Fast Petroleum Well Drilling Monitoring Through Optimum-Path Forest

Ivan R. Guilherme¹, Aparecido N. Marana², João P. Papa², Giovani Chiachia³, Alexandre X. Falcão³, Kazuo Miura⁴, Marcus V. D. Ferreira⁴, Francisco Torres⁴

¹Department of Statistics, Applied Mathematics and Computing - Univ Estadual Paulista
²Department of Computing, UNESP - Univ Estadual Paulista, Brazil
³Institute of Computing, State University of Campinas, Brazil
⁴Brazilian Petroleum - PETROBRÁS, Brazil

ivan@rc.unesp.br, {papa,nilceu}@fc.unesp.br, giovanichiachia@gmail.com, afalcao@ic.unicamp.br, {kazuo.miura,mvdferreira,fatorres}@petrobras.br

doi: 10.4156/jnit.vol1.issue1.7

Abstract

Automatic inspection of petroleum well drilling has become paramount in the last years, mainly because of the crucial importance of saving time and operations during the drilling process in order to avoid some problems, such as the collapse of the well borehole walls. In this paper, we extended another work by proposing a fast petroleum well drilling monitoring through a modified version of the Optimum-Path Forest classifier. Given that the cutting’s volume at the vibrating shale shaker can provide several information about drilling, we used computer vision techniques to extract texture informations from cutting images acquired by a digital camera. A collection of supervised classifiers were applied in order to allow comparisons about their accuracy and efficiency. We used the Optimum-Path Forest (OPF), EOPF (Efficient OPF), Artificial Neural Network using Multilayer Perceptrons (ANN-MLP) Support Vector Machines (SVM), and a Bayesian Classifier (BC) to assess the robustness of our proposed schema for petroleum well drilling monitoring through cutting image analysis.

Keywords: Optimum-Path Forest, Petroleum Well Drilling, Artificial Intelligence

1. Introduction

Offshore petroleum well drilling is an expensive, complex and time-consuming operation and it demands a high qualification level from the drilling executors. One of the trends of the oil industry is the application of real time measurements and optimization of production operations with the purpose of guaranteeing a safe and effective/low cost drilling execution. Nowadays, there exists several data acquisition systems for well drilling monitoring, in which a large amount of data is generated at each time. One of these systems is the Mud-logging, which is responsible for measuring a set of mechanical and geological parameters. The data generated by Mud-Logging, together with the cutting analysis produced during the drilling operation, allows the drilled soil lithological analysis [24], which are carried out in deep intervals defined by geology. The generated cutting samples available at the vibrating shale shakers are examined by some expert technician in order to evaluate whether a problem is occurring during the drilling process. Generally, these cuttings have similar shapes and sizes in typical situations, and any distortion beyond the known normal patterns can indicate the presence of some anomaly, such that the collapse of the well borehole walls.

Some works have been dedicated for monitoring the well drilling process [8,12, 18,22], but none of them were guided by the cutting image analysis. Frantiek et al. [8] proposed to monitor the rock disintegration process at drilling with the application of acoustic signal. A Fourier Transform of the generated signal was performed for further statical analysis. Serapião et al. [22] used artificial immune systems for classification of several stages in petroleum drilling. Also, the use of artificial intelligence techniques in drilling engineering is not new. Coelho et al. [3], Fonseca et al. [9] and Yilmaz et al. [26] used neural networks for drilling activities monitoring, and Fonseca et al. [10] applied Support Vector Machines for classification of petroleum well drilling operations.

An artificial neural network with multi-layer perceptrons (ANN-MLP), for example, can address linearly, piecewise linearly and non linearly separable problems, but not non separable situations [13].

77
As an unstable classifier, collections of ANN-MLP [14] can improve its performance up to some unknown limit of classifiers [21]. Support vector machines (SVMs) have been proposed to overcome the problem, by assuming linearly separable classes in a higher dimensional feature space [25]. Its computational cost rapidly increases with the training set size and the number of support vectors. As a binary classifier, multiple SVMs are required to solve a multi-class problem [6]. Tang and Mazzoni [23] proposed a method to reduce the number of support vectors in the multi-class problem. Their approach suffers from slow convergence and high computational cost, because they first minimize the number of support vectors in several binary SVMs, and then share these vectors among the machines. Panda et al. [20] presented a method to reduce the training set size before computing the SVM algorithm. Their approach aims to identify and remove samples likely related to non-support vectors. However, in all SVM approaches, the assumption of separability may also not be valid in any space of finite dimension [4].

Recently, a novel graph-based classifier that reduce the pattern recognition problem as an optimum-path forest (OPF) computation in the feature space induced by a graph was presented [27]. This kind of classifier does not interpret the classification task as a hyperplanes optimization problem, but as a combinatorial optimum-path computation from some key samples (prototypes) to the remaining nodes. Each prototype becomes a root from its optimum-path tree and each node is classified according to its strongly connected prototype, that defines a discrete optimal partition (influence region) of the feature space. The OPF classifier has some advantages with respect to the aforementioned classifiers: (i) is free of parameters, (ii) does not assume any shape/separability of the feature space and (iii) runs training phase faster. The OPF classifier has been demonstrated to be superior than ANN-MLP and similar to SVM, but much faster.

As aforementioned, the previous works use only the information provided by sensors to monitor and assess the well drilling process. Some important information, such that the cutting volume at the vibrating shale shakers, is also an important data about the drilling conditions, and can be measured using the cutting image analysis. Marana et al. [29] have proposed a system for petroleum well drilling monitoring based on cutting image analysis. The basic idea was to extract texture features from images acquired by a camera positioned at the vibrating shale shaker for further classification into three types of volume concentration: (i) low, (ii) medium and (iii) high. Results using OPF, SVM, ANN-MLP and a Bayesian Classifier demonstrated the robustness of the system with 99% of mean accuracy for all classifiers, being the former the fast one. Recently, Papa et al. [28] have proposed a modification in the OPF algorithm called EOPF (Efficient OPF), which can achieve the same accuracy of the traditional OPF, but much faster. Thus, the main goal of this paper is to extend the work of Marana et al. [29] by validating EOPF in the context of petroleum well drilling monitoring using cutting image analysis. The remainder of this paper is organized as follows. Section 2 presents the system architecture proposed by Marana et al. [29]. Section 3 and 4 present, respectively, the classification by Optimum-Path Forest theory and the experimental results, in which we compared OPF against SVM, ANN-MLP and a Bayesian Classifier (BC) [2]. A second round of experiments were conducted in order to assess the efficiency of EOPF against OPF. Finally, Section 5 discusses the conclusions.

2. Proposed System

This section presents the intelligent system architecture proposed by Marana et al. [29] for drilling monitoring using cutting image analysis, which is composed by two modules: (i) image acquisition and (ii) data analysis (Figure 1). During the well drilling process, the generated cutting is transported through the vibrating shale shaker until a final repository. The image acquisition system captures cutting images during this transportation step and further send them to the data analysis module, which is responsible for classifying images obtained from the vibrating shale shaker. If these images differ from the usual pattern, the shale operator is notified to inspect the well drilling location.

Nowadays, there not exists an image-based system like the proposed one. Based on the assumption that the cutting information is extremely important for well drilling monitoring, the proposed intelligent system has three main advantages: (i) high accuracy into identifying anomalies during the
drilling operation and (ii) low cost and (iii) non invasive system, due to the fact that it only needs one video-camera and one microcomputer (even a laptop, for example) with the software installed.

Basically, the system operates by classifying each received frame from the digital camera as belonging to one predefined class (Section 4). If this frame represents some kind of drilling anomaly, a technician is requested to monitor the drilling step and even so to intervene in the whole process. Depending on the cutting’s size and shape, a well collapse can be identified in time to avoid further damages. The main idea here, in the future, is to use the information provided by this system into Mud-Logging, aiming its better performance.

![Image Acquisition and Data Analysis](image.png)

**Figure 1.** Proposed system architecture.

### 3. Optimum-Path Forest Classifier

Let $Z_1$, $Z_2$ and $Z_3$ be training, evaluation, and test sets with $|Z_1|$, $|Z_2|$ and $|Z_3|$ samples of a given dataset. As already explained, this division of the dataset is necessary to validate the classifier and evaluate its learning capacity from the errors. $Z_1$ is used to project the classifier and $Z_3$ kept unseen during the project. A pseudo-test on $Z_2$ is used to teach the classifier by randomly interchanging samples of $Z_1$ with misclassified samples of $Z_2$. After learning, it is expected an improvement in accuracy on $Z_3$.

Let $\lambda(s)$ be the function that assigns the correct label $i$, $i=1, 2, \ldots, c$, to any sample $s \in Z_1 \cup Z_2 \cup Z_3$, $S \subseteq Z_1$ be a set of prototypes from all classes, and $v$ be an algorithm which extracts $n$ features from any sample $s \in Z_1 \cup Z_2 \cup Z_3$ and returns a vector $\vec{v}(s)$. The distance $d(s,t) \geq 0$ between two samples, $s$ and $t$, is the one between their corresponding feature vectors $\vec{v}(s)$ and $\vec{v}(t)$. One can use any distance function suitable for the extracted features. The most common is the Euclidean one.

Our problem consists of projecting a classifier which can predict the correct label $\lambda(s)$ of any sample $s \in Z_3$. Training consists of finding a special set $S^* \subseteq Z_1$ of prototypes and a discrete optimal partition of $Z_1$ in the feature space (i.e., an optimum-path forest rooted in $S^*$). The classification of a sample $s \in Z_3$ (or $s \in Z_2$) is done by evaluating the optimum paths incrementally, as though it were part of the forest, and assigning to it the label of the most strongly connected prototype.

#### 3.1. Training

Let $(Z_1, A)$ be a complete graph whose the nodes are the samples and any pair of samples defines an arc in $A=Z_1 \times Z_1$. The arcs do not need to be stored and so the graph does not need to be explicitly represented. A path is a sequence of distinct samples $\pi t=(s_1,s_2,\ldots,s_k)$ with terminus at a sample $t$. A path is said trivial if $\pi t=(t)$. We assign to each path $\pi t$ a cost $f(\pi t)$ given by a connectivity function $f$. A path $\pi t$ is said optimum if $f(\pi t) \leq f(\pi t')$ for any other path $\pi t'$. We also denote by $\pi s \langle s,t \rangle$ the concatenation of a path $\pi s$ and an arc $(s,t)$.

We will address the connectivity function $f_{max}$. 
Fast Petroleum Well Drilling Monitoring Through Optimum-Path Forest

Ivan R. Guilherme, Aparecido N. Marana, João P. Papa, Giovani Chiachia, Alexandre X. Falcão, Kazuo Miura, Marcus V. D. Ferreira, Francisco Torres

\[ f_{\text{max}}(\langle s \rangle) = \begin{cases} 0 & \text{if } s \in S \\ +\infty & \text{otherwise} \end{cases} \]

\[ f_{\text{max}}(\pi, \langle s, t \rangle) = \max \{ f_{\text{max}}(\pi_t), d(s, t) \} \]

(1)

such that \( f_{\text{max}}(\pi_s \langle s, t \rangle) \) computes the maximum distance between adjacent samples along the path \( \pi_s \langle s, t \rangle \). The minimization of \( f_{\text{max}} \) assigns to every sample \( t \in Z_1 \) an optimum path \( P^*(t) \) from the set \( S \subset Z_1 \) of prototypes, whose minimum cost \( C(t) \) is

\[ C(t) = \min_{\pi \in S \cap (Z_1 \setminus A)} \{ f_{\text{max}}(\pi) \} \]

(2)

The minimization of \( f_{\text{max}} \) is computed by Algorithm 1, called OPF algorithm, which is an extension of the general image foresting transform (IFT) algorithm [11] from the image domain to the feature space, here specialized for \( f_{\text{max}} \). As explained in the Section I, this process assigns one optimum path from \( S \) to each training sample \( t \) in a non-decreasing order of minimum cost, such that the graph is partitioned into an optimum-path forest \( P \) (a function with no cycles which assigns to each \( t \in Z_1 \) its predecessor \( P(t) \) in \( P^*(t) \) or a marker nil when \( t \in S \). The root \( R(t) \in S \) of \( P^*(t) \) can be obtained from \( P(t) \) by following the predecessors backwards along the path, but its label is propagated during the algorithm by setting \( L(t) = \lambda(R(t)) \).

**Algorithm 1 - OPF Algorithm**

**Input:** A training set \( Z_1 \), \( \lambda \)-labeled prototypes \( S \subset Z_1 \) and the pair \((v, d)\) for feature vector and distance computations.

**Output:** Optimum-path forest \( P \), cost map \( C \) and label map \( L \).

**Auxiliary:** Priority queue \( Q \) and cost variable \( c_{\text{st}} \).

1. For each \( s \in Z_1 \setminus S \), set \( C(s) \leftarrow +\infty \).
2. For each \( s \in S \), do
   1. \( C(s) \leftarrow 0 \), \( P(s) \leftarrow \text{nil} \), \( L(s) \leftarrow \lambda(s) \), and insert \( s \) in \( Q \).
3. While \( Q \) is not empty, do
   4. Remove from \( Q \) a sample \( s \) such that \( C(s) \) is minimum.
   5. For each \( t \in Z_1 \) such that \( t \neq s \) and \( C(t) > C(s) \), do
      6. Compute \( c_{\text{st}} \leftarrow \max \{ C(s), d(s, t) \} \).
      7. If \( c_{\text{st}} < C(t) \), then
         8. If \( C(t) = +\infty \), then remove \( t \) from \( Q \).
         9. \( P(t) \leftarrow s \), \( L(t) \leftarrow L(s) \) and \( C(t) \leftarrow c_{\text{st}} \).
   10. Insert \( t \) in \( Q \).

We say that \( S^* \) is an optimum set of prototypes when Algorithm 1 minimizes the classification errors in \( Z_1 \). \( S^* \) can be found by exploiting the theoretical relation between minimum-spanning tree (MST) and optimum-path tree for \( f_{\text{max}} \) [12]. By computing an MST in the complete graph \((Z_1, A)\), we obtain a connected acyclic graph whose nodes are all samples of \( Z_1 \) and the arcs are undirected and weighted by the distances \( d \) between adjacent samples. The spanning tree is optimum in the sense that the sum of its arc weights is minimum as compared to any other spanning tree in the complete graph. In the MST, every pair of samples is connected by a single path which is optimum according to \( f_{\text{max}} \). That is, the minimum-spanning tree contains one optimum-path tree for any selected root node.

The optimum prototypes are the closest elements of the MST with different labels in \( Z_1 \). By removing the arcs between different classes, their adjacent samples become prototypes in \( S^* \) and Algorithm 1 can compute an optimum-path forest with minimum classification errors in \( Z_1 \). Note that, a given class may be represented by multiple prototypes (i.e., optimum-path trees) and there must exist at least one prototype per class.

**3.2. Classification**

For any sample \( t \in Z_3 \), we consider all arcs connecting \( t \) with samples \( s \in Z_1 \), as though \( t \) were part of the training graph. Considering all possible paths from \( S^* \) to \( t \), we find the optimum path \( P^*(t) \) from \( S^* \) and label \( t \) with the class \( \lambda(R(t)) \) of its most strongly connected prototype \( R(t) \in S^* \). This path can be identified incrementally, by evaluating the optimum cost \( C(t) \) as
Let the node $s^* \in Z_1$ be the one that satisfies (3) (i.e., the predecessor $P(t)$ in the optimum path $P^*(t)$). Given that $L(s^*)=\lambda_t(R(t))$, the classification simply assigns $L(s^*)$ as the class of $t$. An error occurs when $L(s^*) \neq \lambda_t$. Similar procedure is applied for examples in the evaluation set $Z_2$. In this case, however, we would like to use misclassified samples of $Z_2$ to learn the distribution of the classes in the feature space and improve classification performance on $Z_3$.

3.3. Efficient OPF

The EOPF (Efficient OPF) algorithm proposed by Papa et al. [28] is, essentially, a fast label assignment algorithm (Algorithm 2) together with the traditional OPF training approach (Algorithm 1). Given a test node $t \in Z_3$, the classification test of OPF consists, essentially, in connecting it to all training samples and finding the node $s \in Z_1$ that satisfies Equation 3, i.e., the node $s$ that offers the optimum-path cost to conquer $t$ (Section 3.2). This optimum path is given by $\max$ as path-cost function (Equation 1), which computes the maximum arc weight along a path.

As we have a minimization problem (Equation 3), the value of the cost of each node $s \in Z_1$ has a strong relation with the result of the optimization problem. In such a way, it is reasonable to assume that the training nodes with lower costs have higher probability to attend Equation 3. Given that, it is desirable to have a sorted set $Z'_1$ of training samples according to their increasing order of costs.

Hence, the main idea is, given a test node $t \in Z_2$, the method begins with the first node $k_i \in Z'_1$, $i = 1, 2, \ldots, |Z'_1|$, and evaluates it with $\max$, (i.e., $\text{tmp} \leftarrow \max\{C(k_i), d(k_i, t)\}$, in which $C(k_i)$ is the cost value of node $k_i$ computed in the training phase (Section 3.1) and $d(k_i, t)$ denotes the arc weight between nodes $k_i$ and $t$, until $\text{tmp} \leq C(k_{i+1})$ - the next node in the ordered set). This is because the value of the optimum-path ordered by $k_{i+1}$ to sample $t$ will be, at least, $C(k_{i+1})$.

In Algorithm 2, the main loop (Lines 1-9) performs the classification phase. The inner loop (Lines 4-9) evaluates each node $k_i \in Z'_1$, $i = 1, 2, \ldots, |Z'_1|$ until it reaches the stopping criteria or it evaluates the whole $Z'_1$ (Line 4). Line 5 calculates the $\max$ and Lines 7-8 updates the node cost and label if it is necessary (Line 6).

Algorithm 2 - EOPF Test Phase Algorithm

**Input:** Set $Z'_1$, unlabeled set $Z_2$ and the pair $(t, d)$ for feature vector and distance computations.

**Output:** $\lambda$-labeled test set $Z_2$.

**Auxiliary:** Cost variables $\text{tmp}$ and $\text{mincost}$.

1. For each $t \in Z_2$, do
   2. $i \leftarrow 1$, $\text{mincost} \leftarrow \max\{C(k_i), d(k_i, t)\}$.
   3. $L(t) \leftarrow L(k_i)$.
   4. While $i < |Z'_1|$ and $\text{mincost} > C(k_{i+1})$, do
      5. Compute $\text{tmp} \leftarrow \max\{C(k_{i+1}), d(k_{i+1}, t)\}$
      6. If $\text{tmp} < \text{mincost}$, then
         7. $\text{mincost} \leftarrow \text{tmp}$.
         8. $L(t) \leftarrow L(k_{i+1})$.
   9. $i \leftarrow i + 1$.

4. Experimental Results

We present here two rounds of experiments for automatic cutting image analysis: (i) in the first one we reproduced the results obtained by Marana et al. [29] (Section 4.1), (ii) and in the latter (Section 4.2) we applied EOPF to this task.
4.1. Petroleum Well Drilling Monitoring Through Cutting Image Analysis

In this first stage of our system, we evaluate the amount of cutting volume at the vibrating shale shaker by labeling the images according to three predefined volume classes: E (none concentration of cuttings at the shale - empty), L (low concentration of cuttings) and finally, the class H, which means high concentration of cuttings at the shale. Figure 2 shows some examples from each class of cutting volume. We performed a texture analysis to create the feature vector of each image, which is composed by four Haralick features: energy, entropy, homogeneity and contrast [11]. These features are simply and fast computed based on the co-occurrence matrix of each image and have been demonstrated to be very discriminative in our problem.

As aforementioned, we used for classification purposes a collection of supervised classifiers: OPF, ANN-MLP, SVM and BC, which require a labeled set of samples for training that are used to calibrate the system. The training images were obtained by manually selecting some frames from a video (20 frames/samples from each class) and a technician was requested to classify each frame as belonging to class E, L or H. This same technician was requested again to label the remaining images (20 samples from each class) to compose the test set, in order to allow a quantitative evaluation of the accuracy of each classifier. Essentially, we have a dataset with 120 samples equally distributed among the classes: a training set $Z_1$ with 50% of the samples and a test set $Z_2$ with 50% of the samples. These samples were randomly selected and each experiment was repeated 10 times with different sets $Z_1$ and $Z_2$ to compute the mean accuracy values. The average computational time of each classifier in minutes for training and classification is also reported.

For SVM implementation, we use the LibSVM package [16] with Radial Basis Function (RBF) kernel, parameter optimization and the one-versus-one strategy for the multi-class problem. We use the Fast Artificial Neural Network Library (FANN) [19] to implement the ANN-MLP. The network configuration is x:y:z, where x = n (number of features), y = $|Z_1|$-1 and z = c (number of classes) are the number of neurons in the input, hidden and output layers, respectively. For OPF we used the LibOPF package [15] and for BC we used our own implementation.

Table 1 shows the mean accuracy and execution time in seconds for each classifier for recognizing the cutting density at the vibrating shale shaker for well drilling monitoring purposes.

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Mean Accuracy</th>
<th>Mean execution time</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPF</td>
<td>99.89±0.001</td>
<td>0.0023</td>
</tr>
<tr>
<td>ANN-MLP</td>
<td>99.00±1.200</td>
<td>0.0330</td>
</tr>
<tr>
<td>SVM</td>
<td>99.25±0.612</td>
<td>1.5314</td>
</tr>
<tr>
<td>BC</td>
<td>99.33±0.970</td>
<td>0.0020</td>
</tr>
</tbody>
</table>
One can see that OPF classifier outperformed all remaining classifiers, despite the fact of all of them achieved very good and similar results, which emphasize the robustness of the texture features in our problem. The OPF classifier was about 665 times faster than SVM in this case. Note that the SVM algorithm had a slow performance due to the fact of the optimization procedure implemented in the LibSVM [16]. However, by removing the optimization procedures, this processing time could be decreased. In turn, this could produce lower classification rates.

4.2. Evaluating EOPF

In this section we evaluated the efficiency of EOPF for the same dataset used in the previous section. We repeated each experiment 10 times with randomly generated training and test sets for mean accuracy and mean execution time computation. Table 2 shows the results for OPF and EOPF.

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Mean Accuracy</th>
<th>Mean execution time</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPF</td>
<td>99.89±0.001</td>
<td>0.0023</td>
</tr>
<tr>
<td>EOPF</td>
<td>99.95±0.005</td>
<td>0.0016</td>
</tr>
</tbody>
</table>

The EOPF was slightly better than OPF in terms of accuracy, but it was 1.41 times faster regarding the execution time. However, this difference for algorithm’s execution time may be greater when we have large datasets. In our case, the dataset is composed by 120 samples, equally distributed into 3 classes (low, medium, and high volume concentration). Papa et al. [28] argued that EOPF can be up to 6 times faster than OPF in some applications. We would like to stress that any gain of performance in real-time systems needs to be emphasized, mainly in monitoring ones.

5. Conclusions

Well drilling monitoring is an important and crucial task to detect and prevent problems in the drilling process. Several studies have been developed for drilling inspection, but none of them take care about analyzing the generated cutting at the vibrating shale shakers during the drilling process. The cutting’s shape and size, and even so its volume (density) are important features that allow us to identify possible problems during the drilling operation. Sudden changes in the cutting’s volume, for instance, can indicate a possible collapse of the well borehole walls.

Based on this assumption, Marana et al. [29] have proposed an image-based well drilling monitoring system composed by two modules: an acquisition system and a data analysis step. The images from cuttings at the vibrating shale shaker are acquired and sent to the data analysis module, which was previously trained with calibrated images (labeled by a technician). The system can detect any anomaly with respect to suddenly changes in the cutting volume in real time, which can indicate a possible problem during the drilling process, and further send warning alerts. As far we know we are the first to develop an image-based system for monitoring well drilling operations based on the cutting’s analysis. In this work, we essentially extend the Marana et al. [29] seminal paper by applying a fast and robust modification of OPF called EOPF.

We conducted two rounds of experiments: in the former we applied a collection of four supervised classifiers in this task: OPF, ANN-MLP, SVM and BC. The first one outperformed all the remaining classifiers, both in terms of accuracy and efficiency. The images were mapped into the feature space domain by representing each one of them with a 4-D feature vector, composed by four texture Haralick features: energy, homogeneity, contrast and entropy. These features have been demonstrated to be very robust into discriminate each frame with respect to one of the aforementioned class: E, L or H. In the latter experiment, we compared OPF and EOPF in terms of efficiency and effectiveness. Regarding the recognition rates, the EOPF was slightly better than its traditional version, but it was 1.41 times faster. As far as we know, this is the first time that EOPF is applied for petroleum well drilling monitoring.
6. Acknowledgments

The authors thank the financial support from CNPq, PETROBRÁS and FAPESP Proc. 2009/16206-1.

7. References


