs}$ through (8).

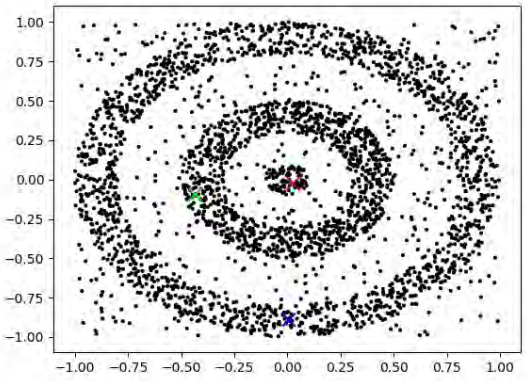
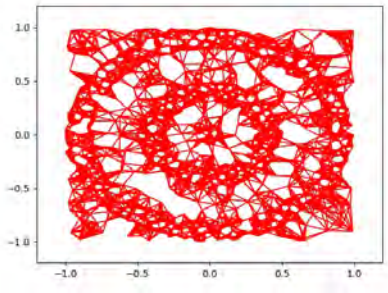


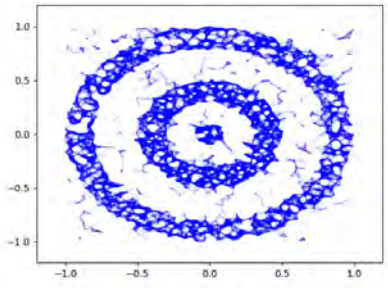
Figure 1: Data visualization of a 2D dataset with 3000 points and 3 clusters. Each data point is represented by a black dot. The 3 labelled data points are shown as crosses, 1 from each class, differentiated by their different colors.

We create the adjacency matrix with 9 nearest neighbours, and create the symmetric normalized graph Laplacian as described in (4), both can be seen visualized in Figure 2.

With the graph Laplacian and observed pseudo-probabil created, we only need to set the hyper parameters. In this case, we set $\alpha = 100$ and $\beta = 10^{-3}$ and cluster the dataset with (13). The results of the clustering can be seen in Figure 3. Note how the points close to a labelled point are similar in color, while points far away, yet in the same cluster are more mixed in their color, which means that those points are significantly more uncertain, which intuitively makes sense, given how we expect the graph Laplacian to spread information to neighbouring points.



(a) Adjacency matrix visualized



(b) Laplacian visualized

Figure 2: Adjacency and graph Laplacian visualized. (a) Shows all the connections in the adjacency matrix. (b) shows the connections in the graph Laplacian with the opacity determined by the strength of the connection.

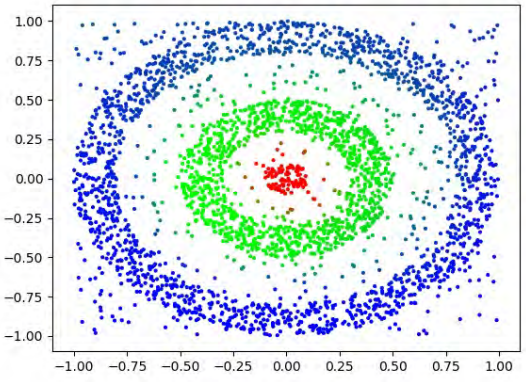


Figure 3: Shows the probability for each point to belong to each of the 3 clusters. Each class is designated by a principal color (red/green/blue) and the color of a point indicates its probability to belong in each of the 3 classes.

4. Experiments

We have used our clustering method to predict oil content in a 3D volume based on seismic data. In the following, we show a synthetic and a field example. Both examples follow very similar data preparation steps, as outlined below, with any deviation noted in the individual examples.

4.1. Data preparation

In both examples we are given a data volume, where each data point in this data volume consist of five inverted parameters, provided from an amplitude versus angle inversion: density, porosity, lithology, VpVs, and acoustic impedance.

We standardize each parameter by setting the mean to zero and the variance to one. As features, we utilize the value and x/y/z-neighbours mean and difference of each of the 5 parameters given to us. This means that each parameter is represented by 7 features that gives information about its value as well as gradient. Finally, for added data-locality we chose to include the standardized x,y,z coordinates as features, with a variance of 7/3. Hence the coordinates have a combined weight similar to each of the 5 other parameters in the feature space. In total this leads to each data point having 38 features. The graph Laplacian is based on the approximate nearest neighbors approach known as Hierarchical Navigable Small World graphs [9], with an ℓ_2 metric and 40 neighbours.

We assign each data point two classes, oil and no-oil, and assume that the oil probability is linearly correlated with oil volume. As labelled data, we utilize the data points closest to each borehole in each layer, and assign oil probability based on the well-log information. So the total oil production of a borehole will be distributed to all intersecting labelled data points, based on their fractional oil-sand content. For the injection boreholes, we only label points containing 0 % oil-sand, which gets labelled as 0 oil probability points. On top of the labelled data points associated with boreholes, we also apply boundary constraints. We set the oil content in the boundary points on all six sides to zero with a weight of 10^{-3} (all other labelled points have a weight of 1).

4.2. Clustering methodology

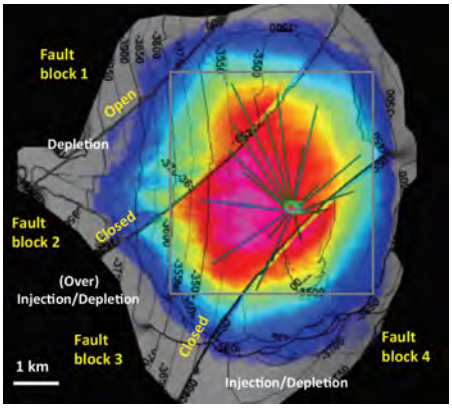
For the clustering, we set $\alpha = 100$, $\beta = 10^{-7}$, which gives a reasonable balance between matching the labelled datapoints and fulfilling the regularization constraints.

From the clustering we get a relative oil prediction in each data point. These oil predictions give a map of relatively high and low oil areas, but should generally not be used to give absolute oil volume predictions directly. In order to estimate the accuracy of our clustering, we assume that each borehole can be approximated as a single vertical drilling, with its horizontal location given as the mean horizontal location of all points of the borehole intersecting the data volume. Secondly, we assume that the oil production in a borehole can be found as:

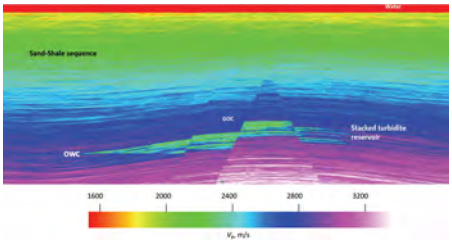
$$O = \frac{1}{2}\min(\mathbf{p}) + \frac{1}{2n_x}\mathbf{p}^\top \mathbf{w}, \tag{15}$$

where $\mathbf{p}$ is a vector containing the vertically summed oil predictions of all points within 100 m of the borehole, n_x is the number of points in $\mathbf{p}$ and $\mathbf{w}$ is the weight of each point, which is given as: $\mathbf{w} = \frac{1}{1+\mathbf{d}}$, where $\mathbf{d}$ is the point distance from the well.

4.3. Synthetic example - SEAM Life of Field



(a) Top-down oil view



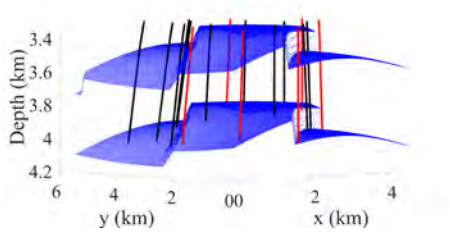
(b) Vp Side-view

Figure 4: Overview of the full SEAM Life of Field dataset, taken and modified from [11]. Figure 4a shows a top down view of the fault lines, as well as the 17 boreholes that goes through the reservoir. The superimposed color scale shows vertically summed oil volume, with purple being high production and dark blue being low. The gray rectangle indicates our modelling area. Figure 4b shows a side view of the full data volume. The full SEAM Life of Field data volume is 12.5 km x 12.5 km x 5 km (x,y,z).

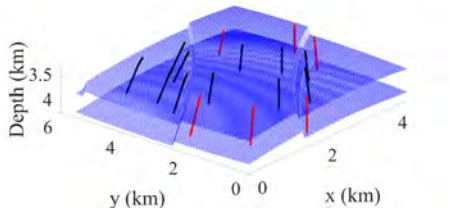
We apply our clustering approach to the synthetic

SEAM Life of Field dataset [11, 10], which is a highly realistic simulation of conditions found in a typical oil reservoir in the Gulf of Mexico. SEAM life of Field contains: a gas cap, an oil leg, a brine section below, and four fault blocks - as shown in Figure 4. Our subset of the data covers the central region of the oil zone, and is shown in Figure 5. Our inverted parameter dataset consist of 181 x 221 x 157 data points (x,y,z). Furthermore, we have 17 boreholes going through our dataset - 11 production wells, and 6 injection wells, each with a well-log. Since some of the wells are very close to the edge of the data volume, we relax the boundary conditions to a weight of 10^{-4}. Furthermore, since the oil reservoir is close to top/bottom of our data volume we only include boundary conditions on the four sides, but not top and bottom.

In total this leads us to have 6,280,157 data points, 2068 labelled data points from boreholes, and 126,228 weakly labelled boundary points.



(a) Side-view



(b) Top-down'ish view

Figure 5: 3D overview of our data from SEAM life of Field, from 2 different angles. The two blue planes are the top and bottom layer of our data, while the black lines are production wells and the red lines are injection wells.

4.3.1. Results & discussion

Figure 6 shows a 2D slice of the clustering prediction in individual data points as well as the values of the five parameters used to create the clustering. We note that the clustering looks sensible, and not trivially derived from any of the five parameters.

From the clustering we get an oil prediction in each data point, which shows a clear central oil deposit, bounded by the fault lines. The exact nature of this oil prediction is rather abstract, since it is based on oil production in the boreholes it would be natural to think of it as an oil production map. However, that assumption can lead to some dangerously wrong conclusion. Instead the clustering result should be considered as an relative map of oil content/volume in the subsurface that lead to the oil production we have currently seen in the boreholes. Figure 7 shows the vertically summed oil predictions, scaled to match the boreholes. Comparing the oil reservoir in Figure 7 with the oil given in Figure 4, we see that our oil prediction looks sensible, especially at the fault line contrast, which have proven difficult in other approaches [12]. The individual well production and prediction can be seen in Table 1, from which we can see that the average prediction error is 22.4%.

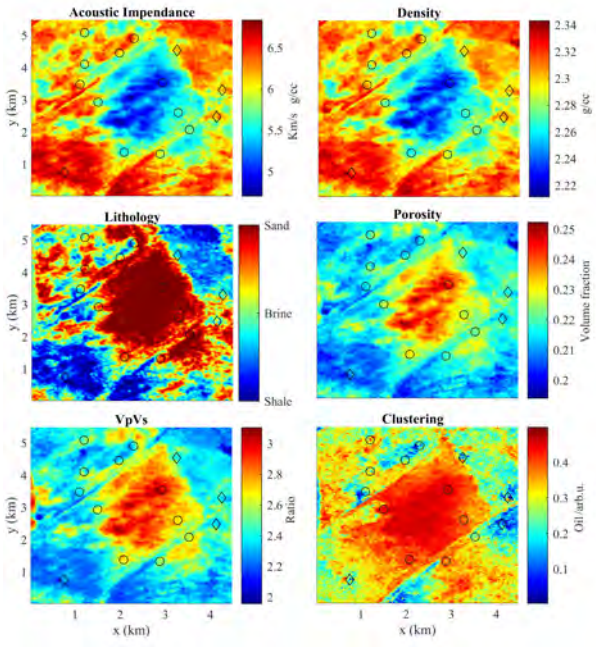


Figure 6: Showing a 2D slice of the data. The location where the wells pass through the layer is marked with circles for production wells and diamonds for injection wells. Note that several injection wells are not shown here, this is because their well logs do not show shale in this layer, and hence they are not used as labelled data in this layer.

4.3.2. Cross-validation

In order to test the stability and predictive capabilities of our approach we perform a cross validation, where we remove 4 of the 11 production wells, and see how well our approach predicts the oil. This approach leads to 330 different scenarios, which we all cluster. The average error across all wells was found

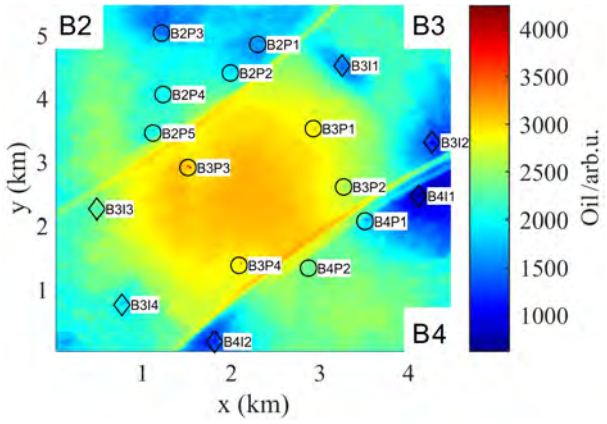


Figure 7: Vertically summed oil prediction for SEAM. The mean horizontal location of each well is marked with circles for production wells and diamonds for injection wells. The wells are further identified by a name which is derived from the fault block the well is in, as well as a unique number.

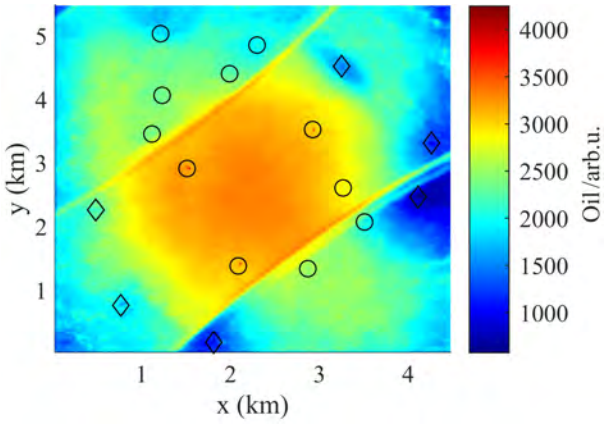


Figure 8: Average vertically summed oil prediction for 330 different cross-validation clusters on SEAM. The mean horizontal location of each well is marked with circles for production wells and diamonds for injection wells. The well ID's can be found in Figure 7.

Table 1
Well predictions. The well ID follows the naming convention used in Figure 7, where a well is identified by the fault block it lies in and a number. The produced oil is normalized to the largest production well. The C-errors refer to the cross-validation error for each well. Both errors are given in percent.

ID	Produced	Predicted	Error	C-Error
B2P1	0.40	0.37	8.78	22.80
B2P2	0.40	0.43	7.45	26.34
B2P3	0.40	0.33	18.75	26.18
B2P4	0.39	0.47	19.50	35.93
B2P5	0.36	0.46	30.14	51.06
B3P1	0.75	0.68	8.87	13.65
B3P2	0.42	0.49	18.45	29.24
B3P3	1.00	0.72	28.35	32.61
B3P4	0.75	0.68	9.50	12.72
B4P1	0.20	0.34	70.99	109.52
B4P2	0.40	0.50	25.51	37.68
Mean	-	-	22.39	36.15

to be 36.15%, as seen in Figure 1, from which the average error for each well can also be seen. Figure 8 shows the average oil prediction across all 330 cross-validation scenarios. Comparing Figure 7 and Figure 8 we see differences in the amounts predicted, but the overall trends are similar, which suggest that our clustering approach is rather stable and can be used to predict future drillings.

4.4. Field example - west African reservoir

Our second example is an offshore West African Field (WAF) [4]. Compared to the synthetic example, which was characterized by the sharp fault line boundaries, WAF is less structurally controlled, and instead dominated by stratigraphic trapping in channelling systems. WAF's data consist of 476 x 94 x 181 data points (x,y,z), with 6 boreholes intersecting the dataset - 3 production wells, and 3 injection wells, each with a well-log.

In total this leads us to have 8,098,664 data points, 523 labelled data points from boreholes, and 289,844 weakly labelled boundary points.

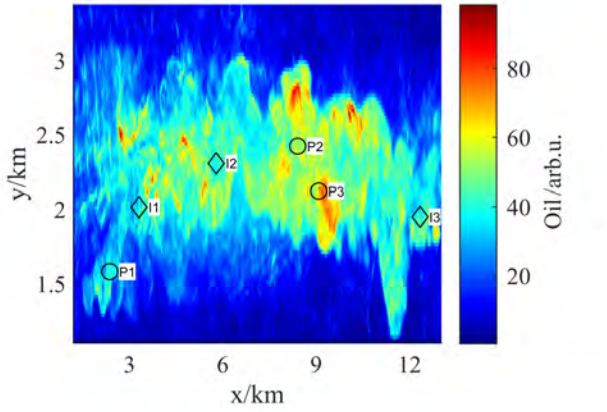


Figure 9: Vertically summed oil prediction for WAF. The mean horizontal location of each well is marked with circles for production wells and diamonds for injection wells.

4.4.1. Results & discussion

From the clustering we find the vertically summed oil volume shown in Figure 9, from which we can get

Table 2
Well predictions. The well ID follows the naming convention used in Figure 9. The produced oil is normalized to the largest production well.

ID	Produced	Predicted	Error (%)
P1	0.35	0.40	13.79
P2	0.43	0.60	41.32
P3	1.00	0.79	23.02
Mean	-	-	26.71

an overview of the oil reservoir. Compared to the synthetic example we see much more heterogeneity in this oil field, which fits the expected stratigraphic trapping and channelling systems. In order to see these heterogeneities, we show slices of the clustering data in Figure 10, from which we can see the heterogeneous pockets of oil and their channelling systems.

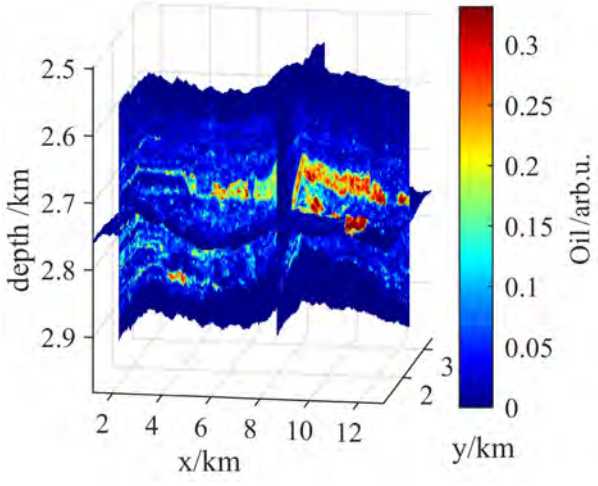


Figure 10: 3 Slices of the cluster results on WAF.

Based on the vertically summed up oil predictions, we can estimate the accuracy of our prediction in the wells, as previously described. We find an average error of 26.71%, as shown in Table 2, which is quite high, but not unexpected given the heterogeneity of the field data and the limited amount of labelled data available to guide our clustering.

5. Performance

The approach presented here offers a simple method for predicting oil that can be done on even modest computers. Calculating the adjacency matrix is the heaviest computational task, but finding approximate nearest neighbours is a field in itself, where highly efficient plug-and-play solutions luckily exist. The adjacency calculation takes us a few hours to days, for 6 million samples with 40 neighbours, depending on the accuracy of these neighbours. A large saving grace for this method is that the adjacency matrix typically only needs to be found once. When the adjacency matrix is computed, you can run as many different clusters as you want on it, where you can adjust the labelled data or the boundary conditions. Each clustering can be done in roughly 10 minutes on a single core on an Intel Xeon Processor E5-2670.

6. Discussion & future work

The semi-supervised clustering approach we have presented here, is conceptually simple, but it still contains a lot of choices that are worth discussing. For the adjacency matrix there are several ways to define which elements to include [17], we tested both k-nearest neighbours and mutual k-nearest neighbours, and both option gave almost identical clusters. Hence we chose the k-nearest neighbours, since that option has much less likelihood of making the graph-Laplacian disjointed. Similarly we tested various numbers of nearest neighbours. Here we saw variations if the number of neighbours became too low, but for 40 and above, we saw no change in the clustering only in the computational time. The next choice comes when creating the graph-Laplacian weight matrix, in this case we chose an ℓ_2 metric, but another popular choice is angular distances as used in [5]. For oil predictions we wanted to include data locality, which makes the angular distance metric a less sensible choice.

We recognize that the work presented here would benefit greatly from a comparison to other similar oil predicting approaches, but unfortunately we are not aware of any published approaches with which we can do a direct comparison, especially not on any of the datasets which we have access to. The closest we can get to such a comparison is to the work shown in [12], however their results are preliminary and does not show enough detail to make a fair comparison.

Looking at the clustering results it is interesting how sharp, yet structurally consistent, the clustering appears to be. Especially the small oil-reservoirs and oil-bearing channels in the field example are interesting to see. If our clustering is accurate, then it suggest several new possible drilling locations. For instance, it would be sensible to place an injection well at the same x coordinate as P2 but around $y = 3$ km, as this would probably push the large oil reservoir between those two locations into the production well. However, while the clustering we found looks sensible, the error is still relatively high, and as such should not be trusted more than the accuracy permits, so drilling based solely on this would

likely be inadvisable. Regarding the error, some of the error stems from the data itself and as such is likely to remain despite improving the clustering. However some of the error is due to the overly simple model for borehole production given in (15), and could probably be improved significantly, especially in the field example where we have high heterogeneity. A more refined borehole production model should be linked with the total amount of oil in the oil chambers intersecting the borehole, and perhaps even account for directional pressure gradients made by injection wells.

One other thing we have tried, but haven't included here is to take the unstructured mesh that the seismic data resides on, and interpolate it onto an octree mesh [2, 3]. While this was possible, it was not very practical for these particular problems, since they originate on a highly irregular mesh that either requires extensive extrapolation or looses significant amounts of data when using the inscribed data cube. For other problems that exists on a more regular mesh the octree approach could be beneficial.

In this work we still rely on standard inversions to produce our input parameters. It would be interesting to see how the clustering would work if used with the raw seismic data as features rather than the inverted parameters.

7. Conclusions

We have developed a simple semi-supervised clustering approach that can be used in a wide range of applications. We applied our clustering to seismic datasets with the goal of finding oil, which is phenomenologically a hard task. We managed to find reasonable results with simple assumptions and limited computational resources. For future work, we have illustrated various ways to increase the accuracy, or expand the domain of applicability.

8. Acknowledgements

We thank Chevron Energy Technology Company and Computational Geoscience Inc. for permission to publish this work and for financial support.

References

[1] Castelvecchi, D., 2016. Can we open the black box of ai? Nature News 538, 20.

[2] Haber, E., Heldmann, S., 2007. An octree multigrid method for quasi-static Maxwell's equations with highly discontinuous coefficients. Journal of Computational Physics 223, 783–796. doi:10.1016/j.jcp.2006.10.012.

[3] Horesh, L., Haber, E., 2011. A Second Order Discretization of Maxwell's Equations in the Quasi-Static Regime on OcTree Grids. SIAM Journal on Scientific Computing 33, 2805–2822.

[4] Hoversten, G., Royle, A., Chen, J., Myer, D., 2017. Mcmc inversion of offshore west africa ava data, in: 79th EAGE Conference and Exhibition 2017, European Association of Geoscientists & Engineers. pp. 1–5.

[5] Iscen, A., Tolias, G., Avrithis, Y., Chum, O., 2019. Label propagation for deep semi-supervised learning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5070–5079.

[6] Jian, W., Fanhua, L., 2009. Prediction of oil-bearing single sandbody by 3d geological modeling combined with seismic inversion. Petroleum exploration and development 36, 623–627.

[7] Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T., 2019. Analyzing and improving the image quality of stylegan. arXiv preprint arXiv:1912.04958 .

[8] Krizhevsky, A., Hinton, G., et al., 2009. Learning multiple layers of features from tiny images. Technical Report. Citeseer.

[9] Malkov, Y.A., Yashunin, D.A., 2018. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. IEEE transactions on pattern analysis and machine intelligence .

[10] Oppert, S., Stefani, J., Eakin, D., Halpert, A., Herwanger, J.V., Bottrill, A., Popov, P., Tan, L., Artus, V., Oristaglio, M., 2017. Virtual time-lapse seismic monitoring using fully coupled flow and geomechanical simulations. The Leading Edge 36, 750–768.

[11] Oristaglio, M., 2016. Seam update: Integrated reservoir and geophysical modeling: Seam time lapse and seam life of field. The Leading Edge 35, 912–915.

[12] Powers, H., Trainor-Guitton, W., Hoversten, G.M., 2018. Classification of total oil production of wells in seam life of field from stochastic ava inversion attributes via machine learning, in: SEG Technical Program Expanded Abstracts 2018. Society of Exploration Geophysicists, pp. 2131–2135.

[13] Rasmus, A., Berglund, M., Honkala, M., Valpola, H., Raiko, T., 2015. Semi-supervised learning with ladder networks, in: Advances in neural information processing systems, pp. 3546–3554.

[14] Saad, Y., 2003. Iterative methods for sparse linear systems. volume 82. siam.

[15] Shaham, U., Stanton, K., Li, H., Nadler, B., Basri, R., Kluger, Y., 2018. Spectralnet: Spectral clustering using deep neural networks. arXiv preprint arXiv:1801.01587 .

[16] Shi, J., Malik, J., 2000. Normalized cuts and image segmentation. Departmental Papers (CIS) , 107.

[17] Von Luxburg, U., 2007. A tutorial on spectral clustering. Statistics and computing 17, 395–416.

[18] Xie, Q., Dai, Z., Hovy, E., Luong, M.T., Le, Q.V., 2019. Unsupervised data augmentation. arXiv preprint arXiv:1904.12848 .

[19] Yu, S., Zhu, K., Diao, F., 2008. A dynamic all parameters adaptive bp neural networks model and its application on oil reservoir prediction. Applied mathematics and computation 195, 66–75.

[20] Zhou, D., Bousquet, O., Lal, T.N., Weston, J., Schölkopf, B., 2004. Learning with local and global consistency, in: Advances in neural information processing systems, pp. 321–328.