Machine learning and computer vision applications for hydraulic fracture monitoring

by

Jorge Nustes Andrade

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

in

Geophysics

Department of Physics
University of Alberta

Abstract

The spatiotemporal distribution of hydraulic fracturing-induced microseismicity is complicated and depends on various mechanical and diffusional parameters. Hydraulic fracture modeling can aid in understanding fluid-induced microseismicity. Nevertheless, the interaction of several physical processes occurring within and around the fracture adds complexity in developing real-time models for microseismic prediction. This study introduces three methodologies to forecast the microseismic cloud size, which engineers can use to improve the treatment's effectiveness during pumping. We compare a random forest model trained with statistical features derived continuously from the injection monitoring data, a physics-based approach based on diffusivity estimates from the microseismic observations, and a convolutional neural network (CNN) trained in real-time with the engineering curves to predict the future microseismic cloud size. The prediction accuracy of all methods varies depending on the microseismic behavior. We postulate that predictive models could be improved by including more physics into the input data.

Distributed acoustic sensing (DAS) is increasingly used in hydraulic fracturing operations. The low-frequency band of DAS (LFDAS) contains high-resolution information of the far-field strain perturbations that can be used to constrain fracture geometry. Nevertheless, locating the time and depth intersection of a propagating fracture with an offset monitoring well in the same pad (frac-hit) is mainly made using simple cumulative strain maps, which can be subjective and inefficient. We introduce a computer vision workflow based on image matching to automate the detection of frac-hits in LFDAS data. Furthermore, we developed a method to remove the frac-hit strain signal using affine image transformations and warping. The workflow is applied to a real LFDAS dataset from Western Canada, and results exceed the detection performance achieved with cumulative strain maps.

Preface

This dissertation is submitted for the degree of Master of Science in Geophysics at the University of Alberta. The research described herein is original, and neither this nor any substantially similar dissertation was or is being submitted for any other degree or other qualification at any other university.

Acknowledgements

I would like to express my appreciation to my supervisor, Dr. Mirko van der Baan, for his continuous guidance and support during the course of my master's. His questions and comments helped me develop a more critical and systematic way of thinking about a scientific problem. I will always be grateful for the opportunity he gave me and for believing in my capabilities.
I would like to thank the members of my final examination committee, Dr. Mauricio Sacchi, Dr. Claire Currie, and Dr. Mathieu Dumberry for their valuable feedback on my thesis.
I would like to acknowledge the Sponsors of the Microseismic Industry Consortium for their financial support. Special thanks to all my peers from the University of Alberta and the University of Calgary.
Finally, I want to express my deepest gratitude to my mom, my dad, my brother, and my sister for their love and support through all these years. I only came this far because of you.

Contents

List of Tables

List of Figures

List of Abbreviations

LEFM	Linear elastic fracture mechanics
PKN	Perkins-Kern-Nordgren
KGD	Khristianovich-Geertsma-de Klerk
S/N	Signal-to-noise-ratio
DAS	Distributed Acoustic Sensing
LFDAS	Low-frequency Distributed Acoustic Sensing
STA/LTA	Short-term average / Long-term average
SRV	Stimulated reservoir volume
OTDR	Optical-time-domain reflectometry
FO	Fiber-optic
RT	Regression trees
RF	Random forests
CNN	Convolutional neural network
LoG	Laplacian of Gaussian
DoG	Difference of Gaussian

Chapter 1

Introduction

1.1 Background

1.1.1 Increasing energy demand

Better access to energy markets and rapid population growth will increase global energy consumption within the next 30 years. For instance, the Energy Information Administration (EIA), in their 2021 Annual Energy Outlook report, projects that energy consumption will grow by nearly 50% by 2050 (EIA, 2021). Though renewables represent the fastest-growing energy source, dry natural gas will continue to lead energy production in the United States (US) over the next 30 years (Figure 1.1a). Historically, demand for oil and gas has relied on conventional hydrocarbon reservoirs. However, reserves from conventional plays are depleting, shifting the interest to unconventional reservoirs to meet the increasing energy demand (Montgomery and Smith, 2010). For example, the EIA projects that the majority of US dry natural gas production through 2050 will be from shale/tight oil gas reservoirs (Figure 1.1b). Though abundant, unconventional resources pose a challenge for development due to their low permeabilities, low to medium porosities, and high viscosities, requiring secondary recovery techniques such as hydraulic fracturing to optimize production.

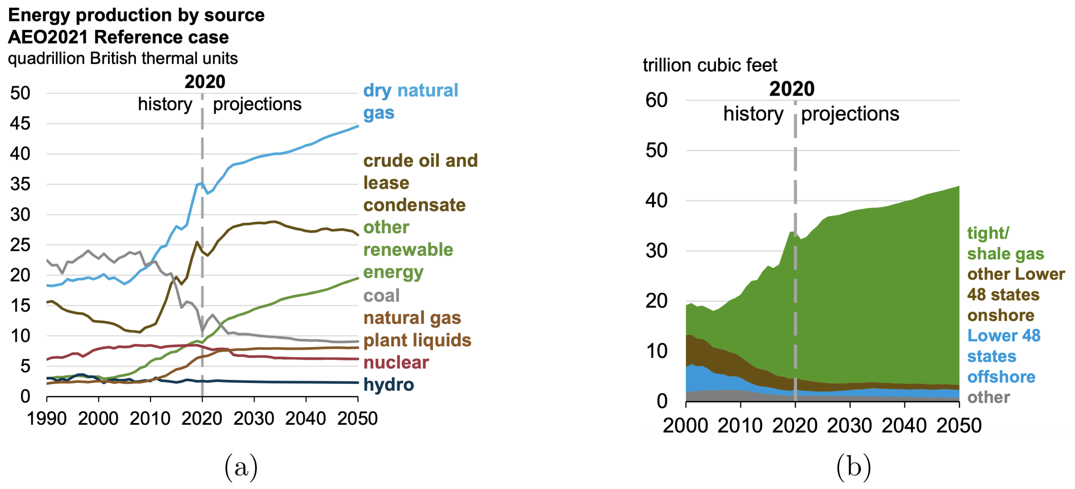


Figure 1.1: Projections of energy production by fuel source in the United States (US) to the year 2050 (a). US dry natural gas production by type (b). The lower 48 states refer to those states with minor oil and gas production. Modified from the EIA February 2021 annual energy outlook report (EIA, 2021).

1.1.2 Hydraulic fracturing

Hydraulic fracturing (*aka* fracking) is a process for injecting highly-pressurized engineered fluids and proppants into low-permeability formations to enhance the connection between the wellbore and the reservoir (Montgomery and Smith, 2010). Proppants are any porous material such as sand or ceramics injected into the fractures to prevent their closing after injection stops. Advancements in horizontal drilling, multi-stage hydraulic fracturing, and slickwater injection (i.e., water with added chemicals like friction reducers used to increase fluid flow) have been described as some of the key disruptive technologies that have enabled the production and development of vast unconventional reserves that were inaccessible only a few years ago (Eaton, 2018; King, 2010). Hydraulic fracturing is done to achieve one or more of the following objectives (Smith and Montgomery, 2015)

- Bypass near-wellbore damage.
- Stimulate the reservoir to increase productivity.
- Alter fluid flow in the formation.

In particular, the main reason behind hydraulic fracturing is to increase the permeability of the medium by connecting the natural fractures in the reservoir, thereby increasing the formation contact with the wellbore (Belyadi et al., 2017). Once the fracture initiates and propagates, the proppant is added to fill the created volume so that the fracture remains open to act as a conductive path for hydrocarbons to flow.

1.1.3 Fracture diagnostics

The classical representation of a hydraulic fracture is a single, symmetrical bi-wing planar crack with the wellbore at the center of both wings (Fisher et al., 2002). However, multiple studies have shown that hydraulic fracturing is more likely to generate complex fracture networks (King, 2010; Maxwell et al., 2009b), requiring advanced mapping technologies to understand fracture growth during the stimulation. Warpinski, 1996 defines fracture diagnostics as any process that provides information about the fracture and its ability to produce. Indeed, understanding the created fracture geometry is vital for proper planning and effective execution of the fracturing job (Cipolla et al., 2011). For example, the fracture height and length determine well spacing and volume of fluid injected, while the fracture density can be used to optimize cluster spacing. There are many diagnostic techniques available (Fisher et al., 2004). Nevertheless, few have had the success of microseismic monitoring in mapping the fracture geometry. This technology works by recording the passive seismic acoustic emissions emitted from rock failure (Maxwell, 2014; van der Baan et al., 2013). Provided an accurate location of the microseismic events, the spatial distribution of the microseismic clouds can be used as a proxy to determine the fracture dimensions, orientation, and complexity.

In recent years ($\approx$ 2009 and onward), other kinds of fracture diagnostics based on fiber-optic technologies like distributed acoustic sensing (DAS) have been increasingly applied to constrain hydraulic fracture geometry (Jin and Roy, 2017). DAS provides continuous measurements of the strain due to acoustic emissions or fracture propagation by interrogating a glass fiber cemented behind casing (Mestayer et al., 2011). At the very-low-frequency band (< 0.05

Hz), DAS can record the time and location when a propagating fracture intersects a fibered offset monitoring well (Ugueto et al., 2019). Moreover, it can be used to monitor fracture opening and closing, stress shadows, and relaxation zones. With low-frequency DAS (LFDAS), engineers can map the depth, azimuth, and fracture speed and use these values to calibrate fracture models and optimize future operations.

1.2 Motivations

We investigate two commonly used hydraulic fracture monitoring tools presently, microseismic monitoring and low-frequency distributed acoustic sensing. Though each technique records different physical processes of the hydraulic fracture, they are very effective in mapping its spatial and temporal evolution. There are two primary motivations in this thesis.

1. Predicting in real-time the microseismic cloud size using machine learning and engineering data recorded during the stimulation.

The spatial extent of the microseismic clouds over time is thought to encapsulate the hydraulic fracture, thus representing a desirable parameter to monitor the effectiveness of the fracturing treatment. The latter is enhanced by large, complex fracture networks that increase the volume of stimulated reservoir rock (Mayerhofer et al., 2010). Historically, hydraulic fracture modeling has been applied to estimate the size of the microseismic cloud based on multiple rock and fluid properties, along with geometrical assumptions of the expected fracture system (see for example (Boroumand and Eaton, 2015) and Barthwal and van der Baan, 2019). Nonetheless, even in the most simplified scenarios, fracture modeling is a complicated process that involves the interplay of at least three physical phenomena (Adachi et al., 2007)

- Mechanical deformation induced by fluid pressure.
- Fluid flow within the fracture.
- Crack-tip propagation.

Besides, other coupled physical processes like fluid leakoff, proppant transport, and interaction with pre-existing fractures add additional complexity for developing fracture models to optimize the treatment design.

In the last years, data-driven methodologies like machine learning have been increasingly applied to many aspects of unconventional resource development such as production forecasting (Cao et al., 2016), optimization of well spacing (Ma et al., 2020), reservoir characterization (Korjani et al., 2016), and real-time drilling prediction (Ben et al., 2019). In simple terms, machine learning refers to a set of techniques that extract information directly from data using well-defined optimization rules (Kong et al., 2019). Therefore, machine learning approaches that can rely on routinely recorded data during the completion time (e.g., the engineering data) for predicting the microseismic cloud size can help overcome the challenges imposed by classical hydraulic fracture modeling.

2. Automate the frac-hit detection in low-frequency DAS using a computer vision workflow based on keypoints matching.

Jin and Roy, 2017 generalized the use of low-frequency DAS (LFDAS) to constrain hydraulic fracture geometry. LFDAS interpretation is made by analyzing the time and intersection location of fracture hits (frac-hits) with an offset monitoring well on the same pad. The frac-hit signal has a specific strain response identified on a waterfall plot of measured depth vs. injection time. Nevertheless, LFDAS is a relatively new technology, and little work has been carried out to optimize fracture hit detection. Typically, frac-hits are identified by either manual inspection or as the maximum cumulative strain change with depth (Ichikawa et al., 2020; Li et al., 2020; Richter et al., 2019; Wu et al., 2020), which can be subjective and inefficient. Moreover, LFDAS data is collected continuously during the well stimulation, causing the strain signal due to fracture closing or opening to persist across multiple stages, thus complicating the frac-hit detection based on cumulative strain maps.

The last decade has seen several advancements in computer vision applications thanks to convolutional neural networks (CNN) and the efficient use of high-performance graphic processing units (GPU) (LeCun et al., 2015). However, before the widespread use of CNN, image matching based on local interest features has proven effective in solving computer vision tasks such as object recognition (Lowe, 1999; Lowe, 2004). Edges, corners, and blobs (i.e., image regions with similar characteristics) are examples of interest features (keypoints) used in image matching. Interestingly, the first hidden layer in a CNN typically represents the presence or absence of edges at particular locations in the image, while a second hidden layer usually detects blobs by spotting distinctive arrangements of edges (LeCun et al., 2015). As a result, keypoints can be conceived as a low-level representation of the image content. Thus, identifying them is one of the first steps to solving any object recognition task. We propose applying a computer vision workflow based on matching interest keypoints between two images to automate frac-hit detection compared to approaches that rely on the cumulative strain.

1.3 Contributions

The contributions of this thesis are as follows:

- Develop a random forest model for microseismic cloud size prediction using local statistical features computed from the engineering curves and past cloud sizes.
- Build upon an existing physical model from Shapiro et al., 1997 to derive in real-time the hydraulic diffusivity from the spatial and temporal distribution of the microseismicity and use it to predict the microseismic cloud size.
- Define and train a 1D CNN that uses the engineering curves to update the network parameters in real-time and forecast future microseismic cloud size values.
- Propose a computer vision workflow to automate the detection of frac-hits in LFDAS by adapting a keypoints matching algorithm developed

by Rublee et al., 2011.

- Subtract the frac-hit strain signal using affine image transformations and warping to aid the matching algorithm to detect more frac-hits and understand the underlying strain response.

1.4 Thesis outline

Chapter 2 of this thesis covers the fundamentals of hydraulic fracturing, such as fracture propagation and fracture modeling. This chapter also provides an extensive description of the theoretical background of microseismic monitoring, distributed acoustic sensing, and low-frequency DAS.

Chapter 3 presents the theory of machine learning with a focus on regression. It provides a detailed description of the random forests algorithm and the most relevant concepts for defining and training a convolutional neural network for time-series analysis. At the end of the chapter, a data augmentation technique known as double-noise injection is presented.

Chapter 4 describes the application of three methods to forecast the size of the microseismic cloud. A random forest model, a 1D CNN, and a physics-based approach are applied to a multi-stage hydraulic fracturing and microseismic dataset. Data pre-processing and a correlation analysis of the engineering curves are also presented.

Chapter 5 describes the theory of image matching for object detection. We start by presenting the most relevant algorithms for feature detection, description, and matching. Also, the fundamentals of image transformations and warping are covered. Finally, we introduce a computer vision workflow for frac-hit detection and subtraction and compare the results on a real LFDAS dataset with detection based on the cumulative strain.

Chapter 6 discusses the conclusions and possible future work.

Chapter 2

Hydraulic fracturing, microseismic monitoring, and distributed acoustic sensing (DAS)

2.1 Introduction

This chapter covers the fundamental physics of fracture propagation and fracture geometry. We describe physics-based 2D fracture models, introduce the material balance principle for fracture growth, and discuss the pore-pressure diffusion concept. We then examine two commonly used fracture diagnostic techniques to monitor hydraulic fracture growth and stimulation efficiency. First, we describe microseismic monitoring, including data acquisition, processing, and interpretation. Next, we cover the theory of distributed acoustic sensing (DAS), describe its applications for hydraulic fracturing monitoring, and define low-frequency DAS for strain measurements.

2.2 Principles of hydraulic fracturing

2.2.1 Stress field around a fracture

Stress distribution around a 2D fracture can be visualized using the classical theory of linear elasticity. Linear elastic fracture mechanics (LEFM) is a continuum mechanics approach that provides a mathematical framework to analyze the stress distribution around a fracture in a linear, isotropic elastic medium. It was initially developed by the work of Griffith, 1921 who observed that failure at lower stresses than the theoretical strength of materials could be explained by the presence of imperfections in the rock. Griffith's theory states that an elastic body with a crack of half-length c subject to an external force σ, will only extend to $c + \delta c$ if the crack allows the total energy of the system U to decrease. The energy-balance equation for a static crack is defined as (Scholz, 2019)

$$U = (-W + U_e) + U_s, \tag{2.1}$$

where W is the work done by external forces σ that could cause a change in the internal strain energy U_e, and U_s is the surface energy of the system required to create new fracture space. The term in parenthesis in equation 2.1 is known as the mechanical energy. To propagate the crack, there must be a reduction in the total energy of the system such that at equilibrium there is a balance between the mechanical and the surface energy. The condition for crack growth at equilibrium is

$$dU/dc = 0. \tag{2.2}$$

If the crack is assumed to be planar, perfectly sharp, and cohesion-free, the energy release rate $\mathcal{G}$, which measures the energy available for crack extension, is defined as (Jaeger et al., 2007)

$$\mathcal{G} = K_i^2/E, \tag{2.3}$$

where K_i is the stress intensity factor and depends on the crack propagation mode i, and E is Young's modulus. Thus, the condition for crack propagation occurs when we introduce K_{ic}, the critical stress intensity factor or *fracture*

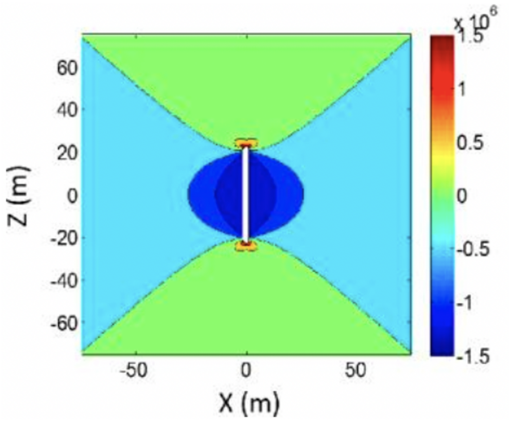


Figure 2.1: Map view of normal stress in Pa around a penny shaped hydraulic fracture. Warm colors represent tensile stress perturbations, while cold colors show compressional stress perturbations. Modified from Barthwal and van der Baan, 2017.

toughness

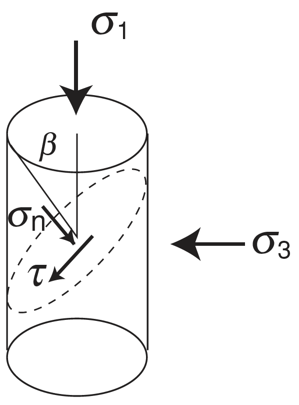
$$\mathcal{G}_c = K_{ic}^2/E, \tag{2.4}$$

and the fracture will propagate as long as $K_i = K_{ic}$, or the energy release rate $\mathcal{G}$ reaches the critical value $\mathcal{G}_c$.

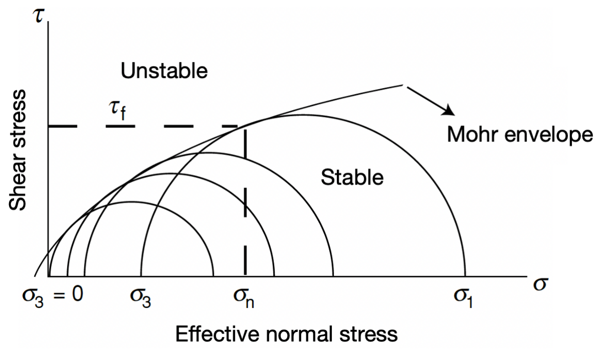
In the case of a mode $i = \text{I}$, tensile fracture (i.e., wall displacements are normal to the crack), the stress field concentrates in the fracture tips (Eaton, 2018), providing a mechanism for self-sustained fracture propagation. For example, Barthwal and van der Baan, 2017 compute the elastic stress perturbations around a static hydraulic fracture modeled as a spheroidal cavity (Figure 2.1). Their results corroborate the concentration of positive stresses on both sides of the crack tip and the occurrence of negative tensile stress that develops alongside the fracture (i.e., compression of the elastic material).

2.2.2 Failure criteria

When describing the rock strength under confined stress conditions, empirical or semi-empirical criteria are usually used. A common approach to represent the shear stress τ and effective normal stress σ_n acting on a fault plane under



(a)



(b)

Figure 2.2: A fault plane under the effect of applied effective principal stresses σ_1 and σ_3 (a). The Mohr failure envelope derived from triaxial strength tests at different effective confining pressures (b). Modified from Zoback, 2007.

confining stresses (Figure 2.2a) is the use of Mohr circles or Mohr envelopes (Zoback, 2007). Using geometrical relationships, the shear and normal stress components in any orientation in the plane can be defined as a function of the principal stresses σ_1, σ_3 and the angle β between the fault normal and σ_1 as (Twiss and Moores, 2006)

$$\begin{aligned} \tau &= \left(\frac{\sigma_1 - \sigma_3}{2}\right) \sin(2\beta) \\ \sigma_n &= \left(\frac{\sigma_1 + \sigma_3}{2}\right) + \left(\frac{\sigma_1 - \sigma_3}{2}\right) \cos(2\beta)\,. \end{aligned} \tag{2.5}$$

The Mohr envelope is derived empirically from the tangent of Mohr circles calculated from several experimental triaxial confining pressure tests ($\sigma_1 > \sigma_2 = \sigma_3$), thus dividing the stress space into an stable and unstable region where failure is expected to occur (Figure 2.2b).

Mohr-Coulomb criterion

For most rocks it is possible to express the Mohr envelope in terms of a linearized Mohr-Coulomb failure envelope defined as

$$\tau = S_o + \sigma_n \mu_i, \tag{2.6}$$

where S_o is the cohesion, and μ_i is the slope of the failure line, also known as the coefficient of internal friction (typical values for μ_i are in the range of $0.6 \leq \mu_i \leq 1$ (Byerlee, 1978)) (Figure 2.3). It is worth noting that cohesion is not a physical parameter that can be measured, but it is related to the unconfined compressive stress C_o (Zoback, 2007)

$$C_o = 2S_o \left[\left(\mu_i^2 + 1 \right)^{1/2} + \mu_i \right]. \tag{2.7}$$

In the case of critically stressed pre-existing faults, where the cohesive stress is very small compared to the shear and normal stresses, the S_o term in equation 2.6 is neglected (Ito and Zoback, 2000). Therefore, the Coulomb failure criterion for fluid-saturated rocks can be represented as (Fossen, 2016)

$$\tau = \mu_i \left(\sigma_n - p_p \right), \tag{2.8}$$

where p_p is the pore-pressure. Frictional sliding will occur on a plane where the ratio of shear to normal effective stress is larger than the coefficient of internal friction ($\tau / \left(\sigma_n - p_p \right) \geq \mu_i$)Ito and Zoback, 2000. This condition is satisfied at two planes oriented at acute angles to the axis of maximum compressive stress σ_1 (Healy et al., 2006; Scholz, 2019). From equation 2.8, it can be seen that the presence of fluids in the pore spaces play an important role on the strength of rocks. Increasing the pore pressure will tend to increase the ratio between the shear and normal stresses, thus reducing the rock strength and increasing the probability of inducing failure.

Griffith Criterion

The Mohr-Coulomb criterion is only valid for compressive stresses. Nevertheless, hydraulic fracturing mostly induces the formation of tensile fractures (Eaton, 2018; Maxwell et al., 2008). To understand the failure of rock in the tensile domain, a parabolic representation of the tensile and shear stresses acting on a fault plane can be expressed using the Griffith criterion (Griffith, 1924; Jaeger et al., 2007)

$$\tau = \sqrt{4T_o \left(\sigma + T_o \right)}, \tag{2.9}$$

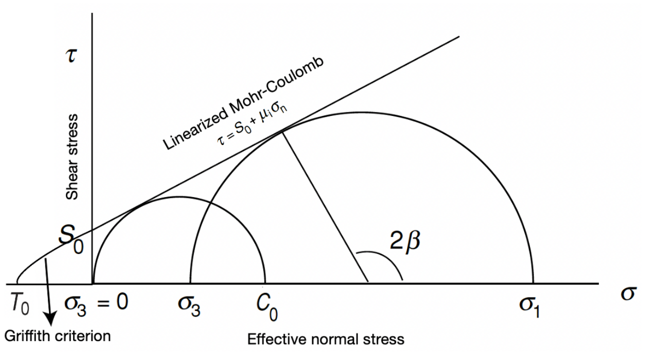


Figure 2.3: Linear Mohr-Coulomb failure criterion and the parabolic Griffith Criterion.

where T_o is the tensile strength (Figure 2.3). Pure tensile fractures will occur when $\sigma_n < 0$ & $\tau = 0$, while pure shear will be observed when $\sigma_n = 0$ & $\tau > 0$. During hydraulic fracturing the presence of natural fractures can trigger hybrid events with both shear and tensile components ($\sigma_n < 0$ & $\tau > 0$) (Vavryčuk, 2001). Fischer and Guest, 2011 suggest that the occurrence of hybrid events is highly dependent on the fracture plane orientation, with the biggest likelihood of hybrid failure on faults whose strike orientation falls within $\approx 22.5°$ of σ_1.

2.3 Hydraulic fracturing modeling

Most hydraulic fracturing models predict the growth of symmetrical bi-wing hydraulic fractures that propagate perpendicular to the direction of minimum compressive stress in the formation (Belyadi et al., 2017). As a result, an effective fracturing treatment aims to optimize the surface area of the fracture in contact with the reservoir (Smith and Montgomery, 2015). The latter can be approximated as the fracture width w times the tip-to-tip fracture length (i.e., $L = 2x_f$ with x_f being the fracture half-length on a bi-wing hydraulic fracture). Thus, fracture models help understand hydraulic fracture geometry, which is critical for optimizing the treatment effectiveness.

2.3.1 Basic 2D fracture models

One of the first models used to predict the width of hydraulic fractures was developed by Perkins and Kern, 1961 by assuming that the fracture length is much greater than its height (Figure 2.4a). Indeed, hydraulic fractures tend to be contained in height growth as the fracture encounters varying rock layers (Fisher and Warpinski, 2012). Under this condition, and assuming an infinite extent (i.e., plane strain), the maximum width w of a hydraulic fracture with a fixed height h_f is defined as (Sneddon and Elliot, 1946)

$$w = \frac{2P_{net}h_f}{E'}, \tag{2.10}$$

where P_{net} is the net pressure and E' is the plane strain modulus defined as

$$E' = \frac{E}{1-\nu^2}, \tag{2.11}$$

with E being the Young's modulus and ν the Poisson's ratio of the medium. Equation 2.10 indicates that the fracture width is linearly proportional to the net pressure. In other words, the net pressure within the fracture compresses the formation and increases fracture width. Nordgren, 1972 introduced the effects of fluid loss to develop the *Perkins-Kern-Nordgren* (PKN) 2D fracture model. The leakoff velocity $u_L(\mathbf{x})$ at a position $\mathbf{x}$ on the fracture surface is given by (Carter, 1957)

$$u_L(\mathbf{x}) = \frac{C_L}{\sqrt{t - t_{exp}}}, \tag{2.12}$$

where C_L is the leakoff coefficient, t is the current time, and t_{exp} is the time at which the fracture opened at $\mathbf{x}$.

Conversely, Geertsma and de Klerk, 1969 derived the *Khristianovich–Geertsma–de Klerk* (KGD) 2D fracture propagation model under the assumption that the fracture height is much greater than its length (Figure 2.4b). The KGD model assumes a constant pressure across the majority of the fracture body, except near the fracture tips. For this model, the fracture width w is described as

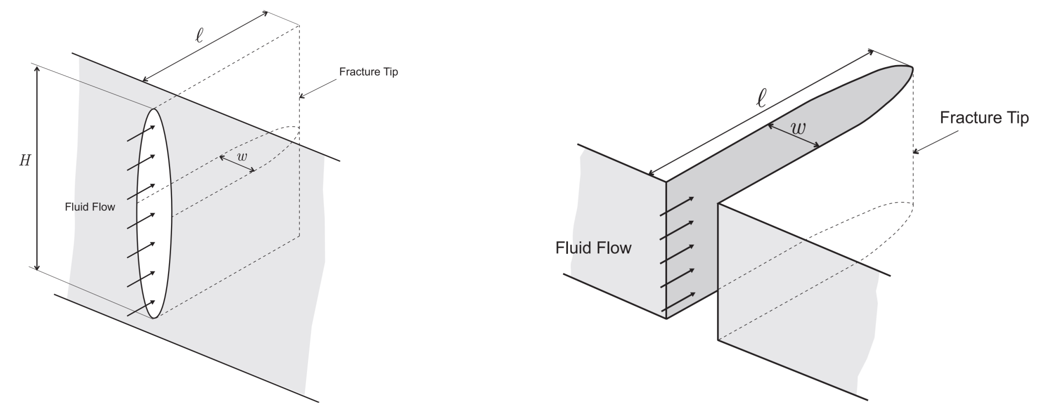


Figure 2.4: Schematic drawing of the PKN fracture geometry (a). Schematic of the KGD fracture geometry (b). H is the height, l is the length, w width. Modified from Adachi et al., 2007.

(Mack and Warpinski, 2000)

$$w = \frac{4LP_{net}}{E'}, \tag{2.13}$$

where L is the fracture length. The third 2D fracture propagation model is the *radial or penny-shaped model* (Abe et al., 1976). In this case, the reservoir is modeled as an infinitely extended, homogeneous, and isotropic medium with a point injection source.

2.3.2 Material balance

The material balance equation (Economides and Nolte, 2000) implies that the volume of fluid injected to the formation V_i must be equal to the sum of fluid lost thought leakoff V_L and the hydraulic fracture volume V_f

$$V_i = V_L + V_f. \tag{2.14}$$

For a PKN fracture geometry, equation 2.14 takes the form (Economides and Nolte, 2000)

$$Q_i t = 4LhC_L\sqrt{2t} + 2hwL, \tag{2.15}$$

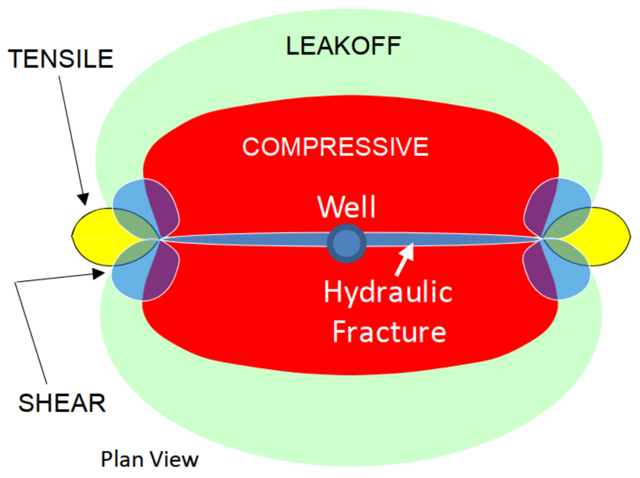


Figure 2.5: Schematic representation of the regions of influence for fluid leakoff, fracture deformation, and crack tip propagation. Modified from Cipolla et al., 2011.

where Q_i is the injection rate, L is the half-length, h is the height of the fracture, w its width, and C_L is the fluid-loss coefficient. Rearranging equation 2.15 gives the fracture half-length as a function of time

$$L(t) = \frac{Q_i t}{\left(4hC_L\sqrt{2t} + 2hw\right)}. \tag{2.16}$$

The first term in the denominator of equation 2.16 describes the fluid loss to the formation and shows a $1/\sqrt{t}$ behavior (see equation 2.12). The second term, $2hw$, represents the contribution of the fracture volume and depends on the geometry of the fracture's vertical cross-section. Figure 2.5 shows a schematic diagram with the regions of influence for fluid leakoff and stress deformation due to the opening of a propagating hydraulic fracture (Cipolla et al., 2011). Tensile regions appear at the fracture tips along with lobes of shear stress. At the same time, positive compression zones develop alongside the fracture walls due to the formation resistance to tensile opening (see Figure 2.1 for comparison). Also, a leakoff-influenced area develops on each side of the fracture. Depending on the permeability of the medium, it can extend to far distances from the main fracture body.

2.3.3 Pore-pressure diffusion

During hydraulic fracturing treatments, the injection of fluids causes an increase of the pore pressure and a decrease in the effective normal stress (Holland, 2013). If the pore pressure change is enough to make the ratio of shear to normal stress maximum, failure could be induced at pre-existing favorable oriented cracks (Zoback and Harjes, 1997). In other words, to initiate the creation of a tensile hydraulic fracture, the pore pressure induced by the injection of fluid must exceed the minimum principal confining stress (Economides and Nolte, 2000). However, when the pore-pressure is smaller than the minimum principal stress, seismicity is mainly controlled by a process of stress relaxation and diffusion at the injection source (Shapiro, 2015). In the low-frequency limit of Biot's equations (Biot, 1962), the pore pressure changes due to a point source of injection in an infinite, heterogeneous, anisotropic medium can be described by solving the diffusion equation (Rothert and Shapiro, 2003; Shapiro et al., 2002) as

$$\frac{\partial p}{\partial t} = \frac{\partial}{\partial x_i}\left(D_{ij}\frac{\partial}{\partial x_j}p\right), \tag{2.17}$$

where D_{ij} are the components of the tensor of hydraulic diffusivity, $x_j = (1, 2, 3)$ are the components of the radius vector from the injection point, and p is the pore pressure. Shapiro et al., 1997 introduced the microseismic triggering front to understand the spatial and temporal relationship between the microseismic events induced by pore pressure diffusion. The latter defines the maximum distance r from the injection source at which events induced by a pore-pressure perturbation occur at a given time t. The triggering front equation on a medium with scalar diffusivity D is defined as (Shapiro et al., 1997)

$$r = \sqrt{4\pi D t}. \tag{2.18}$$

Equation 2.18 matches the upper bound of the microseismic cloud in a plot of the radial distance r from the injection source to the hypocenters of events versus time t of occurrence ($r-t$ plot). Assuming that the medium's diffusivity is unknown, equation 2.18 can be used to calculate D from the spatio-temporal distribution of fluid-induced microseismicity in an $r-t$ plot. This relationship

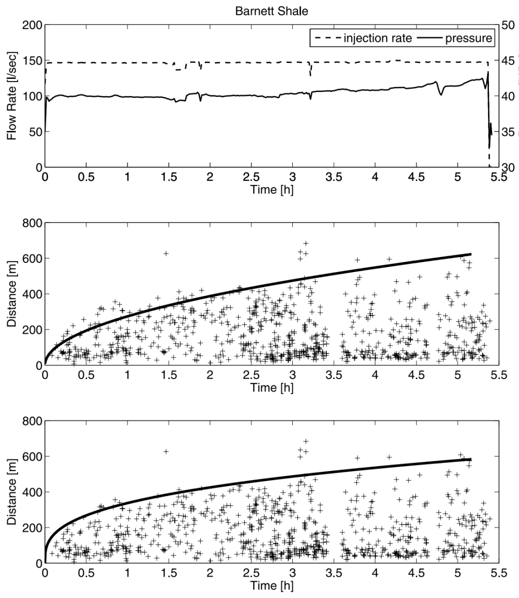


Figure 2.6: Bottom hole pressure and injection rate (top). Distance of the microseismic events from the injection source versus time of occurrence $r - t$ plots (middle and bottom). The black curve on the middle plot shows the scaling of the triggering from as a square root of time $\sqrt{t}$. The black curve on the bottom plot shows the scaling as a cubic root of time $\sqrt[3]{t}$. Modified from Shapiro and Dinske, 2009b.

will be explored in chapter 4 and used in a prediction model to forecast the size of the microseismic cloud. It is worth mentioning that equation 2.18 can be derived from 2.16 by ignoring fracture propagation (that is, width $w = 0$).

Figure 2.6 presents an example of the distribution of the microseismic events over time induced by a hydraulic fracturing treatment in the Barnett shale gas reservoir (Fisher et al., 2002). The middle figure shows that during early times the triggering front behaves close to a linear function with respect to time. The latter is a direct consequence of fracture volume dominance at the beginning of the injection (second term in the denominator of equation 2.16). However, in the long term, diffusion fluid-loss processes dominate the fracture growth and the fracture length becomes proportional to $\sqrt{t}$.

The previous analysis assumes that the effects of either pore-pressure diffusion or elastic stress changes dominate fracture growth, thus ignoring the coupling between both. Nevertheless, multiple studies have analyzed poroelastic coupling's contribution to injection-induced seismicity (Rozhko, 2010; Segall and Lu, 2015). For a non-linear diffusional process (e.g., coupling between pore pressure and stress), the triggering front of microseismic events on an $r - t$ plot fits better as a $\sqrt[3]{t}$ (Figure 2.6, bottom) (Shapiro and Dinske, 2009a). Such behavior corresponds to a pore-pressure diffusion where the medium's permeability is strongly enhanced by the injection of fluids (Shapiro, 2015).

2.4 Fracture diagnostics

Understanding the spatial and temporal evolution of hydraulic fractures and their interaction with the reservoir is required to enhance stimulation effectiveness and improve treatment design. Fracture diagnostics provide relevant information about the fracture geometry and stimulation process used to optimize field development and well economics (Cipolla and Wright, 2000; Warpinski, 1996). Some techniques include pressure diagnostics such as discrete fracture injection tests (DFITs) (Economides and Nolte, 2000), tiltmeters (Warpinski et al., 1998), radioactive tracers (Warpinski, 1996), fiber-optic based methods like distributed acoustic sensing (DAS) (Ugueto et al., 2014), and microseismic monitoring (Maxwell, 2014). In this section, we focus on the last two.

2.4.1 Microseismic monitoring

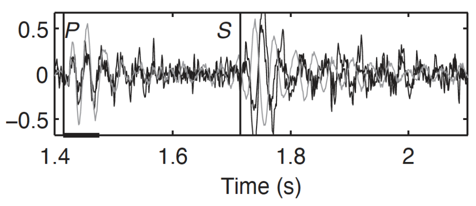
During hydraulic fracturing, the injection of fluids and proppant at high pressures causes stress and pore pressure changes, thereby inducing microseisms at pre-existing planes of weakness (Le Calvez et al., 2005). Microseismicity refers to tiny discrete rock failure events that often have negative moment magnitudes (Maxwell et al., 2010; van der Baan et al., 2013) and are adequately represented by a double-couple source mechanism related to shear slippage (Rutledge and Phillips, 2003). Microseismic monitoring involves the detection, location, and analysis of the microseismic events using the principles of earthquake seismology (Albright and Pearson, 1982). Today, it is one of the most widely used tools for mapping the fracture growth and geometry (Fisher et al., 2004; Maxwell et al., 2009a; Warpinski et al., 2004), providing details of the fracture azimuth, length, height, and complexity (Cipolla et al., 2011).

Microseismic data acquisition

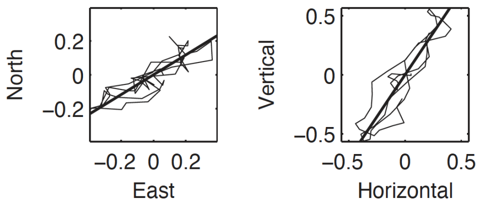
Microseismic acquisition deals with the continuous passive seismic monitoring of microseismic events using various sensor configurations (Maxwell, 2014). The P and S-waves radiated from the microseismic sources are detected using nearby arrays of receivers or regional seismic networks (Eaton, 2018). Thus, a successful microseismic acquisition requires recording high-amplitude seismic signals with a sufficiently large signal-to-noise ratio (S/N). For hydraulic fracturing monitoring, receivers such as 3-component (3C) geophones or accelerometers are placed at the surface (Duncan and Eisner, 2010) or in downhole vertical or horizontal monitoring wells (Maxwell et al., 2010). The latter offers the advantage of proximity to the source, thus reducing anelastic attenuation and leading to high S/N. However, the location and orientation of downhole receivers are less constrained than that of surface arrays. With the advent of multi-stage hydraulic fracturing treatments on multiple pads, a combination of downhole horizontal, vertical, and surface arrays can be processed together to characterize the source better and avoid potential detection biases (Maxwell, 2014).

A detailed velocity model is required to reduce the uncertainty in the locations of the microseismic sources (Eisner et al., 2011). The accuracy of the model depends on how constraint are the receiver positions, a good knowledge of the velocity structure in the reservoir, and precise phase-arrival pickings (Warpinski et al., 2005). The model must contain both compressional and shear-wave velocities (Le Calvez et al., 2016). Typically, the initial velocity model is built from sonic and density well-log data. However, during hydraulic fracturing, stress and pore-pressure perturbations induce local changes that affect mostly the horizontal velocity structure (Warpinski et al., 2005). Thus, the arrival times of a known source like perforation shots, string shots, or ball drops are used to calibrate the initial velocity model following an iterative methodology.

A similar process can be used to identify the orientation of downhole 3C geo-



(a)



(b)

Figure 2.7: A microseismic event recorded at one station using a 3C geophone. Vertical lines show the P and S-wave arrivals, and the bold line at the bottom is the time window used for the hodogram analysis (a). Horizontal (left) and vertical (right) hodograms with the P-wave polarizations shown by bold lines (b). Modified from De Meersman et al., 2006.

phones. The receivers are rotated into a geographic coordinate system using the P-wave arrivals from a calibration source with a known location (Drew et al., 2008). For that, a *hodogram* is built for each sensor to cross-plot two orthogonal components of ground motion during a time window following the phase arrival (Figure 2.7). From the computed hodograms, it is possible to determine the P-wave polarization's azimuth ϕ using an approach like the covariance matrix method (CMM) (Jurvekys, 1988). Once the P-wave polarization angle is known, the receiver orientation κ with respect to geographic coordinates is calculated as (Akram, 2020)

$$\kappa = \phi - \phi_s, \tag{2.19}$$

where ϕ_s is the known source azimuth of the calibration shot. For each receiver level, a linear transformation can be applied on the 3C waveforms (h_1, h_2, z) to rotate them into the geographic coordinate reference east-north-vertical (e, n, v) as

$$\begin{pmatrix} e \\ n \\ v \end{pmatrix} = \begin{pmatrix} \cos\kappa & \sin\kappa & 0 \\ -\sin\kappa & \cos\kappa & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} h_1 \\ h_2 \\ z \end{pmatrix}. \tag{2.20}$$

Microseismic data processing

Microseismic data processing involves calculating characteristics of the microseismic source from the continuous waveform observations. Such characteristics could be but are not limited to event locations, magnitudes, and source mechanisms (Maxwell, 2014). The microseismic processing workflow usually starts with filtering any unwanted noise from the recorded waveforms to enhance S/N (Vera Rodriguez et al., 2011). Common filtering methods used include frequency filtering (Han and van der Baan, 2015), polarization filtering (Pinnegar, 2006), and time-frequency thresholding (Mousavi et al., 2016). After preconditioning the data, a detection algorithm is applied to trigger signals with a coherent arrival relative to background noise. Most detection algorithms compute attributes from the recorded signal and determine the P and S-wave arrivals using a threshold-based criterion.

Event detection and arrival-time picking

Among many of the methods available for microseismic event detection (Akram and Eaton, 2016), the *short-term average / long-term average* (STA/LTA) (Allen, 1978) from earthquake seismology is one of the most commonly used ones. In STA/LTA, the ratio of continuously averaged energy is calculated for consecutive time windows and used as a criterion for picking (Vaezi and van der Baan, 2015). The STA window is sensitive to sudden increases in the amplitude while the LTA window measures the local background noise. Therefore, the ratio of the STA with the LTA is a measure of the local S/N. If the ratio is higher than a predefined threshold value, an arrival time is declared (Earle and Shearer, 1994). The STA/LTA ratio R calculated at the ith data sample of a time series $\mathbf{u}$ is defined as (Eaton, 2018)

$$R_i(\mathbf{u}) = \frac{\mathrm{STA}_i(\mathbf{u})}{\mathrm{LTA}_i(\mathbf{u})}, \tag{2.21}$$

where

$$\begin{aligned}\mathrm{STA}_i(\mathbf{u}) &= \frac{1}{\mathrm{N_{STA}}} \sum_{j=i}^{i+\mathrm{N_{STA}}-1} y_j(\mathbf{u}),\\ \mathrm{LTA}_i(\mathbf{u}) &= \frac{1}{\mathrm{N_{LTA}}} \sum_{j=i-\mathrm{N_{LTA}}+1}^{i} y_j(\mathbf{u}).\end{aligned} \tag{2.22}$$

In equation 2.22 $\mathrm{N_{STA}}$ and $\mathrm{N_{LTA}}$ are the number of samples in the short and long windows respectively, and $y_j(\mathbf{u})$ is the characteristic function (CF), which is chosen in such a way that it enhances the signal changes (e.g., the energy $\mathbf{x}_i^2$) (Vaezi and van der Baan, 2015). If $R_i(\mathbf{u})$ in equation 2.21 exceeds a predetermined threshold value Λ, a detection is declared (Figure 2.8). When using STA/LTA, the window lengths and threshold value should be carefully selected to get optimal results. It is recommended that the length of the STA window should be at least two to three times the dominant period of the signal, while the length of the LTA window should be five to ten times the length of the STA (Akram and Eaton, 2016). Conversely, the detection threshold should be large enough to avoid detecting false positives while ensuring that small-magnitude events are triggered.

Other methods used for event detection and arrival-time picking include those based on waveform cross-correlation (Arrowsmith and Eisner, 2006) like the matched filtering algorithm (Caffagni et al., 2016) or the subspace detector (Harris, 2006), Akaike information criterion (AIC) (Oye and Roth, 2003), neural networks (Maity et al., 2014), digital image segmentation (Mousa et al., 2011), and high-order statistics such as the Kurtosis (Li et al., 2014).

Hypocenter estimation

Determining the hypocenter locations of the microseismic events constitutes one of the essential elements of the microseismic catalog and the basis for interpreting the geometry of hydraulic fractures. Akram, 2020 classifies the absolute location methods (i.e., locations relative to a fixed coordinate system) into two main categories, traveltime-based methods, and waveform-based methods. The first approach involves arrival-time inversion combined with event back-azimuth information derived from hodogram analysis (see figure 2.7). Given a set of N arrival-time picks and M inferred back-azimuth directions, an event hypocenter can be determined by minimizing the objective function E (Eaton, 2018)

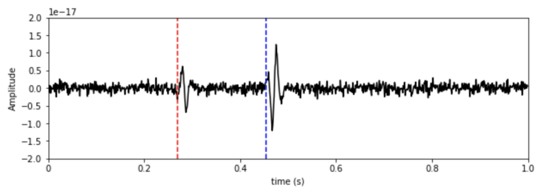
$$E = \sum_{i=1}^{N} \left(t_i(\mathbf{m}) - \hat{t}_i\right)^2 + w \sum_{i=1}^{N} \left(\phi_i(\mathbf{m}) - \hat{\phi}_i\right)^2, \tag{2.23}$$

where t_I are the modeled arrival-times, $\hat{t}_i$ are the observed arrival-time picks, ϕ_i are the modeled back-azimuth orientations, $\hat{\phi}_i$ the derived back-azimuths obtained from hodogram analysis, and w is a weight factor. Equation 2.23 can be solved through least-squares to find the model parameters $\mathbf{m} = \{x, y, z, \tau\}$ containing the geographic source coordinates and origin time, τ. For a single vertical monitoring well on a horizontally layered isotropic medium, the arrival-time inversion can be analyzed using cylindrical coordinates (Maxwell, 2014), thus reducing the model parameters to $\mathbf{m} = \{r, \theta, z\}$. In this case, the 2D hypocenter is mostly constrained by the difference of the S and P-wave time-arrivals $(t_s - t_p)$. The hypocenter locations can then be transformed to a Cartesian reference frame for further interpretation. Lastly, the back-azimuth average over all receivers with rotated waveforms (see equation 2.20) can be computed as an estimate of the azimuthal direction from the receiver to the source.

An alternative way to solve equation 2.23 is based on the Geiger's method (Geiger, 1912), which uses an initial estimate of the model parameters $\mathbf{m}_o$ to obtain a solution to the generalized inverse matrix (Akram, 2020; Oye and Roth, 2003)

$$\mathbf{G}\Delta\mathbf{m} = \Delta\mathbf{r}, \tag{2.24}$$

where $\mathbf{G}$ is a matrix that contains traveltime derivatives with respect to the model parameters $\mathbf{m}$, $\Delta\mathbf{m} = (\Delta x, \Delta y, \Delta z, \Delta\tau)$ is the model difference vector, and $\Delta\mathbf{r}$ are the arrival-times residuals. Equation 2.24 can be solved using an iterative least-squares inversion approach, such that the model parameters at


(a)

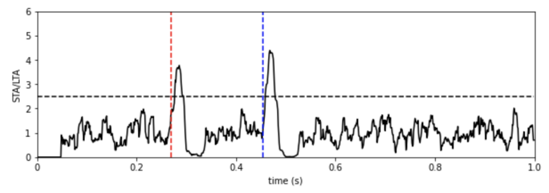

(b)

Figure 2.8: Synthetic example of event detection using the STA/LTA method. Input waveform (a). STA/LTA of the input signal (b). Vertical dashed lines show the P (red) and S-wave (blue) arrival time-picks obtained using a detection threshold ($\Lambda = 2.5$) represented by the dashed horizontal line.

iteration $i + 1$ are updated as

$$\mathbf{m}_{i+1} = \mathbf{m}_i + \Delta\mathbf{m}. \tag{2.25}$$

Other techniques for microseismic hypocenter locations include the grid search method (Rodi, 2006), waveform stacking (Grigoli et al., 2016), and reverse time migration (Nakata and Beroza, 2016).

Magnitude estimation

The microseismic source can be characterized using source parameters derived from fitting a source model to the amplitude spectrum of the recorded waveforms (Maxwell, 2014). Typically, for microseismic events, a Brune source model (Brune, 1970) is fitted to the amplitude spectrum of a time-window containing the P or S-wave arrival. The Brune model at a particular frequency f is defined as

$$\Omega_f = \frac{\Omega_0 \mathrm{e}^{-\pi f t/Q}}{\left[1 + \left(\frac{f}{f_c}\right)^2\right]}, \tag{2.26}$$

where Ω_0 is the low-frequency level of the amplitude spectrum, Q is the quality factor, and f_c is the corner frequency. Once the model gets fitted to the spectrum, the seismic moment M_0 is computed as (Stork et al., 2014)

$$M_0 = \frac{4\pi\rho V^3 r\Omega_0}{R}, \tag{2.27}$$

where ρ is the rock density, V is the P or S-wave velocity, r is the source-receiver distance, and R is a P or S-wave radiation pattern correction term. A more physical meaning of M_0 is found from slip characteristics induced by the failure of intact rock. For instance, M_0 can also be defined as

$$M_0 = \mu A d, \tag{2.28}$$

where μ is the shear modulus, A is the area of slip, and d is the displacement of slip Cipolla et al., 2011. For microseismic monitoring, the preferred source strength estimate is the moment magnitude M_W scale. The latter can be computed from the seismic moment M_0 using the definition from Kanamori, 1977

$$M_W = \frac{2}{3}\log(M_0) - 6. \tag{2.29}$$

Finally, the frequency of occurrence for microseismicity with different magnitudes is usually analyzed using earthquake seismology theory. The Gutenberg-Richter relationship can describe the magnitude distribution of microseismic events (Gutenberg and Richter, 1944). The latter states that the number of cumulative events N with a magnitude M or greater during a certain period of time follow the relationship

$$\log(N) = a - bM, \tag{2.30}$$

where a is the relative activity rate and b is the relative size distribution of earthquakes the slope on a frequency-magnitude plot.

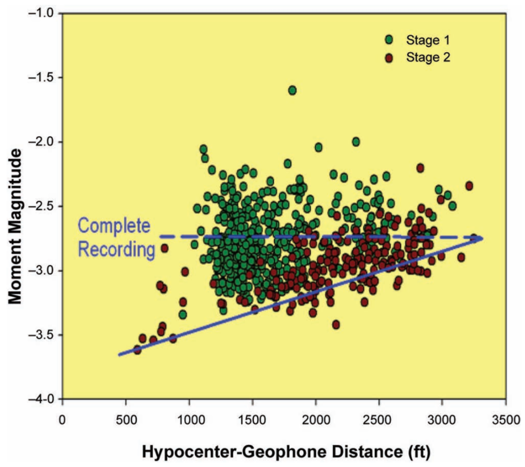


Figure 2.9: Magnitude-distance plot of microseismic events recorded during a two-stage hydraulic fracturing stimulation monitored with a single vertical array of 3C geophones. Events above the dashed line (moment magnitude of -2.8) represent a complete recording of unbiased events. From Maxwell et al., 2009b

Microseismic interpretation

The ultimate goal of microseismic monitoring is to characterize fracture growth and geometry while providing details about the reservoir's geomechanical response during hydraulic fracturing treatments (Cipolla et al., 2010). Maxwell, 2014 lists the typical hydraulic fracture geometric characteristics inferred from microseismic event locations, which include:

- fracture orientation
- fracture height
- fracture length
- fracture complexity
- stimulated reservoir volume (SRV) (Mayerhofer et al., 2010)
- fault reactivation (Warpinski, 2009a)

In the case of simple planar fractures, fracture dimensions from microseismic spatial clusters can be measured by fitting a rectangular box, with the longest axis defining fracture length, and the remaining axes representing fracture width and height. For a more rigorous analysis, statistical approaches based on clustering can also be used to constraint fracture dimensions (McKean et al., 2019).

Data preconditioning

Although the interpretation of microseismic images may seem straightforward, it is important to be aware of the inherent uncertainties associated with the acquisition and processing of microseismic data. For example, Eisner et al., 2009 analyzed the impact in the uncertainty of hypocenter locations due to errors arising from the receiver geometry, arrival-time picks, and velocity model.

For a single-downhole monitoring array, large magnitude events are more likely to be successfully detected compared to small magnitude events (Warpinski, 2009b). This introduces an observation well bias resulting in an apparent greater event density close to the sensors and a minimum detection limit that increases with distance from the source (Cipolla et al., 2011; Maxwell, 2014). To correct for the effects introduced by the detection bias, a *magnitude-distance plot* can be used to filter events below a magnitude threshold (Figure 2.9). Similarly, surface microseismic recordings, which are not affected by the well detection bias, can be filtered using a S/N cut-off.

Once events have been filtered to remove any monitoring bias effects, they are clustered into distinct groups for interpretation. Microseismic-event clustering is usually performed by stage number based on time of occurrence. In this case, events are assigned to a particular stage if they occur within a time window spanning the stage injection. However, hydraulic fracturing fluid-induced microseismic events can persist after the end of a stage (Shapiro, 2015), and complex fracture behavior with strong spatio-temporal dependencies between stages have been reported in multiple case studies (Maghsoudi et al., 2016; Rafiq et al., 2016). Therefore, a clustering of events that accounts for both spatial and temporal relationships among stages could improve the interpretation workflows relying on the stage-by-stage separation of microseismic events (e.g., the calculation of the radial distance of the events from the assigned stage perforation). In chapter 4, we describe a method to cluster events by stage by minimizing a spatio-temporal constraint.

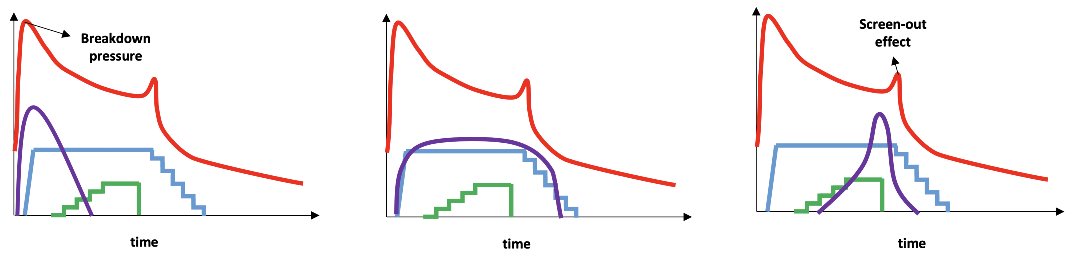
Integration with engineering data

The temporal distribution of microseismicity recorded during a hydraulic fracturing treatment can be compared to the engineering curves (i.e., injection rate, proppant concentration, pressure) to understand fracture growth associated with specific injection characteristics (Cipolla et al., 2011). Maxwell, 2014 provides an interpretation of the temporal distribution of microseismicity relative to the engineering curves. Assuming a constant injection, a high rate of microseismic occurrence at the beginning of the treatment indicates fracture initiation achieved at the breakdown pressure (Figure 2.10a). Conversely, consistent microseismic rate through the injection time represents fracture growth (Figure 2.10b), while an increase in microseismicity towards the end of the stage indicates fracture blockage (e.g., proppant screen-out) or growth into a fracture zone (Figure 2.10c).

Frequency-magnitude distributions and advanced interpretation

The distribution of event magnitudes can provide insights into the mechanisms that caused the microseismicity (Schorlemmer et al., 2005). Assuming a Gutenberg-Richter relationship (equation 2.30) for the recorded microseismicity, the b-value calculated from the slope of a frequency-magnitude plot can be used to separate hydraulic fracture events from those associated with fault activation. Typically, a $b \approx 1$ implies fault activation, while hydraulic fracturing induced microseisms show a $b \approx 2$ (Maxwell et al., 2009a). The latter implies that the falloff of events with increasing magnitude is faster for hydraulic fracturing induced microseismicity than for events associated with fault systems.

Other interpretation insights from microseismic monitoring include fractal di-



(a) (b) (c)

Figure 2.10: Schematic drawings of three different physical models of the microseismicity (purple) relative to the engineering curves (i.e., injection rate in blue, surface pressure in red, and proppant concentration in green). Microseismicity at the beginning of a stage is associated with fracture initiation (a). A constant microseismic histogram at a stage is caused by fracture growth (b), and a spike in events at the end of a stage relates to fracture termination (c). After Maxwell, 2014.

mension D analysis (Grob and van der Baan, 2011), moment tensor inversion (Baig et al., 2010), and S/P amplitude ratio (Rutledge and Phillips, 2003).

2.4.2 Distributed acoustic sensing (DAS)

Distributed Acoustic Sensing (DAS) is a fiber-optic (FO) based technology that has been recently applied in a variety of unconventional development settings, including real-time hydraulic fracturing in-well operations monitoring (Molenaar et al., 2011), down-hole geophysical surveillance (Mestayer et al., 2011), microseismic monitoring (Webster et al., 2013a), vertical seismic profiling (VSP) (Mateeva et al., 2014), and geomechanical analysis (Jin and Roy, 2017). During DAS, an opto-electronic system *aka* interrogator unit (IU) sends successive short pulses of highly coherent laser light down a single-mode silica optical fiber. The IU uses optical-time-domain reflectometry (OTDR) principles to measure the distortions in Rayleigh back-scattered light generated from small heterogeneities within the fiber's glass core. The back-scattered signal is used to derive the strain or strain rate along spatially-localized regions of the fiber. Essentially, instead of using conventional arrays of discrete sensors, DAS repurposes the FO cables as multi-channel seismic arrays capable of measuring axial acoustic disturbances spanning thousands of meters.

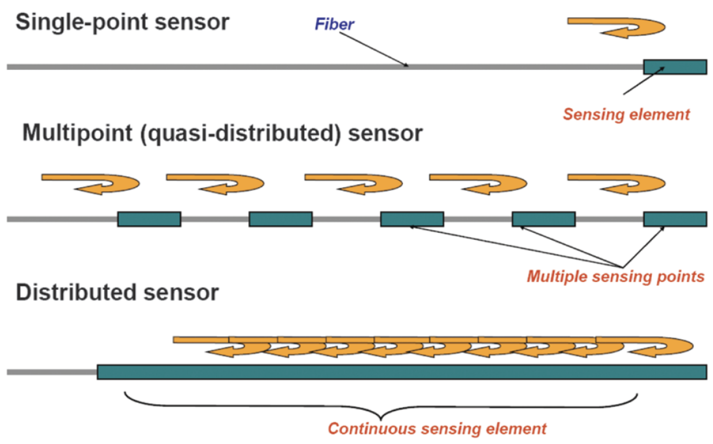


Figure 2.11: Classification of optical fiber sensors according to their topology (Krohn et al., 2014).

Distributed fiber-optic systems

Optical fiber sensors can be broadly classified as point sensors, discrete sensors, and distributed sensors. In a point sensor, the sensing unit is located at the end of the fiber, whereas in a discrete sensor, fiber is modified at two or more discrete points that serve as sensors (Hartog, 2017). In a distributed FO sensor, every portion along the length of the fiber is sensitive to one or more external physical quantities (e.g., temperature, strain), enabling continuous measurements for up to tens of kilometers (Figure 2.11).

DAS uses the scattered light along the entire length of the sensing fiber to measure the strain due to incident seismic waves or hydraulic stresses in pores and fractures. The system relies on a modified version of OTDR in which a series of highly-coherent laser pulses are launched into an optical fiber connected to an IU that detects the backscattered signals (Barnoski and Jensen, 1976; Parker et al., 2014). The scattered light results from inhomogeneities in the fiber's glass core associated with three distinct types of scattering mechanisms, namely Rayleigh, Brillouin, and Raman scattering (Krohn et al., 2014).

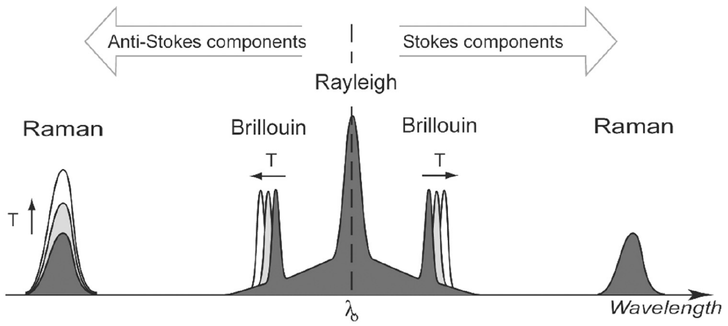


Figure 2.12: Different components of the backscattered light from silica glass fiber. Modified from Molenaar and Cox, 2013.

Rayleigh scattering

When a light beam travels on a glass medium, most of the energy propagates forward, thus experiencing a loss from the medium's attenuation (Healey, 1984). However, a small portion of the light ($\approx 0.1\% - 1.0\%$) is scattered back within the fiber's acceptance solid angle and recorded by the IU (Hartog, 2017). The three most important scattering mechanisms used for distributed sensing are Rayleigh, Brillouin, and Raman scattering. Rayleigh backscatter occurs when photons collide with microscopic-scale inhomogeneities frozen in the glass medium. Rayleigh scattering is an elastic process with no energy transfer between the incident and the returning light, hence preserving the original wavelength of the input pulse. On the contrary, Raman and Brillouin scattering are caused by thermally excited molecular vibrations involving a loss of energy linked to an inelastic scattering behavior. Raman scattering is highly sensitive to temperature fluctuations and is therefore used on distributed temperature sensing (DTS) systems (Bakku, 2015). Figure 2.12 shows the full scattering spectrum from a silica glass fiber. Note that while Rayleigh scattering preserves the input wavelength λ_o, Raman and Brillouin scattered photons shift to lower or higher frequencies (i.e., Stokes or anti-Stokes scattering) depending on whether there is a loss or gain in energy respectively.

Optical phase change and fiber strain

The backscattered response from multiple pulses on an undisturbed FO cable has a constant phase measured in radians as (Grattan and Meggitt, 2000)

$$\Phi = \frac{4\pi n L}{\lambda}, \tag{2.31}$$

where n is the refractive index of the medium, L is the distance traveled, and λ is the laser pulse wavelength. However, when the fiber is distorted, there is a change in phase between the scattered signals coming from two points in the cable. The physical distance between the two points is referred as the *gauge length* L_G. One should not confuse L_G with channel spacing. While the latter is the physical separation between consecutive sample locations on the fiber ($\approx 0.5 - 1m$), L_G is the effective spatial resolution of the DAS system, which has to be large enough ($\geq 7m$) to ensure optimal S/N (Dou et al., 2017) (Figure 2.13). The phase-lag $\Delta\Phi$ between the backscattered signals from the two points is assumed to be linearly represented by the expansion (Grattan and Meggitt, 2000)

$$\Delta\Phi = \frac{4\pi n L_G}{\lambda}\left[\frac{\Delta x}{x} + \frac{\Delta n}{n} + \frac{\Delta\lambda}{\lambda}\right]. \tag{2.32}$$

Therefore, changes in the optical phase $\Delta\Phi$ are caused by modifications in the fiber length $\frac{\Delta x}{x}$ due to axial strain ϵ_{xx}, the refractive index $\frac{\Delta n}{n}$, and the optical wavelength $\frac{\Delta\lambda}{\lambda}$. To isolate the strain effects in DAS, the wavelength dispersion may be ignored ($\frac{\Delta\lambda}{\lambda} = 0$) by filtering out all other frequencies but that of the incident pulse $1/\lambda_o$ (Lindsey et al., 2020). Besides, the changes in the index of refraction can be reduced to a scalar multiplicative photo-elastic factor ζ dependent on known material properties of a silica glass fiber. Thus, the axial strain ϵ_{xx} averaged over a gauge length is

$$\epsilon_{xx} = \frac{\lambda}{4\pi n L_G \zeta}\Delta\Phi. \tag{2.33}$$

Assuming typical values for the pulse's incident wavelength $\lambda = 1550$nm, refractive index of the fiber $n = 1.445$, gauge length $n = 10$ m, and photo-elastic

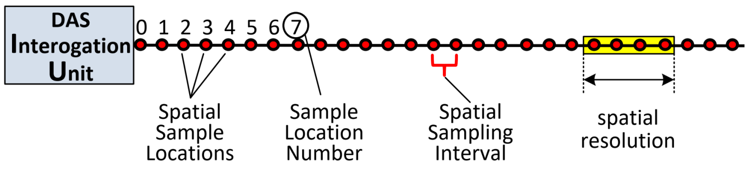


Figure 2.13: Graphical representation of the sampling parameters of a DAS system. The spatial sampling locations are represented by red circles, while the spatial resolution defined by the gauge length, is shown in yellow.

factor $\zeta = 0.735$, equation 2.33 can be written as (Lindsey et al., 2020)

$$\epsilon_{xx} = 11.6 \times 10^{-9} \Delta\Phi \text{ (rad)}, \tag{2.34}$$

which only depends on the phase change between two consecutive backscattered signals separated by the gauge length.

In-well DAS measurements

Shell conducted the first down-hole application of DAS for hydraulic fracturing monitoring in an unconventional reservoir in 2009 in the Groundbirch Montney area (Molenaar et al., 2011; Ugueto et al., 2019). Since then, multiple studies have shown the advantage of using DAS as a cost-effective and non-intrusive diagnostic tool for hydraulic fracture stimulation (Molenaar and Cox, 2013; Webster et al., 2013b). Usually, an FO cable is installed on a well to monitor the treatment's effectiveness in that same well. The information obtained from in-well DAS recordings can be used to optimize fluid, and proppant placement Ugueto et al., 2014; Ugueto et al., 2018, diagnose the effectiveness of the perforation design (e.g., bridge plug setting and ball placement) (Boone et al., 2015; Holley and Kalia, 2015), and monitor well integrity (Raab et al., 2019). However, while an FO-equipped well is not being treated, it can be used as an observational well to monitor the poro-elastic buildup within the reservoir and the propagation of hydraulic fractures (Ugueto et al., 2019; Webster et al., 2013b). This is the principles behind low-frequency DAS strain measurements.

Low-frequency DAS

Typical sampling rates for DAS data are in the range of 10 to 100 kHz, depending on the length of the fiber (Mateeva et al., 2014). However, in terms of sensitivity, DAS performance is comparable to that of geophones around 10 Hz but far more sensitive for frequencies under 1 Hz (Richter et al., 2019). Jin and Roy, 2017 generalized the use of low-frequency DAS (LFDAS), which can be treated as a linear-strain measure, to monitor fracture propagation during hydraulic fracturing. To analyze the low-frequency band of DAS, raw data is usually down-sampled after a low-pass anti-aliasing filter and corrected for any DC drift (Jin and Roy, 2017). At the low-frequency band (< 0.05 Hz), DAS can be sensitive to real-time acoustic disturbances on the order of pico-strain and temperature variations of less than one milli-kelvin (Li et al., 2020). Hence, LFDAS provides an exact location and timing where fractures intersect a neighboring fibered well, thereby allowing measurements of the fracture's azimuth, fracture widths, and propagation speeds (Wu et al., 2020). Figure 2.14 shows the hypothetical strain rate response of a hydraulic fracture approaching an FO monitoring well whose cable is mechanically coupled to the formation. At the dimensionless times $T1$ and $T2$, the fracture strains the rock ahead of the fracture tip producing an extending strain band that narrows and intensifies as it gets closer to the fiber. When the fracture physically arrives at the offset well ($T3$), it creates a localized extending zone surrounded by compressing areas (stress shadow) that decrease in magnitude with distance from the main fracture body. At $T4$, once the injection rate stops, the DAS strain pattern reverses as the fracture closes due to the compressive stresses in the formation.

Figure 2.15 (top) shows the LFDAS strain rate signal for one stage of a nearby well stimulated via open-hole well completion. The vertical black-dotted lines represent the start and end times of the pumping period. The fracture hits are interpreted as extension areas (red) in the fiber due to fracture opening during the injection time. Normally, the precursor of the fracture hit is recorded as a chevron or heart-shaped extension signal. Once the fracture hits the fiber, the areas adjacent to the propped region of the fracture experience compression (blue), and give rise to stress shadows. Usually, the portion of fiber that detects the fracture hit is comparable to the length of the stage at the injection well, while the stress shadow can be recorded at much more channels. After the fracture hit, multiple extension and compression strain pulses appear, though it is not clear yet what these pulses represent, Ugueto et al., 2019 suggests they could correspond to the generation of successive hydraulic fractures created within the main fracture domain. When the injection stops, the polarity of the LFDAS signal reverses causing a compressive zone in the fiber at the fracture hit zone, and extension signals at the surrounding rock due to formation relaxation. The closing signal can last for hours and be seen at later stages. In fact, the compressive horizontal band around 270 m in figure 2.15 corresponds to the closing signal remnant from a previous stage.

The lower panel of figure 2.15 shows the normalized surface treating pressure, injection rate, and proppant concentration, on the same time scale of the LFDAS measurements. There is a clear temporal correlation between the engineering curves from the injection well and the strain rate behavior recorded at the offset fibered well. As a result, comparing the treatment curves with the LFDAS data can be used to understand the timing and extent of communication between different wells in the same pad.

DAS microseismic

A well equipped with a DAS system can be used to detect microseismic activity during a hydraulic fracturing treatment done in a neighboring well in the same pad. Using the broadband and wide-aperture response achieved with DAS, it is possible to record the P-waves and S-waves generated from a shear failure event with much higher quality than classical geophones (Karrenbach et al., 2017; Webster et al., 2013a). The location of the microseismic event along the fiber can be estimated from the apex of the time moveout curves, while the distance from the source to the fiber is inferred from the P-to-S travel time difference (Karrenbach et al., 2017). DAS is analogous to an array of single-component sensors capable of measuring travel-times but no angle information. Hence, the location in 3D of microseismic events using DAS is only possible using multiple FO-equipped wells or a single deviated well (Cole et al., 2017). For instance, if

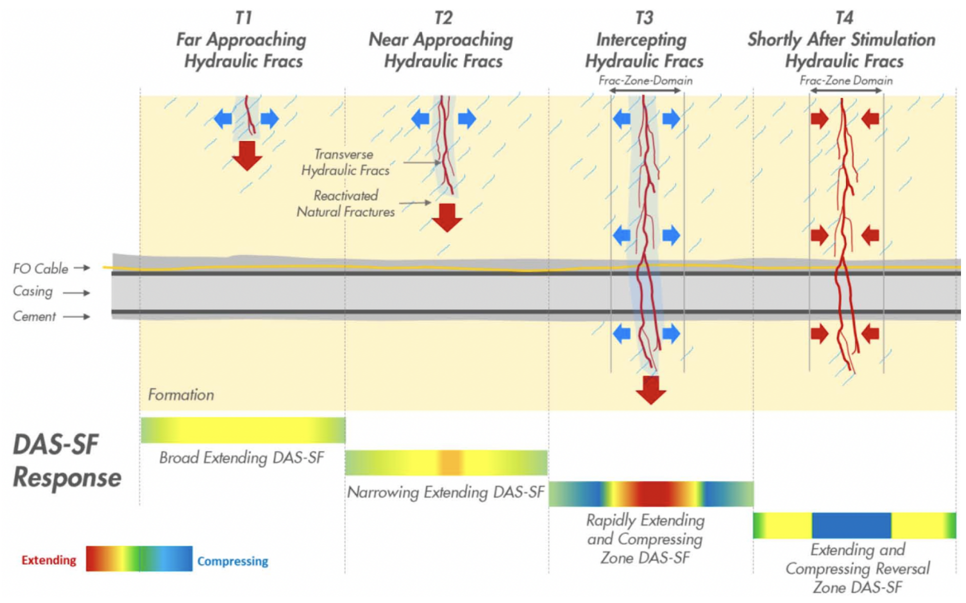

T1
Far Approaching Hydraulic Fracs
T2
Near Approaching Hydraulic Fracs
T3
Intercepting Hydraulic Fracs
Frac-Zone-Domain
T4
Shortly After Stimulation Hydraulic Fracs
Frac-Zone Domain
Transverse Hydraulic Fracs
Reactivated Natural Fractures
FO Cable
Casing
Cement
Formation
DAS-SF Response
Broad Extending DAS-SF
Narrowing Extending DAS-SF
Rapidly Extending and Compressing Zone DAS-SF
Extending and Compressing Reversal Zone DAS-SF
Extending
Compressing

Figure 2.14: Hypothetical evolution over dimensionless time (T) of the LFDAS response from an approaching hydraulic fracture recorded at an offset wellbore with fiber deployed. From Ugueto et al., 2019.

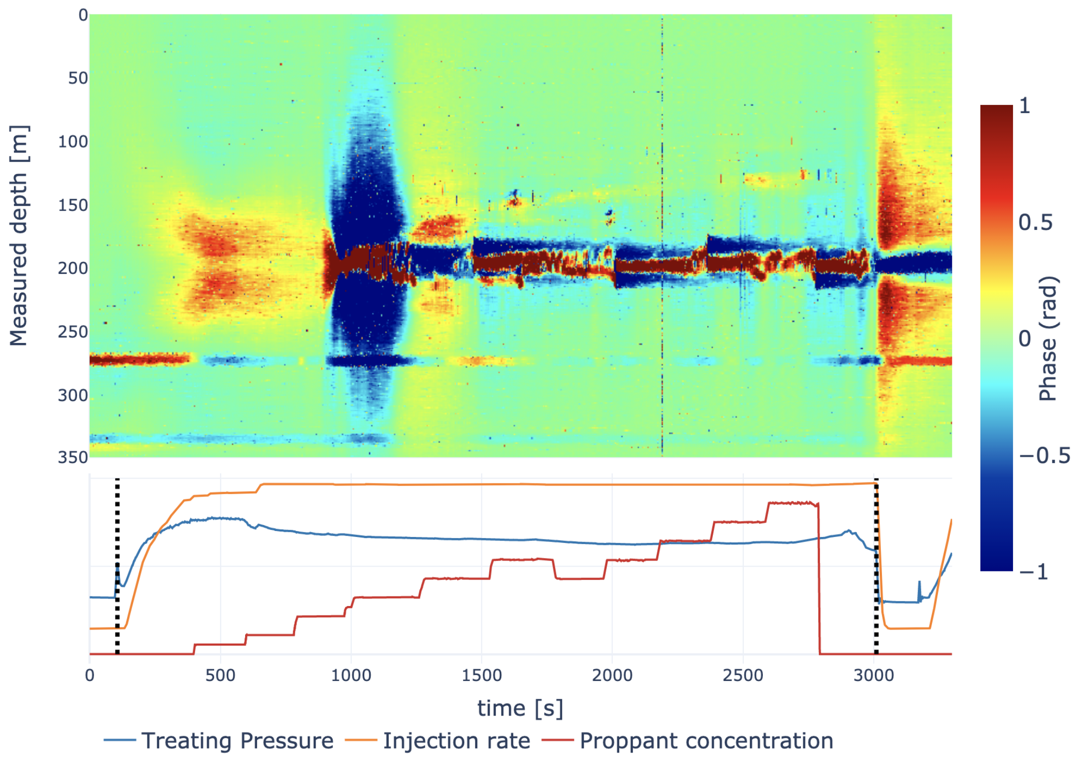


Figure 2.15: LFDAS strain signal of a single stage from a nearby offset well stimulated via a ball activated open-hole packer sleeve (top). Engineering curves for the stimulated stage (bottom).

an event is recorded along the horizontal, heel, or vertical portions of the cable, it can be accurately mapped in space. Although fewer events are detected on DAS compared to 3C geophones (Webster et al., 2016), DAS can locate events with uncertainties ranging from $1 - 2m$ thanks to its large aperture and dense spatial sampling.

Chapter 3

Machine learning, random forests, and convolutional neural networks

3.1 Introduction

This chapter focuses on machine learning theory and the specific algorithms used in this work, presently random forests (RF) and convolutional neural networks (CNN). The first part provides an overview of machine learning, regression, and the random forests algorithm. The concepts of regression trees, bagging, and feature importance are covered. Next, the fundamentals of convolutional neural networks focusing on 1D CNNs for time-series forecasting are presented. Finally, a data augmentation technique known as double-noise injection is introduced. The latter allows increasing the amount of training data available to the machine learning model so-as-to improve the CNN performance on unseen data.

3.2 Machine learning

Machine learning (ML) can be understood as an automated process that extracts patterns from data and uses experience to improve performance (Mohri et al., 2012). It consists of designing efficient prediction algorithms that can learn from data by combining fundamental concepts from probability theory, statistical analysis, and optimization rules. Most ML algorithms can be grouped into two categories depending on whether target labels are provided (supervised learning) or not (unsupervised learning) (Figure 3.1). Supervised learning builds a predictive model by mapping an input data vector to a corresponding label or target value. Furthermore, supervised ML is divided into classification (categorical) and regression algorithms (quantitative), based on the nature of the target output. On the other hand, unsupervised learning relies on unlabeled data to make predictions on unseen points. In this learning setting, the idea is to discover inherent patterns or structures in the data. Clustering and dimensionality reduction are two subcategories of unsupervised learning problems.

The final goal of ML is *generalization*, which refers to the ability of the model to perform well on previously unseen data (Bishop, 2006). Typically, in machine learning, we have access to a training set used to generate the model and a test set used to evaluate its performance. The test set is drawn from the same probability distribution as the training set, but it is never used in the training process, thereby providing a measure of the generalization error (Goodfellow et al., 2016). A properly designed and trained model should avoid memorizing the training labels (also known as overfitting the training set) to improve generalization on validation or test sets. In this thesis, we focus on supervised learning with numerical labels for regression tasks.

3.2.1 Regression

Linear regression is perhaps one of the best examples of a supervised machine learning algorithm. The purpose of regression is to predict as closely as possible the value of one or more continuous target variables t given the value of an input

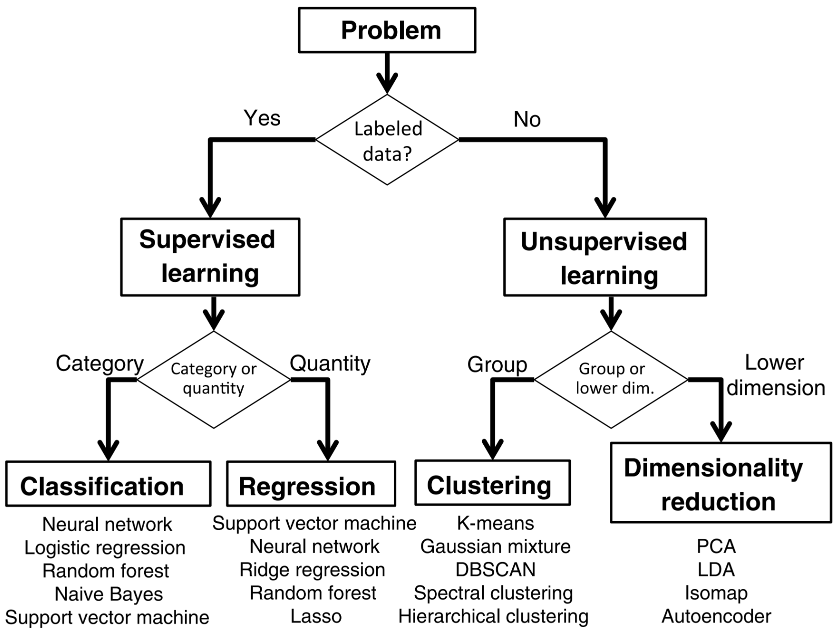


Figure 3.1: Types of machine learning (ML) algorithms (supervised and unsupervised). Some common examples of each category are listed at the bottom. From Kong et al., 2019.

vector $\mathbf{x}$ (Bishop, 2006). That is, given a training data set with N observations $\mathbf{x}_n$ and corresponding target values $\mathbf{t}_n$, the goal is to predict the value of t for a new value of $\mathbf{x}$. A linear model function $y(\mathbf{x}, \mathbf{w})$ of the input vector $\mathbf{x}$ can be written as

$$y(\mathbf{x}, \mathbf{w}) = w_0 + \sum_{j=1}^{M-1} w_j \phi_j(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}), \tag{3.1}$$

where w_j are the weighs or model parameters, $\phi_j(\mathbf{x})$ are known as basis functions, w_0 is the bias parameter, $\mathbf{w}^T = (w_0, \ldots, w_{M-1})^T$, and $\phi(\mathbf{x}) = (\phi_0, \ldots, \phi_{M-1})^T$. The performance of the prediction depends on the magnitude of the difference between the real and predicted value, usually defined by an error function. Some commonly used error metrics are the mean squared error (MSE) and the mean absolute error (MAE); which are defined as

$$\begin{aligned}\mathrm{MSE}\left(y, \hat{y}\right) &= \frac{1}{n}\sum_{i=0}^{n-1}\left(y_i - \hat{y}_i\right)^2, \\ \mathrm{MAE}\left(y, \hat{y}\right) &= \frac{1}{n}\sum_{i=0}^{n-1}\left|y_i - \hat{y}_i\right|,\end{aligned} \tag{3.2}$$

where y_i is the target value and $\hat{y}_i$ is the prediction of the model at the i'th sample. Thus, a good regression model is quantified by the combination of weights and bias that produce the minimum training error while delivering the smallest error on the test set (Goodfellow et al., 2016). Usually, linear combinations of fixed, nonlinear basis functions can be used in equation 3.1 to model nonlinearities between the input and target variables. However, defining the basis functions before the model has seen the dataset can lead to poor prediction performance (Bishop, 2006). An alternative to fixed $\phi_j\left(\mathbf{x}\right)$ is to use non-parametric regression approaches like random forests (Breiman, 2001) and convolutional neural networks (LeCun and Bengio, 1998) that allow non-linearities to be learned from the data without being explicitly modeled.

3.2.2 Cross-validation

k-fold cross-validation

When working with small datasets, dividing the data into fixed training and test sets (e.g., using 80% of the examples for training and the remaining 20% for testing) can result in statistical uncertainty if the test size is too small (Goodfellow et al., 2016). An alternative approach for dealing with limited data availability is training and evaluating the prediction model on randomly chosen subsets or splits of the original training dataset. A common technique used is the k-fold cross-validation procedure, where the dataset is divided into k non-overlapping subsets called folds, and the model is fitted k times. On each fold, the training data uses $k-1$ of the folds and evaluates on the remaining fold (i.e., the validation set). The validation error may then be estimated as the average validation error across the k trials.

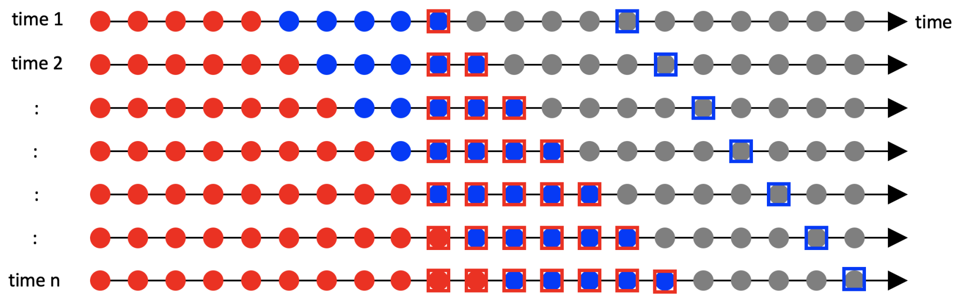


Figure 3.2: Schematic drawing showing the time-series cross-validation based on the rolling forecasting strategy. Training set (red circles), training labels (red squares), test set (blue circles), and test labels (blue squares). Each circle represents 1 time-series data sample. The gray circles show future time, which is not accessible during the training or testing. Note how the size of the training set increases as more data become available.

Time-series cross-validation

A second kind of cross-validation updates the model parameters as new data become available. Time-series cross-validation uses a rolling forecasting strategy where the input data at which the forecast is based on rolls forward in time (Hyndman and Athanasopoulos, 2018). The corresponding training set consists only of observations that occurred prior to the observation that forms the test set. Figure 3.2 shows a schematic diagram of time-series cross-validation for multi-step forecasts (blue squares). The training data increases by one time sample with time, while the test data size remains the same, but rolls forward.

3.3 Random forests for regression

Random forests (RF) are an ensemble learning method for classification and regression tasks developed by Breiman, 2001. In recent years, they have become very popular due to their applicability to a wide range of prediction problems, their ability to handle large numbers of features without requiring many observations, and the need for a few parameters to be tuned. Besides, RF provide an estimate of the variable importance without extra computation cost, which can help assess the impact of different features on the target value.

3.3.1 Regression trees

Regression trees (RT) are the basic units making up the RF algorithm. A regression tree is built by the recursive partition of a sample known as the root node, into subgroups called internal nodes, and down to the terminal nodes (Grömping, 2009). In this algorithm, each partition is done based on a specified splitting test and the final prediction, once reaching a stopping criterion (e.g., when each terminal node has less than a minimum number of observations), is the average value of the desired terminal node. RT use an approach known as *recursive binary splitting*, in which the predictor space at the top of the tree is successively partitioned into two new branches at each node. The goal is to find the terminal nodes R_j that minimize the sum of the squared residuals (RSS) given by (James et al., 2013)

$$\mathrm{RSS} = \sum_{j=1}^{J} \sum_{i \in R_j} \left(y_i - \hat{y}_{R_j} \right)^2, \tag{3.3}$$

where y_i is the ith target value of the variable to be predicted and $\hat{y}_{R_j}$ is the mean response for the training observations within the jth terminal node. RT can easily overfit the training set if the resulting tree is too complex. Therefore, a strategy known as *pruning* is commonly applied to reduce its complexity. The idea is to grow a large tree and then, prune it back until a *subtree* emerges that leads to the lowest test error.

CART-split criterion

In RT, the splitting of each node is done following the classification and regression tress (CART) criterion initially proposed by Breiman et al., 1984. At each node of the tree, the best cut is selected by optimizing the CART-split criterion, which measures the overall node impurity as the variance between the data available at the current node and the two subsets of resulting data after the split. More specifically, the criterion for a possible split $\Delta\sigma(s, j)$ is written as (Rouet-Leduc et al., 2017)

$$\Delta\sigma(s,j) = \sigma(S_j) - \frac{N_{j,L}}{N_j}\sigma(S_{j,L}) - \frac{N_{j,R}}{N_j}\sigma(S_{j,R}), \tag{3.4}$$

where $\sigma(S_j)$ is the variance at the current node with N_j data points, $\sigma(S_{j,L})$ is the variance at the left subset made up of $N_{j,L}$ points , and $\sigma(S_{j,R})$ is the variance of the right subset with $N_{j,R}$ data points. Hence, at each node the split that maximizes the reduction of equation 3.4 is selected. For a detailed description of the CART algorithm, see Scornet et al., 2015.

Bagging

If a dataset is randomly divided into two equal batches and a RT with the same parameters is fitted on each batch, the results will be quite different in each case. The latter is a direct consequence of the high *variance* associated with the RT algorithm (Breiman et al., 1984). To reduce the high variance effect, Breiman, 1996 introduced the concept of *bagging* for generating multiple versions of a predictor and aggregating them to obtain a single low-variance statistical model. Bagging is based upon the *bootstrap* tool, in which random samples with replacement (i.e., same observation can occur more than once on the samples) are repeatedly selected from the original training set to fit and aggregate various predictive models. For B different bootstrapped training sets, the final prediction after bagging for a single RT ($\hat{f}_{\text{bag}}$) is the average from all models defined as

$$\hat{f}_{\text{bag}} = \frac{1}{B}\sum_{b=1}^{B}\hat{f}^{*b}(x), \tag{3.5}$$

where $\hat{f}^{*b}(x)$ is the bth trained set. In other words, averaging a set of RT reduces the overall variance of the final model.

Besides reducing variance and improving accuracy, bagging can also measure the generalization error of the combined ensemble of trees. In each bootstrap training set, about two-thirds of the instances are used on the fitting process, while the remaining one-third of observations are left out. The remaining data are referred to as *out-of-bag* (OOB) observations (Grömping, 2009) and provide an estimate of the test error for the bagged model. The accuracy of an RT prediction can be estimated from the OOB data as

$$\text{OOB-MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{y_{i\text{OOB}}}\right)^2 , \tag{3.6}$$

where $y_i - \hat{y_{i\text{OOB}}}$ denotes the average prediction for the ith observation using only the trees in which that observation was OOB (i.e., around $B/3$ of all bagged regressors).

Feature importance

The RT algorithm can be used to predict feature importance using the OOB observations. The importance of a variable x_m is measured by quantifying how much the weighted impurity (i.e., the variance of the data at node j in equation 3.4) decreases for all nodes where x_m is used, averaged over all B trees in the forest (Louppe et al., 2013). A large decrease indicates a relevant variable. This RT property can be beneficial for identifying features with a low variance that can contribute to overfitting during training.

3.3.2 The Random Forests algorithm

Random forests consist of many regression trees whose nodes are split into random subsets of candidate variables following the CART criterion (section 3.3.1). The overall prediction is the average of predictions from each tree in the forest. However, unlike regular bagged regression trees, each tree node in a RF considers only a random sample of m features from the full set of p input features available. More specifically, each node only has access to around $m \approx \sqrt{p}$ features, thereby reducing the correlation between trees and making the overall average more reliable. Figure 3.3 shows a diagram with the general structure of a random forest made up of unpruned regression trees. The RF algorithm for regression is as follows (Liaw and Wiener, 2002):

1. Draw n bootstrap samples from the original data.

2. For each n grow a regression tree by maximizing the CART-split criterion using only m randomly sampled features at each node.

3. At each bootstrap iteration, make a prediction using the OOB data.

4. Predict new data by averaging the predictions from all trees.

5. Estimate the test error as the average error from all OOB predictions.

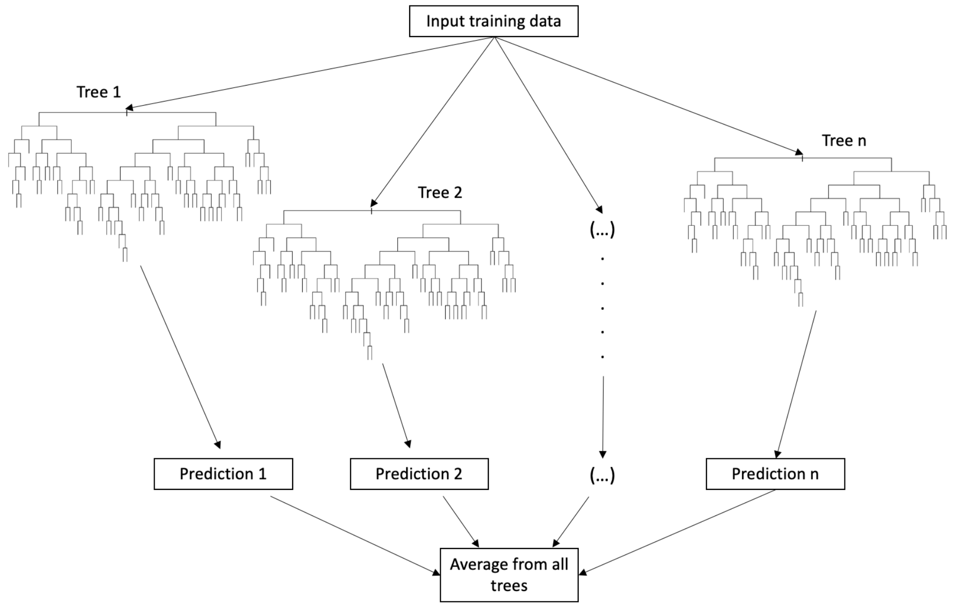


Figure 3.3: Schematic of a random forest with n unpruned regression trees. The final prediction is the average from all trees in the forest.

3.4 Convolutional Neural Networks

Convolutional Neural Networks (CNN) are a specialized kind of artificial neural network for processing data with a known grid-like topology such as time-series in 1D or image pixels in 2D (Goodfellow et al., 2016). Successful applications of CNN go back to the mid-nineties (LeCun and Bengio, 1998; LeCun et al., 1998) where they were applied to solve text recognition tasks. However, it was not until 2012 that a deep CNN (i.e., a CNN with several hidden layers and millions of free parameters) beat the state-of-the-art in the ImageNet classification challenge (Krizhevsky et al., 2012), paving the path for most of the recent

CNN architectures like VGGNet (Simonyan and Zisserman, 2015), UNet Ronneberger et al., 2015, and ResNet (He et al., 2016). CNN are widely used for time-series forecasting applications thanks to their fast training times, simple architectures, and no dependency constraints on previous steps (Wang et al., 2019). Kiranyaz et al., 2019 describe four advantages of applying CNN when dealing with 1D signals:

- Rather than matrix operations, forward and back-propagation require simple array operations.
- Shallow architectures (e.g., small number of hidden layers and few neurons) can learn challenging tasks.
- With a few hidden layers (≤ 2) and a few neurons (< 50) they can be trained using a standard CPU implementation.
- Due to their low computational requirements, are well suited for real-time, low-cost applications.

3.4.1 CNN basic architecture

Regardless of an specific CNN architecture, all CNN models consist of the same basic components including convolutional layers, activation functions, pooling layers, and fully-connected layers. In the next sections I describe each of these components.

Convolutional layer

A convolutional layer consists of kernels or filters that are convolved with patches of the input signal to produce *feature maps.* The discrete convolution between a time-series patch $x(t)$ and a kernel $w(a)$ is defined as (Goodfellow et al., 2016)

$$s(t) = (x * w)(t) = \sum_{a=-\inf}^{inf} w(t-a)\,, \tag{3.7}$$

where $*$ is the convolution operator. For a multi-channel input time series at channel c ($\mathbf{x}_{l-1}^{(c)}$), the m^{th} output feature map ($\mathbf{x}_l^{(m)}$) at layer l is given by

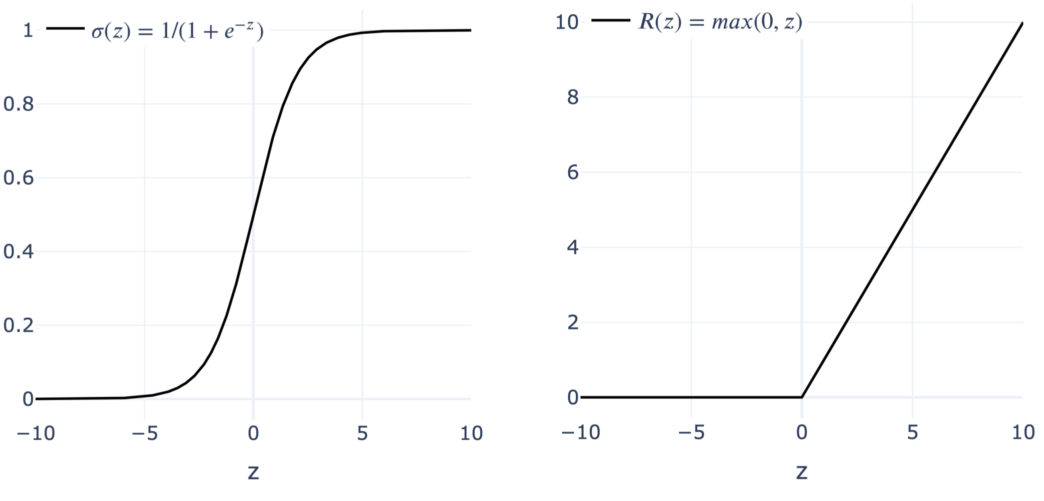
$$\mathbf{x}_l^{(m)} = \sum_{c=1}^{C} \mathbf{W}_l^{(c,m)} * \mathbf{x}_{l-1}^{(c)} + \mathbf{b}_l^{(m)}, \tag{3.8}$$

where $\mathbf{W}_l^{(c,m)}$ is the m^{th} filter at channel c, $\mathbf{b}_l^{(m)}$ is the m^{th} bias vector, and C is the total number of input channels or features. The output from equation 3.8 is passed as input to an activation function to introduce non-linearity. Figure 3.4 shows two commonly used non-linear activation functions. The sigmoid function—a special case of the logistic function—is defined as

$$\sigma(z) = \frac{1}{1 + e^{-z}}. \tag{3.9}$$

Equation 3.9 is affected by the vanishing gradients problem when z is close to 0 or 1. To overcome this limitation, Nair and Hinton, 2010 introduced the rectified linear unit (ReLu). A ReLu leaves positive numbers unchanged, but sets all negative numbers to zero. It is defined as

$$R(z) = \max(0, z). \tag{3.10}$$



(a) (b)

Figure 3.4: The sigmoid (a) and ReLu (b) activation functions.

Pooling layer

A pooling layer is typically added after a convolutional layer to reduce the size of the feature maps by merging similar features into one (LeCun et al., 2015). A *pooling function* is used to modify the input feature maps. For example, the *max-pooling* operation returns the maximum value within the analyzed time window.

Fully-connected layer

After a series of convolutional layers, non-linear activation functions, and pooling layers are stacked, a fully-connected layer is the typical output layer on a CNN. The output vector of a fully-connected layer $\mathbf{x}_l$ is

$$\mathbf{x}_l = \sigma\left(\mathbf{W}_l \mathbf{x}_{l-1} + \mathbf{b}_l\right), \tag{3.11}$$

where is the activation function, $\mathbf{x}_{l-1}$ is the input vector, $\mathbf{W}_l$ is the weights matrix, and $\mathbf{b}_l$ is the bias vector. In the case of regression, the output layer in a CNN uses the identity function ($y(z) = z$) as activation since the output is a scalar unbounded value.

3.4.2 CNN training

Finding the optimal parameters of a CNN can be done by formulating the training as a minimization problem. Given the training pairs (x_i, y_i) of inputs x_i and targets y_i, the optimal parameters θ made up of weights and biases can be solved by minimizing the empirical risk (Lehtinen et al., 2018)

$$\underset{\theta}{\operatorname{argmin}} \sum_i L\left(f_\theta\left(x_i\right), y_i\right), \tag{3.12}$$

where f_θ is a mapping function like a CNN model and L is the loss function. For a regression task, the mean squared error (MSE), eq. 3.2, is commonly employed as the loss function. The network parameters can be trained by *back-propagation* (Rumelhart et al., 1986) and *gradient descent*. The idea is to pass the training examples through the CNN's hidden layers and compute the gradients to update the weights and bias at each node. There are some variations of gradient descent implementations used in deep learning (Goodfellow et al., 2016). *Stochastic gradient descent* (SGD) uses only a single example at a time to update the network parameters. On the contrary, *batch gradient descent* processes all the training examples simultaneously in a batch. Most optimization algorithms use more than one but less than all the training examples. The latter is known as *mini-batch gradient descent* which considers a random mini-batch from the entire training set to optimize the network parameters.

Optimizers are commonly used to automatically adjust learning rates for each parameter by modifying the step length of the gradient descent, thus improving the optimization process. The adaptive moment estimation (ADAM) method (Kingma and Ba, 2015) has gained popularity for training CNN. ADAM offers an efficient alternative to classic gradient descent by adapting individual learning rates to different parameters, thus improving training performance.

3.4.3 Data augmentation

Having fewer training examples than free parameters (i.e., weights and biases) can lead to poor network generalization (van der Baan and Jutten, 2000), reducing the model's ability to deliver good performance on unseen data. Often, generalization can be improved by splitting the examples into training and test sets or applying cross-validation strategies (section 3.2.2). However, in situations where a limited number of examples are available, data augmentation techniques such as noise injection have proven to be effective in reducing overfitting and improving generalization (Jiang et al., 2009).

In this work, we apply a double-noise injection technique to increase the number of available training examples. With the double-noise injection, input and output data are contaminated with random or structured noise (e.g., white Gaussian noise), which is assumed to be uncorrelated to the data (Lehtinen et al., 2018; Zhang and van der Baan, 2021). Furthermore, the standard deviation of the input may differ from that of the target, and both can be randomized. Double-noise injection builds on a property of the L_2 loss stating that the network learns to output the average of all plausible explanations (Lehtinen et al., 2018). Hence, if the input and target are contaminated with zero-mean Gaussian noise, the CNN can be trained without changing what it learns.

Chapter 4

Machine learning for microseismic prediction

4.1 Introduction

The first part of this chapter introduces a workflow for implementing the random forests (RF) algorithm (Sec. 3.3.2) to predict the microseismic cloud size at the end of an hydraulic fracturing stage. We use the first 70% of the stage data as the training set and the last 30% as the test set. A rolling window strategy is employed to derive statistical features from the engineering curves and the present microseismic cloud size to be used as input to the RF model.

We then introduce a physic-based method to forecast the size of the microseismic cloud based on an analytical diffusion model proposed by Shapiro et al., 1997 and detailed in section 2.3.3. In our implementation, the medium's apparent diffusivity is derived in real-time and used to forecast the microseismic cloud size at any future time of the stage.

Finally, we developed a second machine learning workflow employing a convolutional neural network (CNN) trained with the engineering curves to predict the future microseismic cloud size. We apply time-series cross-validation (Sec. 3.2.2) to train the model and make predictions in real-time. More specifically, a CNN model is trained with the first several minutes of data to estimate the cloud size in the upcoming minutes. We then increase the size of the input data, train a second CNN model, and predict for the next couple of minutes. Furthermore, to deal with limited data available on the first trained models, we apply double-noise injection (Sec. 3.4.3) to increase the number of available training examples and improve generalization.

The three proposed methodologies are applied to a real multi-stage hydraulic fracturing and microseismic dataset. We describe the field layout and dataset used in this chapter. Data pre-processing and correlation analysis of the engineering curves are presented, followed by the methodology employed on each method. Finally, we compare the prediction results at specific stages of the treatment and make recommendations on how model performance could be enhanced.

4.2 Geological setting, field layout, and dataset

The microseismic and engineering data used in this work were acquired during a multi-stage hydraulic fracturing treatment at the Hoadley Field in Central Alberta, Canada. The target formation of the stimulation was the Glauconitic member of the Lower Cretaceous Upper Mannville Group, a series of shallow marine sandstone deposits formed as an extensive barrier bar complex trending southwest-to-northeast (Eaton et al., 2014; Hayes et al., 1994; Rafiq et al., 2016). The Hoadley barrier complex is approximately 25 km wide and more than 200 km long, marking the northern limit of continental to marginal-marine depositional environment (Hayes et al., 1994). The Hoadley lithology includes eolian dune, tidal channel, leeve, interbar lagoon, and bar sediments (Chiang, 1984). The system contains multiple progradational marine sandstone bodies up to 32 km in length, each hosting several distinct reservoirs (Newbert and Trick, 1987). In the last years, hydrocarbon production from the low-permeability Hoadley sandstones has become economically viable thanks to the use of multi-stage hydraulic fracturing with horizontal drilling (Reynolds et al., 2012).

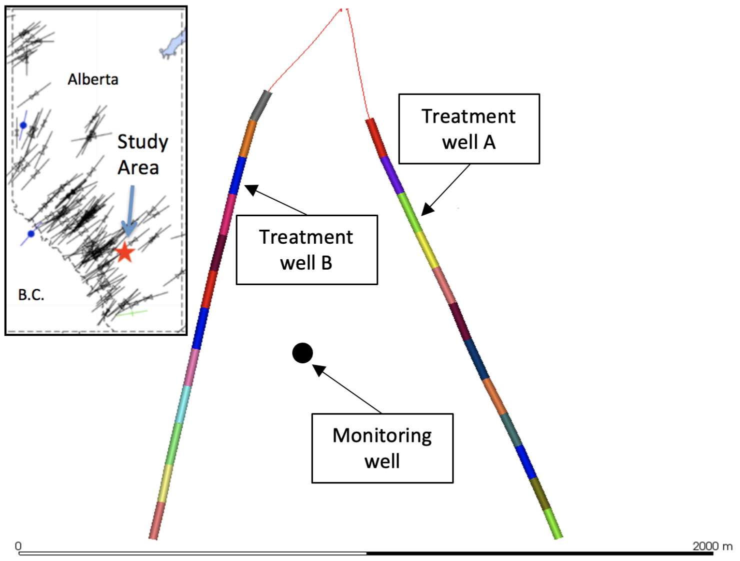


Figure 4.1: Plan view of the treatment configuration. Inset shows maximum horizontal stress direction (SHmax) and approximate location of the study area (red star). Modified from Eaton et al., 2014. The subset used in this work comes from the engineering and microseismic monitoring data on treatment well B.

The field layout consisted of two horizontal wells with 12 stages each, stimulated via open-hole completion (Eaton et al., 2014) (Figure 4.1). Microseismic data were recorded and processed in real-time using one vertical monitoring well with a 12-sensor down-hole array of triaxial geophones. The microseismic catalog consists of the estimated (x, y, z) coordinates, local date/time of occurrence, and moment magnitude of each event. The engineering curves were continuously monitored and time sampled at every second during the stimulation of the 24 stages.

We use a subset of the original data set comprising 425 minutes of five engineering curves and the locations and origin times of 666 microseismic events recorded during the stimulation of treatment well B (Figure 4.2). Most events have a moment magnitude $M_W < 0$ (Eq., 2.29) with spatial uncertainties of a couple of meters. To limit the temporal extent of our analysis, we do not use any post-pumping events in our data set (that is, only the events with occurrence times between the start of pumping of stage 1 and the end of pumping of stage 12 in well B are considered). The engineering time-series used in this work are:

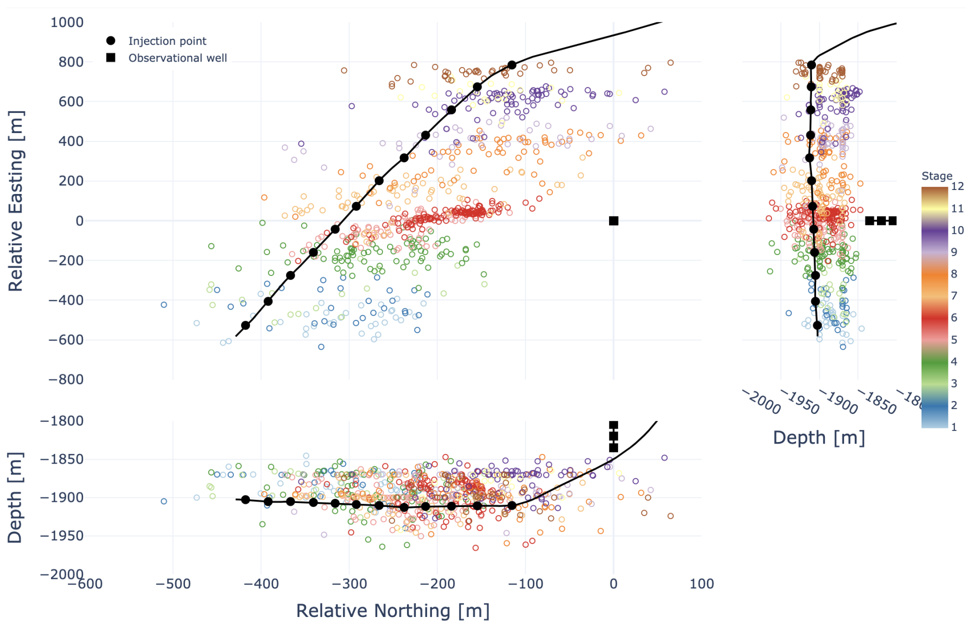
- Treating pressure (Pa).
- Bottom-hole pressure (Pa).
- Injection rate (fluid + sand + gas) (m^3/s).
- Proppant concentration (kg/m^3).
- Bottom-hole proppant concentration (kg/m^3).

4.3 Data pre-processing

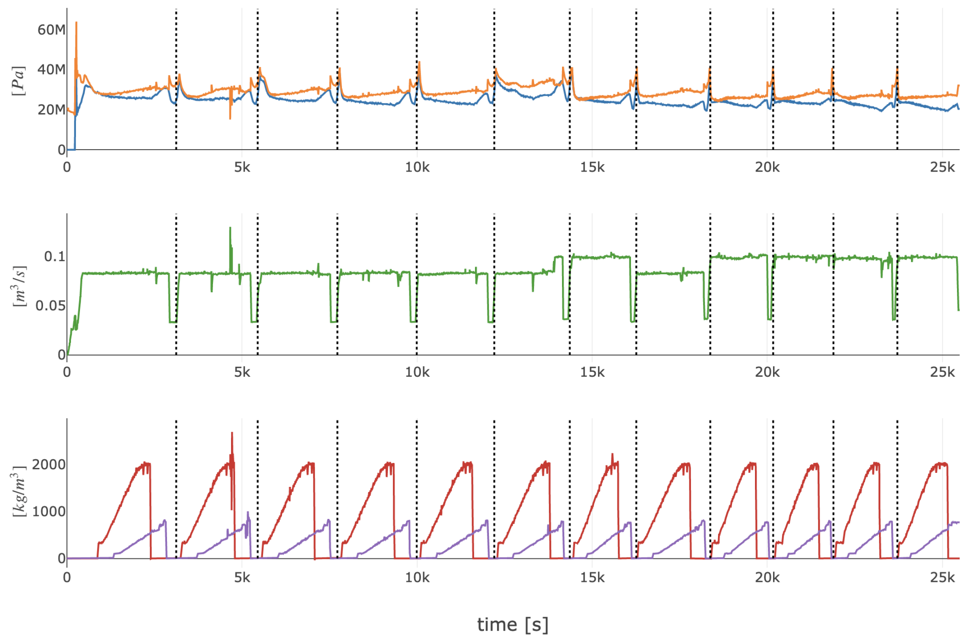
Microseismic event clustering for the Hoadley dataset was performed by the operator using a binning-by-stage-number approach. In this kind of clustering, events are assigned to a particular stage S if they occurred within a specified time window after the initiation of S and prior to the start of the next stage $S + 1$ (Eaton, 2018). However, Hoadley wells were stimulated via open-hole completion at 12 discrete injection points with a short time interval between stages. Hence, a strong spatio-temporal overlapping of events between stages is expected, meaning that a binning-by-stage separation will erroneously classify events that are a continuation of the hydraulic fracturing process of a previous stage.

We use the same approach as McKean et al., 2019 to reassign events into stages by considering their location and time of occurrence relative to the pumping time and location of each injection point. First, events are spatially assigned into stages based on a sum of squares combination of Euclidean distance from the center of the perforations. For S stages and N events the Euclidean distance d is defined as

$$d = \sum_{s=1}^{S} \sum_{n=1}^{N} ||c_s - c_n||_2, \tag{4.1}$$



(a)



(b)

Figure 4.2: Plan view and side views of the microseismic dataset colored by stage number assigned by the operator. Black circles show the perforation points for the 12 stages, and black squares show the geophones locations (a). Engineering curves used in this work. The top panel displays bottom-hole pressure (orange) and surface pressure (blue), middle panel injection rate (green), and bottom panel proppant concentration (red) and bottom-hole proppant concentration (purple). The vertical black dotted lines mark the start and end times of each stage (b).

where c_s are the spatial coordinates (x, y, z) of the perforation point of stage s and c_n the spatial coordinates of event n. Then, this is combined with the time difference between the start of each stage t_s and the occurrence time of each event t_n (i.e., $t = t_s - t_n$), such that the goal is to minimize the spatio-temporal constraint r defined as

$$r = \sqrt{d^2 + t^2} \tag{4.2}$$

This procedure reclassifies 369 events or 55% of the microseismic catalog (Figure 4.3). After applying the proposed reclustering strategy, we compute the microseismic cloud size in all stages to be used as labels for the training examples provided to the machine learning algorithms. We build $r-t$ plots (Shapiro et al., 2006; Shapiro et al., 1997), with r being defined as the distance of the microseismic events from the injection point in each stage as a function of time t (Section 2.3.3). A stage-by-stage analysis is used, meaning that events reassigned to a previous stage are not included in the $r-t$ plot of the next stage.

To calculate the radial distances r, we used a one-minute expanding window starting at the beginning of each stage. We computed the 90th percentile of the distance between the microseismic events and the perforation points within the expanding time windows. The 90th percentile was selected to avoid including outlier events into the microseismic cloud calculations. Furthest events may not be connected to the main fracture network, resulting in overestimating the actual size of the microseismic clouds. Figure 4.4 shows the $r - t$ plots for all stages in the well. Few events appear at radial distances $r < 20$ m from their reassigned injection point. The latter could be explained by events triggered due to tensile stress perturbations near the crack tip or due to microseismicity associated with fracture termination as described in figure 2.10c. Figure 4.5 shows a zoomed-in view of the microseismic clouds on selected stages of the treatment.

4.3.1 Correlation analysis

Real-time injection parameters recorded during hydraulic fracturing provide valuable information to understand fracture growth. Therefore, time dependen-

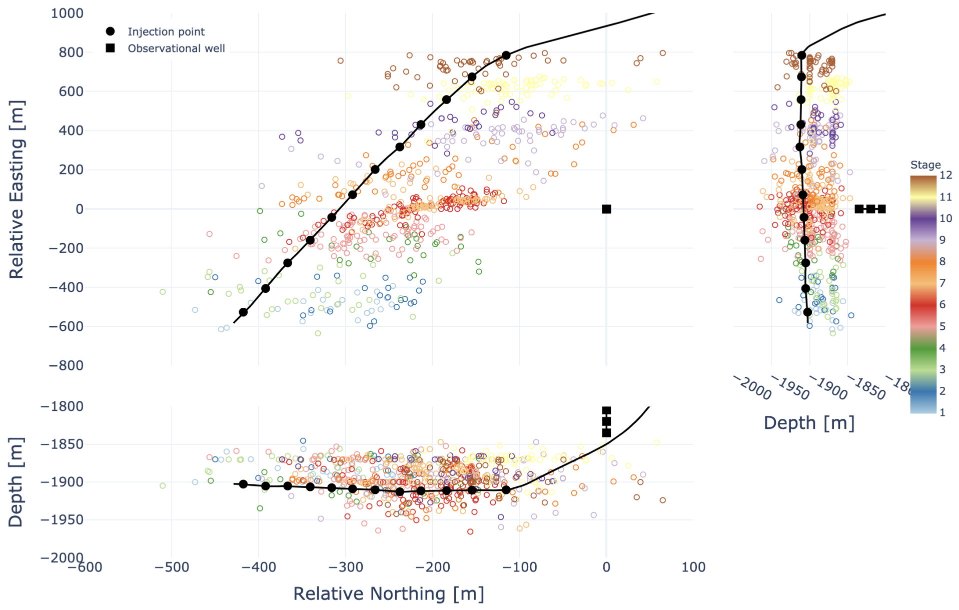


Figure 4.3: Plan view and side views of the microseismic dataset reclustered to each stage using the spatio-temporal constraint from equation 4.2. Compare to figure 4.2a.

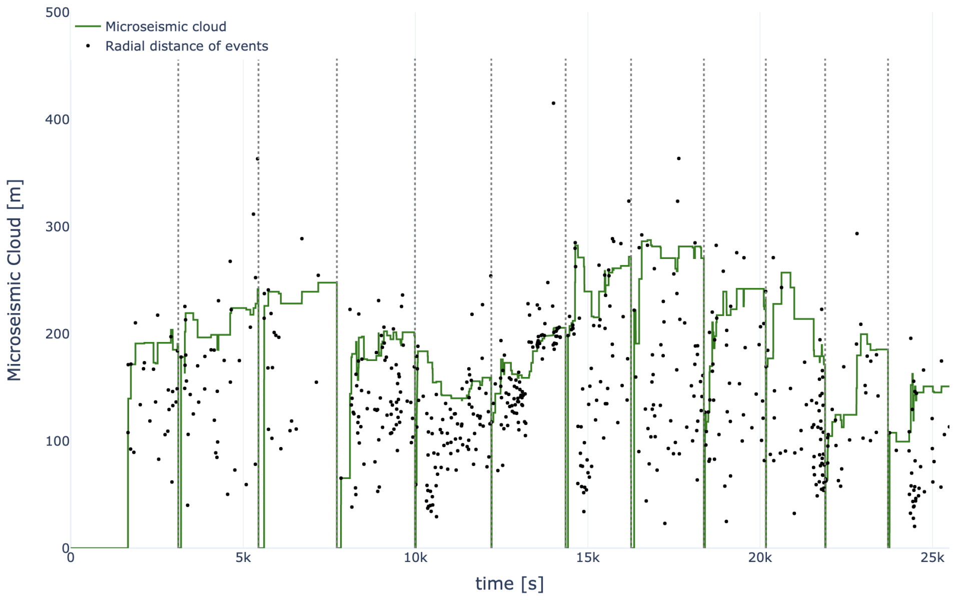


Figure 4.4: $r - t$ plots displaying the microseismic cloud size (green) for all stages in the treatment. Black points mark the radial distance r of events from their reassigned injection source as a function of time t. Vertical gray-dotted lines mark the start and end times of each stage.

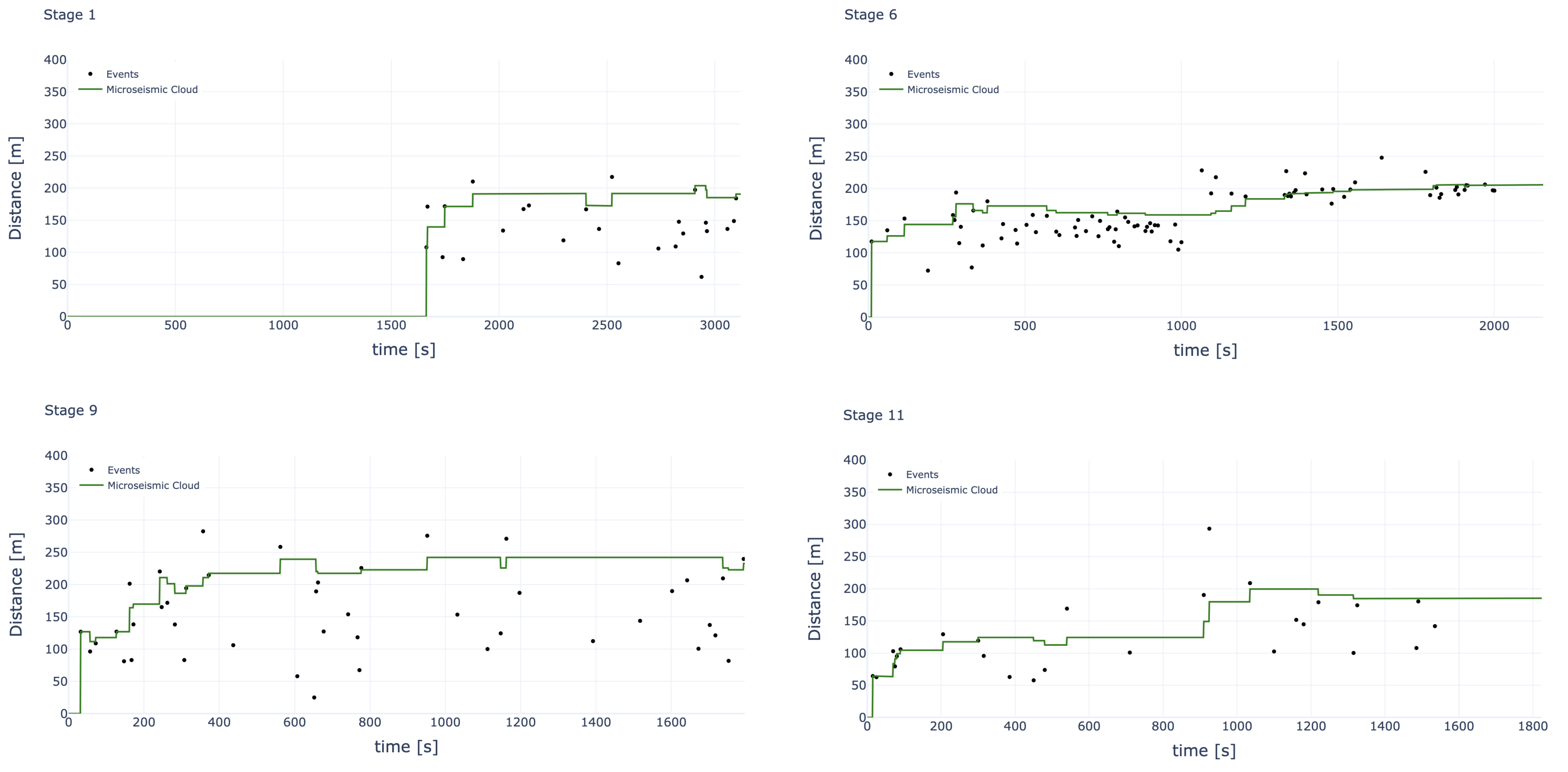


Figure 4.5: $r - t$ plots of stages 1, 6, 9, and 11. Black points show the radial distance of events from the injection source as a function of time. The green line displays the computed microseismic clouds.

cies should be expected between the injection curves and the spatio-temporal distribution of microseismicity. Before training the RF and CNN models, we investigate existing correlations between the available engineering curves and the microseismic cloud size for stage 6 of the treatment. The linear correlation between two variables x and y can be computed using the Pearson correlation coefficient $\rho_{x,y}$ defined as

$$\rho_{x,y} = \frac{\mathrm{cov}(x, y)}{\sigma_x \sigma_y}, \tag{4.3}$$

where $\mathrm{cov}(x, y)$ is the covariance and $\sigma_x \sigma_y$ is the product of the standard deviations of x and y. A perfect absolute correlation between the variables will yield a $\rho_{x,y} = 1$, while a $\rho_{x,y} = 0$ will be obtained if variables are not correlated. Figure 4.6a shows a Pearson correlation heatmap for the engineering curves and the microseismic cloud size in stage 6 after removing the mean and scaling each feature to a standard deviation of 1. In addition, figure 4.6b shows cross plots for some of the variables with large absolute correlation values. Proppant concentration has a strong negative correlation value of $\rho = -0.63$ with surface pressure. This is the typical behavior of a fracturing job, where a fall in pressure is observed after the breakdown (Figure 2.10), while proppant increases progressively to ensure a proper distribution within the fractures (Maxwell, 2014). The bottom-hole treating pressure (that is., the amount of pressure required to cause fracture extension) and the surface pressure display a positive correlation of $\rho = 0.51$. This is expected as the bottom-hole pressure is usually a time-lagged, larger magnitude expression of the surface pressure.

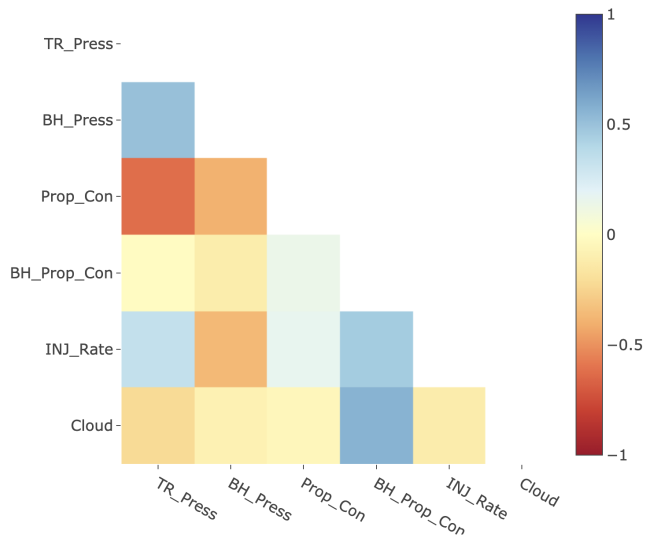
The microseismic cloud size displays a positive correlation value of $\rho = 0.56$ with the bottom-hole proppant concentration. The high correlation is most likely due to the sustained growth of proppant concentration with time, which resembles the growing behavior of the microseismic cloud on stage 6. There is a negative correlation value of $\rho = -0.22$ between surface pressure and microseismic cloud size. The latter grows over time while pressure decreases consistently after the breakdown.

To analyze changes in correlation with the microseismic cloud size over time, we computed a rolling correlation with the engineering curves using sliding time windows of length 10 minutes (Figure 4.7). In general, high absolute correlation values of $\rho > 0.7$ with treating pressure, proppant concentration, and both bottom-hole measurements can be observed during multiple periods of the stage. For example, at times $t = [600s : 800s]$ the microseismic cloud slightly contracts (Figure 4.4) when both pressures decrease after the breakdown (Figure 4.2b). On the contrary, during times $t = [1200s : 1400s]$ pressures, proppant concentrations, and microseismic cloud size increase. The less correlated feature over time is the injection rate, which remains constant through most of the stage (Fig. 4.2b). We expect the injection rate to have a small influence on predicting the size of the microseismic cloud. The stage correlation analysis will be validated when computing the feature importances derived from the trained random forest model.

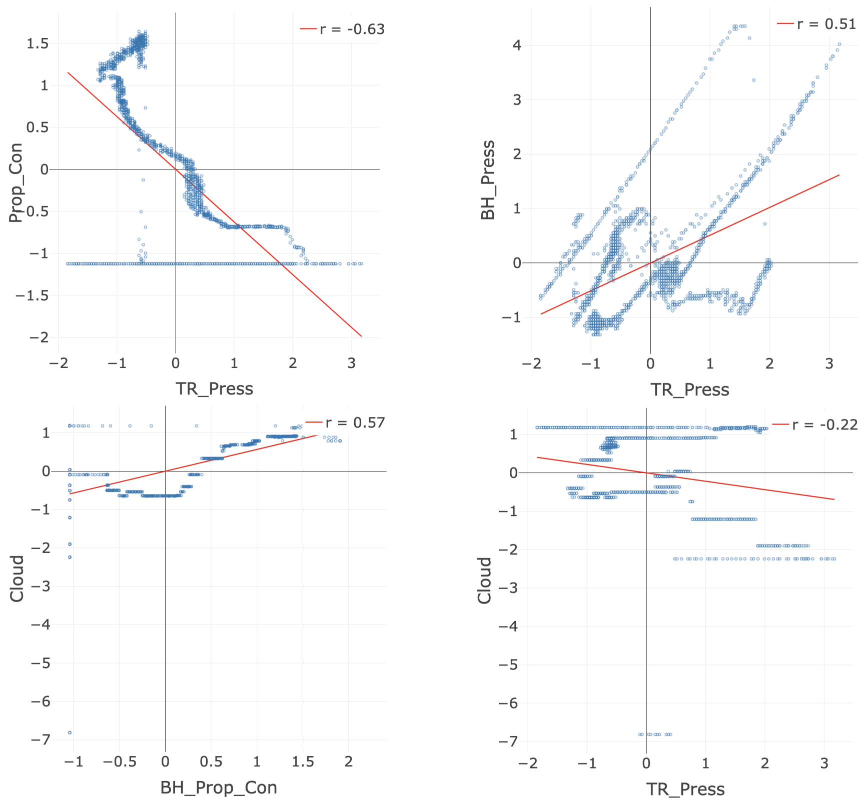
4.4 Machine learning-based model for microseismic prediction, a random forests approach.

One of the most common applications of machine learning in unconventional reservoirs is the prediction of a variable of interest, given some knowledge about the completion strategy and the reservoir's on-site geological features. For example, the random forests algorithm has been successfully applied to predict the cumulative production of hydraulic fractured wells and quantify the relevance of different completion parameters on production (Luo et al., 2018; Maucec and Garni, 2019; Wang and Chen, 2019). Typically, all of these studies use discrete input values as training data. For instance, features like well location coordinates, maximum proppant concentration, average bottom hole pressure, type of completion fluid, or the number of stages on each well are used as input features to train the predictive models.

Forecasting models using completion and geological features that distill pe-



(a)



(b)

Figure 4.6: Pearson correlation heatmap for the engineering curves and the microseismic cloud in stage 6 (a). Cross-plots of proppant concentration vs. surface pressure (top left), bottom-hole pressure vs. surface pressure (top right), cloud size vs. bottom-hole proppant concentration (bottom left), and cloud size vs. surface pressure in stage 6 (bottom right). Red lines show the best fitting line, and legends display the associated Pearson correlation value (b).

riods of the hydraulic fracturing treatment into single values as input offer a practical data-driven tool for engineers to optimize hydraulic fracturing treatments. However, picking a few representative points per well or stage ignores that hydraulic fracture growth is a dynamic process with a time-varying component (see equations 2.16 and 2.18). Thus, a different approach that considers the variations of the data available during completion (for example, the engineering curves) over time, should be used for developing fracture growth and microseismicity predictive models.

The random forests regression algorithm applied to continuous time-series data has been successfully used in geophysical problems. For example, Rouet-Leduc et al., 2017 used RF to predict the time remaining for an upcoming laboratory earthquake based solely on statistical features calculated from the continuous acoustic signal. Their results show that the RF algorithm applied over consecutive time windows detects small shear failure signals emitted from the fault gouge that presage failure. Similarly, Lubbers et al., 2018 construct an RF regression model to predict the shear stress and time to failure using only features derived from sliding time windows applied on the event catalog, rather than the continuous acoustic signal.

The radial distance between a point injection source and the location of a microseismic event is proportional to the square root of the time $\sqrt{t}$ of injection (see equation 2.18). Moreover, pressure analysis during pumping is commonly used to understand fracture evolution through multiple pressure diagnostic techniques (Belyadi et al., 2017; Economides and Nolte, 2000). The engineering curves and microseismic events are both monitored in real-time and properly time-sampled for interpretation. We propose training a random forest model with features derived continuously from the engineering curves to predict future microseismic cloud size values during a hydraulic fracturing stage.

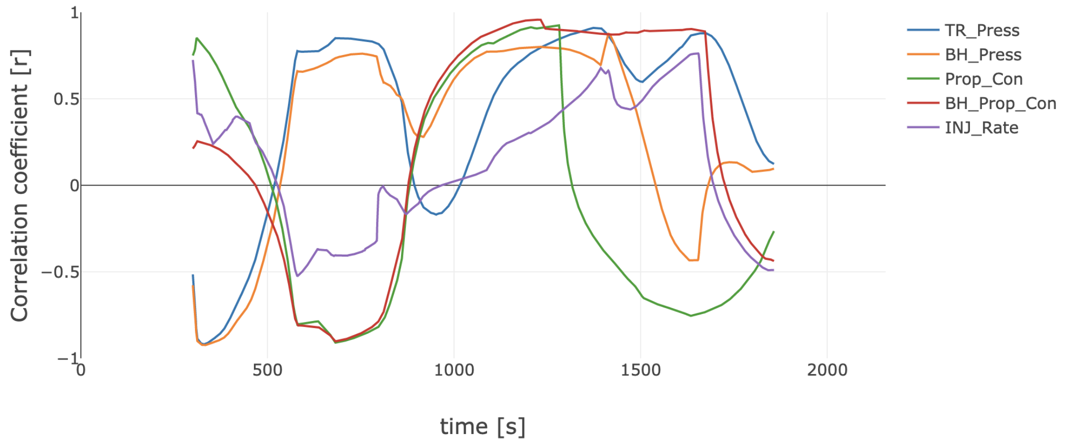


Figure 4.7: Rolling correlation between the engineering curves color-coded in the legend and the size of the microseismic cloud for stage 6. A time window of 600 seconds was used to compute the rolling correlation.

4.4.1 Machine learning dataset and feature selection

We use a similar approach as the one from Rouet-Leduc et al., 2017 to construct our machine learning dataset by computing statistical features from local time windows of the engineering curves and the size of the microseismic cloud. A stage-by-stage analysis is used, such that a different dataset and random forest model has to be built and trained on every stage. Each example contains a set of 66 statistical features derived from the 6 continuous time-series data (those are the 5 engineering curves plus the size of the microseismic cloud). Table 4.1 summarizes the input curves and features calculated on each local time window. In general, we compute the mean, variance, and higher order moments to characterize the distribution and energy of the input time-series. The n^{th} centered moment π_n of a signal f with length T is given by

$$\pi_n = \frac{1}{T} \int (t - \mu)\, f(t) dt, \tag{4.4}$$

with μ the mean of f over T. For example, the third-order moment, also known as skewness, measures a distribution's asymmetry around its peak, while the fourth-order moment—usually referred as kurtosis—measures peakness or flatness of the distribution. Also, we rely on different high and low percentiles to further characterize the input data distribution. Finally, other features include minimum, maximum, and cumulative values within the time windows.

We then label each example x_i made of several statistical features with the average size of the microseismic cloud on a time window located 5 minutes into the future y_i. The machine learning dataset is then

$$D_n = \{(x_i, y_i)_{i=1\ldots n}\}, \tag{4.5}$$

where n is the number of examples that depends on the size and offset between consecutive time windows. In this work, we used a window length $w = 30s$ with 90% overlap, meaning that the offset between windows $w_o = 3s$. The window parameters are somewhat arbitrary; longer window lengths and larger offset values will lead to fewer examples to use in the random forest model training and testing. Given that each stage has an average duration of 30 minutes, we consider that the selected window parameters produce enough examples to use in our RF model.

Table 4.1: Input parameters and main statistical features extracted from each time window.

Input time-series data	Statistical Features
- Treating pressure	- Mean
- Bottom-hole pressure	- Standard deviation
- Injection rate	- Kurtosis
- Proppant concentration	- Skewness
- Bottom-hole proppant concentration	- q01
- Microseismic cloud size	- q05
	- q95
	- q99
	- Minimum
	- Maximum
	- Cumulative

4.4.2 Random forest details

We use the *Random Forest Regressor* function from Pedregosa et al., 2011, which implements the random forest algorithm proposed by Breiman, 2001 (see section 3.3.2). Before training the model, one must specify the number of trees in the forest and the criteria to build each tree. For example, the tree's depth is one of the parameters that controls the complexity of the random forest. Deeper trees allow the model to perform better on highly complicated datasets. However, it can also make the model prone to overfitting (James et al., 2013). Other parameters include the minimum number of samples required to split an internal node and the maximum number of features to consider at each node. We employed 50 regression trees with bootstrap resampling and mean squared error (MSE) as the cost function (Equation 3.2). The maximum depth of the each tree equals 3. Also, we calculated the out-of-bag error to quantify the accuracy of the RF model (see equation 3.6). The remaining parameters were left as default.

4.4.3 Training strategy

We select the first 70% of the data available on each stage as the training set and use the remaining 30% as the test set. Figure 4.8 shows the engineering curves and the microseismic cloud size on stage 6. Training data $t = [0s : 1509s]$ is shown in blue and test data $t = [1509s : 2157s]$ appears in green. We then build the machine learning dataset following the strategy detailed in section 4.4.1. The training set on stage 6 has a size $[383, 67]$, representing 383 examples each with 66 features and a label with the microseismic cloud size in the next 5 minutes. Similarly, the test set has a size $[96, 67]$. Machine learning models perform better with features on a similar range (Bishop, 2006). However, the engineering curves differ by several orders of magnitude (Figure 4.2b). Hence, we standardize the training set such that each variable has zero mean and unit standard deviation. The same standardization is applied to the test set.

We trained the random forest model as defined in section 4.4.2 using a 5-fold cross-validation strategy (see section 3.2.2). More specifically, during each fold, we train the model using $k - 1$ subsets, validate using the remaining fold, and make a prediction using the current model on the test set. We emphasize that the test set is independent of the training procedure, the 5-fold cross-validation is applied exclusively to the training set. Finally, the resulting model prediction is the average test set prediction over the five folds.

A workflow for the training, validation, and prediction of the proposed random forest model is as follows:

1. Divide the stage data (i.e., engineering curves and microseismic cloud size) into a training (first 70% of time samples) and test set (remaining 30% of time samples).

2. Create the machine learning datasets on the training and test data using a sliding window of length 30 seconds with 90%. On each time window, compute the statistical features displayed in table 4.1 and label the examples with the average size of the microseismic cloud in a time window at the next 5 minutes.

3. Standardize the machine learning training set by subtracting the mean and scaling to unitary standard deviation. The test set is subject to the same standardization.

4. Define the random forest details. In this case, 50 trees, a maximum depth of 3 on each tree, and bootstrap resampling with the MSE error function.

5. Train the random forest model using the k-fold cross-validation strategy. In this work we use $k = 5$. Calculate the model error as the average validation set error across the 5 trials.

6. Predict on the test set at each fold and aggregate the predictions from all 5 trials.

4.5 Physics-based model for microseismic prediction, a diffusion approach.

In the absence of any dominant planar hydraulic fracture, the spatiotemporal pattern of the microseismic cloud L over time t can be analyzed using the

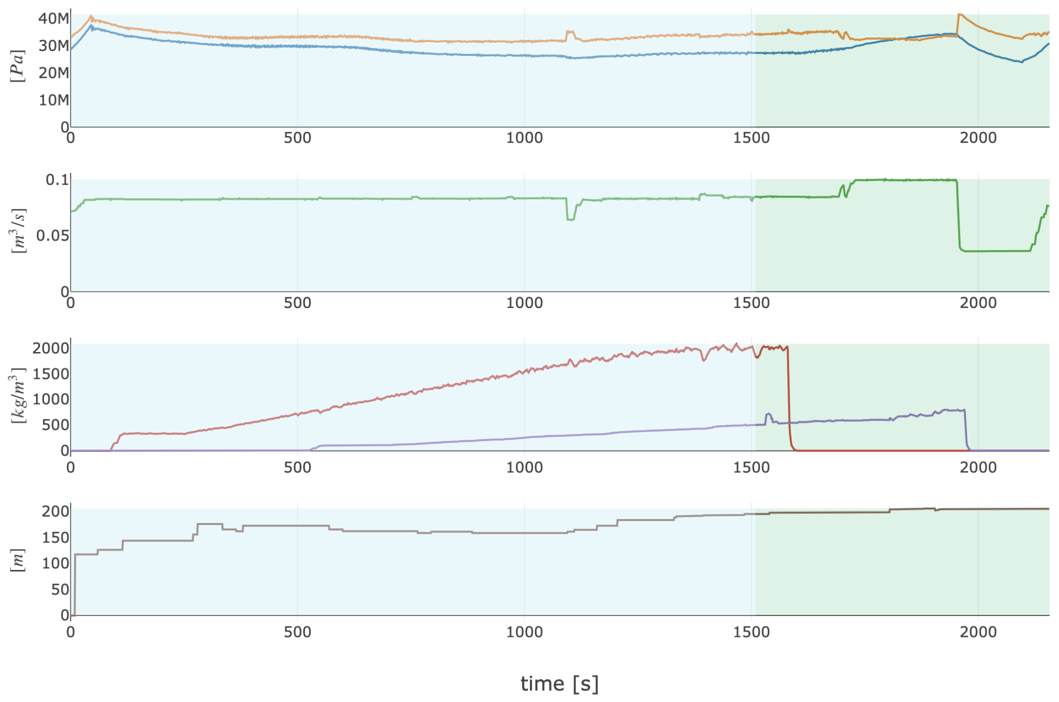


Figure 4.8: Input curves used to derive the statistical features of the machine learning dataset of stage 6. From top to bottom, bottom-hole pressure (orange), surface pressure (blue), injection rate (green), proppant concentration (red), bottom-hole proppant concentration (purple), and microseismic cloud size (brown). The blue background marks the data used in the training set, and the green background shows the data on the test set.

triggering front equation shown in 2.18. Here it is repeated for ease of understanding. The triggering front equation is defined as (Shapiro et al., 1997)

$$L = \sqrt{4\pi D t}, \tag{4.6}$$

where D is the medium's hydraulic diffusivity. Assuming a constant growth of the microseismic cloud proportional to the squared root of time, we define the model function equation $\hat{L}_t$ as

$$\hat{L}_t = \hat{\alpha}_t \sqrt{t}, \tag{4.7}$$

where $\hat{\alpha}_t$ is a scalar that relates to the apparent diffusivity of the medium $\hat{D}_t$ as

$$\hat{D}_t = \frac{\hat{\alpha}_t^2}{4\pi}. \tag{4.8}$$

We propose fitting equation 4.7 to the microseismic cloud observations in real-time. The goal is to find the optimal parameter $\hat{\alpha}_t$ that minimizes the sum of the squared errors between $\hat{L}_t$ and the size of the microseismic cloud at times $t = [t_0 : t_1]$, with t_0 and t_1 being the considered start and end times used in the fitting. We use the Levenberg-Marquardt method (Nocedal and Wright, 2006), which combines the gradient descent and Gauss-Newton algorithms, to find the optimal model parameter $\hat{\alpha}_t$. The latter can then be used as input to equation 4.8 to derive the apparent diffusivity $\hat{D}_t$ as a function of time.

We use the estimated apparent diffusivity of the medium to forecast the size of the microseismic cloud at any future time step using the triggering front equation (Eq. 4.6). The microseismic cloud in all stages does not vary significantly from second-to-second (Fig. 4.4). Therefore, we resampled all input data (i.e., microseismic cloud and engineering curves) to minutes. To honor the condition that all parameters must be derived in real-time, we fit triggering fronts using every additional minute of microseismic data as it became available during treatment. A flow chart displaying the proposed method at time t is shown in figure 4.9. We decided to fit the first $\hat{D}_t$ value after 5 minutes of microseismic data and the last one, 10 minutes before the end of each stage.

Figure 4.10 shows all the fitted triggering fronts (continuous lines) and forecasted microseismic clouds (dashed lines) for stages 1, 6, 9, and 11 (compare to figure 4.5). Each color represents the time used to fit equation 4.7 to the microseismic cloud observations, starting with red ($t = [0 : 5 \text{ minutes}]$) up to purple ($t = [0 : \text{end} - 10 \text{ minutes}]$). Also, color codes denote the associated estimated $\hat{D}_t$ values, which decrease as more data are provided to the fit. Often, the diffusivities inferred at the beginning of the stage will overestimate the actual size as the cloud's growth rate is larger earlier into the stimulation compared to later times. As more data become available, the $\hat{D}_t$ values stabilize, and the forecasts get closer to the size of the microseismic cloud towards the end of the stage. Nevertheless, the diffusion-based approach incorrectly estimates the microseismic cloud size during all times, even for the last fitted $\hat{D}_t$ values calculated 10 minutes before the end of pumping time.

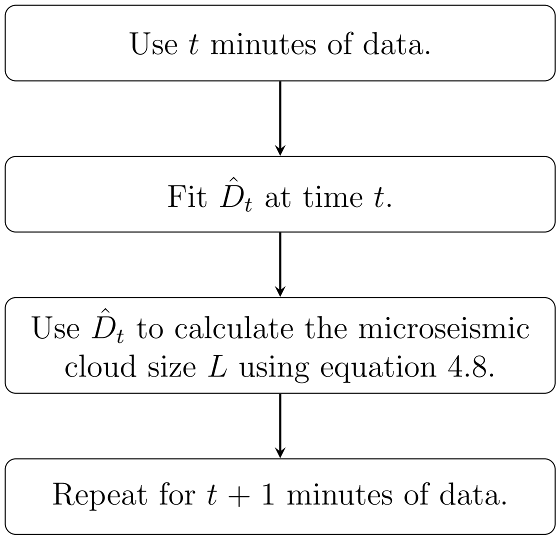

Use t minutes of data.
Fit $\hat{D}_t$ at time t.
Use $\hat{D}_t$ to calculate the microseismic cloud size L using equation 4.8.
Repeat for $t+1$ minutes of data.

Figure 4.9: The proposed methodology for applying the a diffusion-based approach to forecast the microseismic cloud size at time t.

4.6 Machine learning-based model for microseismic prediction, a convolutional neural network approach.

Time-series forecasting can be defined as predicting a future variable by analyzing the historical sequence of observations (Box et al., 2015). Recently, forecasting methodologies using convolutional neural networks (CNN) have been applied for optimizing hydraulic fracturing operations in areas like real-time surface pressure prediction (Ben et al., 2020). This section introduces a convolutional neural network (CNN) approach to forecast the microseismic cloud size. We first describe the network architecture, followed by the training strategy used.

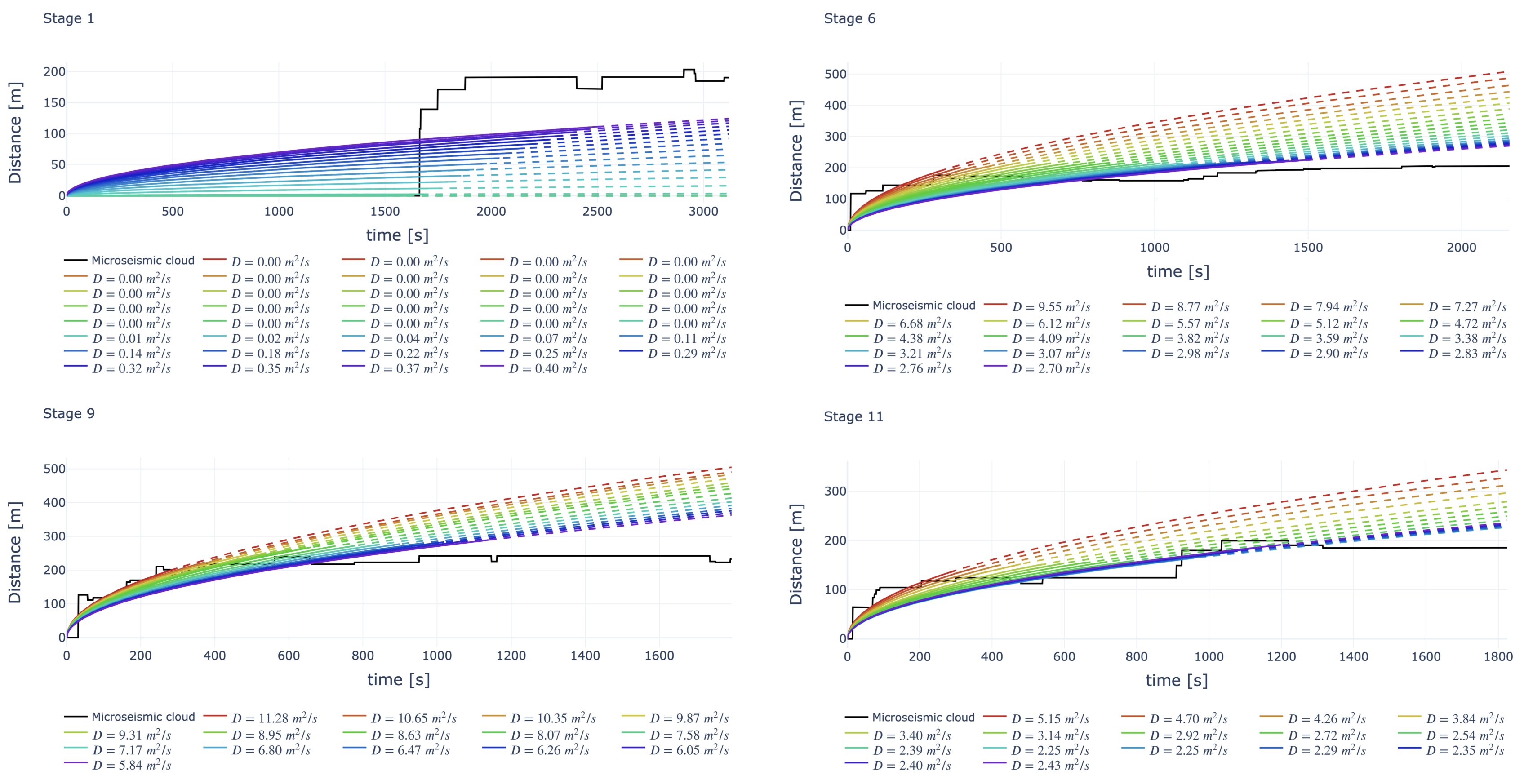


Figure 4.10: Fitted microseismic triggering fronts (solid line) for stages 1, 6, 9, and 11. Predictions appear as dashed lines. Colors vary gradually between red ($t = [0 : 5 \text{ minutes}]$) and purple ($t = [0 : \text{end} - 10 \text{ minutes}]$). Color labels show associated estimated $\hat{D}$ values.

4.6.1 Network architecture

A 1D CNN is used to detect temporal correlations between the input data and the future size of the microseismic cloud [1]. Table 4.2 summarizes the proposed network architecture. We use the same 6 time-series used in the RF model (Section 4.4.1) as input to the network. That is, 5 engineering curves: surface pressure, bottom-hole pressure, injection rate, proppant concentration, and bottom-hole proppant concentration, as well as past observations of the microseismic cloud size. First, the input multichannel time-series is fed into a 1D convolutional layer with $m = 16$ parallel feature maps and a kernel size k of $(6, 1)$. The input time series is a 3D tensor of size $(N_c \times N_s \times C)$, where N_c is the batch size or the number of samples and N_s is the number of time-steps used. All convolutions use no padding and a stride $s = 1$. The latter defines the step size of the kernel when sliding through the input time-series. The output then goes through a ReLu activation function to introduce non-linearity (Section 3.4). Therefore, the output from the first hidden layer has a size of $(N_c \times N_s - k + s \times m)$, with the last dimension representing the number of convolutional filters.

The convolutional layer's output feature maps are passed as input to a hidden dense (i.e., fully-connected) layer with 16 units and a ReLu activation function. Finally, the output from the second hidden layer goes through a last fully-connected layer, which returns a single value containing the microseismic cloud size forecast at a future time lag. In total, the network has 2 hidden layers and 881 free parameters (Table 4.2). Note that no pooling layer was included in our implementation as we used a shallow network architecture with a few hidden layers and neurons. The reasons for having a simple architecture with a few free parameters are twofold. First, we do not have a vast data set for training very deep, complex models. Second, we aim to predict the cloud size in real-time, meaning that each CNN needs to be trained and must forecast in a few seconds.

[1]The CNN used in this work was built and trained following the forecasting tutorial in TensorFlow available at https://www.tensorflow.org/tutorials/structured_data/time_series.

Table 4.2: The proposed CNN architecture.

Type	Feature maps	Kernel Size	Units	Free parameters
1D Convolution (16 outputs)	16	(6,1)	–	592
Activation (ReLu)	–	–	–	0
Dense 1 (16 outputs)	–	–	16	272
Activation (ReLu)	–	–	–	0
Dense 2 (1 output)	–	–	1	17
Total free parameters	–	–	–	881

4.6.2 Network training

The network parameters are trained by back-propagation using stochastic gradient descent and the mean squared error (MSE) as the loss function (Section 3.4.2). The adaptive moment estimation (ADAM) method with default settings was used to update and optimize the network parameters. Finally, the network was trained for 20 epochs.

For a real-time prediction system, one would like to update the model as new data become available. Thus, we train successive CNNs and cross-validate each model's predictions using a rolling forecasting strategy (Section 3.2.2). The earliest CNN model is trained after 10 minutes of injection and microseismic monitoring. We use the first 5 minutes as the training set ($t = [0 : 5]$) and label the example with the cloud size at 10 minutes ($t = [10]$). To validate the model, we use the time-series data from 5 to 10 minutes ($t = [5 : 10]$) and forecast the cloud size at 15 minutes ($t = [15]$). A second CNN gets trained one minute later using the first 6 minutes of data ($t = [0 : 6]$) and forecasting the size of the cloud at time 16 minutes ($t = [16]$). The rolling forecasting strategy is repeated up to 10 minutes before the end of each stage ($t = [0 : \text{end} - 10]$), with a new CNN model trained every minute.

The first models have access to very few data points. For instance, the CNN trained at 10 minutes can only use 30 data points for building the training and validation sets (that is, 5 minutes $\times$ 6 features = 30). We increase the number of training and validation examples through all the models trained over time using the double-noise injection technique (Section 3.4.3). In our implementation, the standard deviation of the added noise equals 1 for both input and target. We augment the training examples by a 50-fold increase, providing each CNN model with 50 input 3D tensors of size $(1 \times 6 \times 6)$.

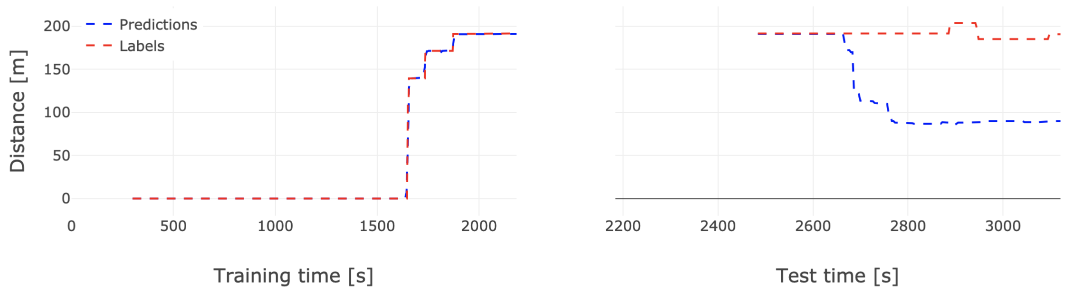
A step-by-step workflow for the training and validation of the proposed CNN model at time step t is as follows:

1. Resample the input time-series data from seconds to minutes.
2. Get the time indices for training and validation sets using the rolling-forecasting strategy depicted in figure 3.2.
3. Normalize the training set by subtracting the mean and scaling to unitary standard deviation. The validation set is subject to the same normalization.
4. Add white Gaussian noise with a standard deviation of 1 to both input and target on the training and validation sets.
5. Train the CNN model for 20 epochs using the ADAM algorithm as the optimizer and mean-squared error (MSE) as the loss function.
6. Make a prediction using the validation set. The model forecasts the size of the microseismic cloud 10 minutes after the last time index used in the training set.
7. Convert the prediction back to the original data scale.
8. Repeat steps 2 through 7 for time $t + 1$.

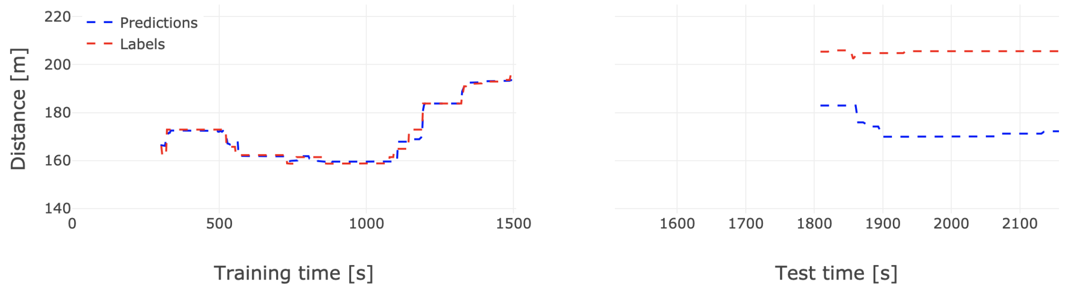
4.7 Results

4.7.1 RF-based approach

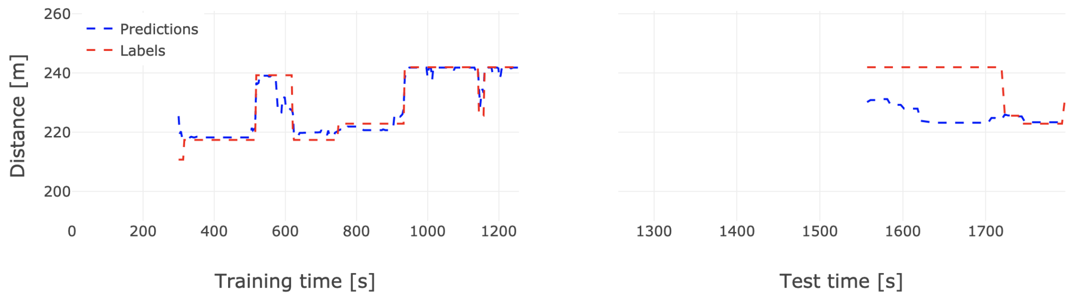
Figure 4.11 shows the random forest model predictions on the training and test sets for stages 1, 6, 9, and 11. Predictions are made 5 minutes into the future



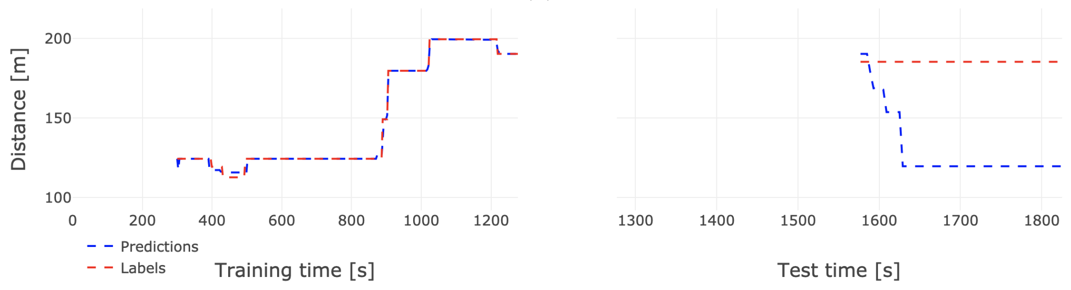
(a)



(b)

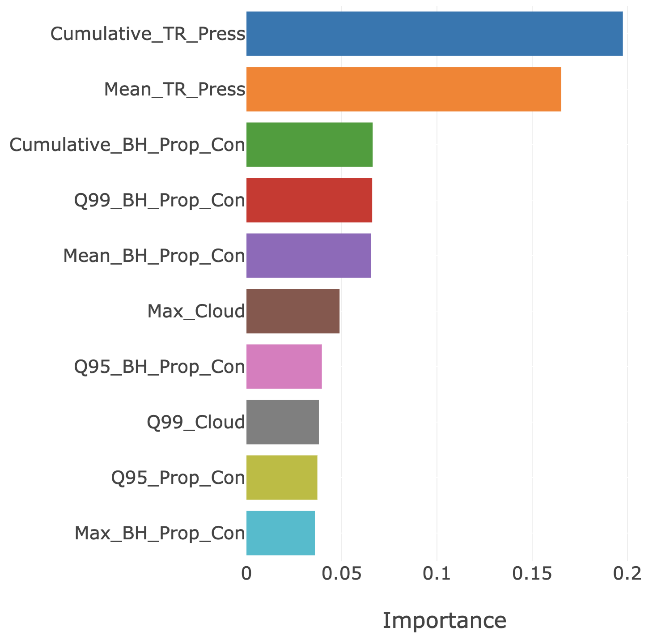


(c)

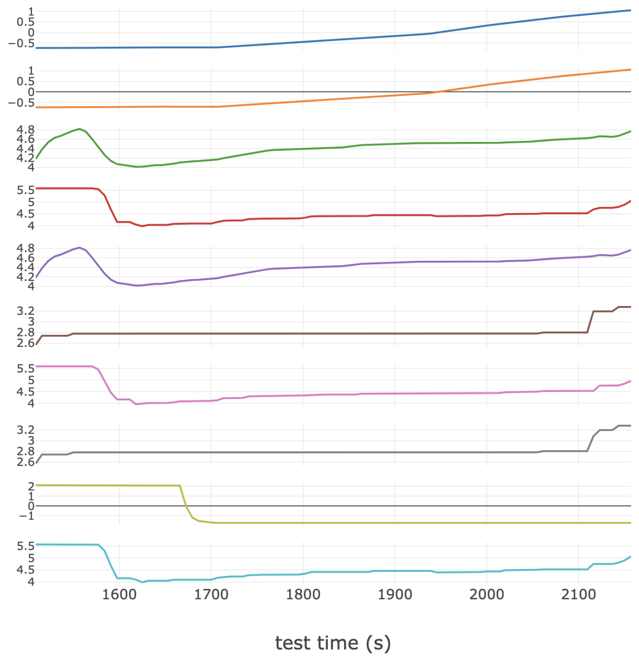


(d)

Figure 4.11: Random forest labels (red) and predictions (blue) on the training (left side) and test (right side) sets for stages 1 (a), 6 (b), 9 (c), and 11 (d). Microseismic cloud size predictions start 5 minutes after the first time sample on each set.



(a)



(b)

Figure 4.12: Normalized feature importance of the top 10 variables on stage 6 used to predict the microseismic cloud size 5 minutes into the future (a). Higher importance means more relevant variables. The same features in (a) computed in the test set (b). Curves are color-coded by their corresponding bar color.

using only statistical features derived within a single time window of the engineering curves and the present size of the microseismic cloud. We quantify our model's accuracy using the mean absolute error (Eq. 3.2) shown in table 4.3 (lower MAE means higher accuracy). In general, the model accurately fits the training data in most of the stages. However, in stage 9 (figure 4.11c, left), the RF misses some periods with rapid growth and contraction of the microseismic cloud, thereby resulting in higher training and validation set MAE compared to the rest of the stages. The latter could be addressed by increasing each tree's depth in the forest to allow the model to fit more complex cloud behavior. Nonetheless, as seen in the test set MAE for all stages (right column in table 4.3), better accuracy in the training set does not equate to a better generalization in the test set. Even though we are using a relatively small forest (i.e., 50 trees) with shallow trees (i.e., maximum depth of 3), our results exhibit a high degree of overfitting.

Figure 4.12 shows the 10 most relevant features on stage 6 for predicting the microseismic cloud size in the next 5 minutes. Variable importance of a feature x_m is computed as the normalized total reduction brought to equation 3.4 by the out-of-bag observations containing x_m in the training set. The 2 most relevant variables correspond to the mean, and cumulative values computed from the surface pressure. The remaining top features are mostly derived from the bottom-hole proppant concentration curve and the microseismic cloud size. These results are consistent with the stage correlation analysis done in section 4.3.1. Figure 4.12b displays the normalized 10 most relevant feature curves in the test set color-coded as in figure 4.12a. All the variables computed from the bottom-hole proppant concentration curve show a sharp decrease around $t = 1550s$. This is around the time when the proppant concentration shuts down (figure 4.8), and five minutes before the model predicts the first decrease in the test set microseismic clouds (figure 4.11b, right). Out of the 10 top features, 6 are related to either surface or bottom-hole proppant concentration. Therefore, the shut in of proppant injection at the end of each stage is one of the most likely causes of underestimated predictions during the 4 stages analyzed (Figure 4.10).

The low prediction accuracy achieved in the test set of all stages (Figure 4.11, right) is influenced by the training and testing strategy used in our RF model implementation. The input time-series data (i.e., the engineering curves and the present microseismic cloud size) differ significantly between the training and test sets. For example, at the end of all stages, proppant concentration goes to zero. Moreover, the proposed training and testing strategy limits the model's applicability for real-time predictions, as it requires at least 70% of the treatment data for training, leaving only the last 30% of the stage for predictions.

Table 4.3: Mean absolute error (MAE) in the training, validation, and test sets. The validation MAE was averaged over the 5-fold cross-validation.

Stage	MAE training set	MAE validation set	MAE test set
1	0.20	0.64	60.36
6	0.92	1.01	32.67
9	1.88	2.40	11.31
11	0.56	0.88	56.07

4.7.2 Diffusion-based approach

Forecasting results using the proposed diffusion-based approach (Section 4.5) on stages 1, 6, 9, and 11 are shown in figure 4.13. One advantage of this approach is that once $\hat{D}_t$ is calculated, it can be used to predict the size of the cloud at any time using the triggering front equation 4.6. This allows us to forecast the size of the microseismic cloud 10 minutes into the future (filled circles) and at the end of the stage (empty circles). The accuracy in stages with a growing microseismic cloud during later times (e.g., stages 6 and 11) is higher than that seen in stages with constant clouds towards the end of the injection time like in stage 9 (Figure 4.13, bottom left). The latter is a direct consequence of the fixed $t^{1/2}$ dependence of the triggering front equation, implying continuous microseismic cloud growth over time. Stage 1 represents a challenge for forecasting due to the microseismicity's nature at the beginning of the treatment. It takes around 27 minutes for the first microseismic events to be detected, causing an underestimation of the cloud size at all times in that stage.

Two main conclusions can be drawn from the results achieved with this physics-based diffusion approach. First, predictions get better with time because the cloud's growth rate is higher at the beginning than towards the end, thus reducing the estimated diffusivity and scaling the fronts closer to the microseismic cloud as more data are provided to the fit. Second, except for stage 1, the accuracy of forecasts on all stages is low because predictions are based on overestimated hydraulic diffusivities.

4.7.3 CNN-based approach

Figure 4.14 shows the forecasting results obtained using the proposed CNN model on stages 1, 6, 9, and 11 of the treatment. Forecasts are done following the training-validation strategy detailed in section 4.6.2. In total, six time-series of surface and bottom-hole hydraulic fracturing monitoring measurements were used as input to the CNN models (Sec. 4.6). After double-noise injection, the first CNN model trained on each stage has access to 1800 training examples, exceeding the number of free parameters by a factor of 2 (Table 4.2). Then, for every additional minute available during training, the number of input examples increases by 300.

In general, the CNN predictions are better during times without much cloud growth. For example, the last forecasts on stage 9 (Figure 4.14, bottom left) show a good accuracy when the cloud remains flat. In such cases, the model learns to output constant cloud sizes similar to the historical observations. However, predictions on stages with steadily growing microseismic clouds (e.g., stages 6 and 11) underestimate the size by tens of meters. In particular, the accuracy decreases when the cloud size expands during a short time, that is displays growth spurts (see figure 4.14, bottom right between 15 and 20 minutes). In these cases, the CNN model fails to capture the complex spatio-temporal behavior of microseismicity, resulting in inaccurate forecasts. Predictions for stage 1 (Figure 4.14, top left) cannot account for the rapid growth of the micro-

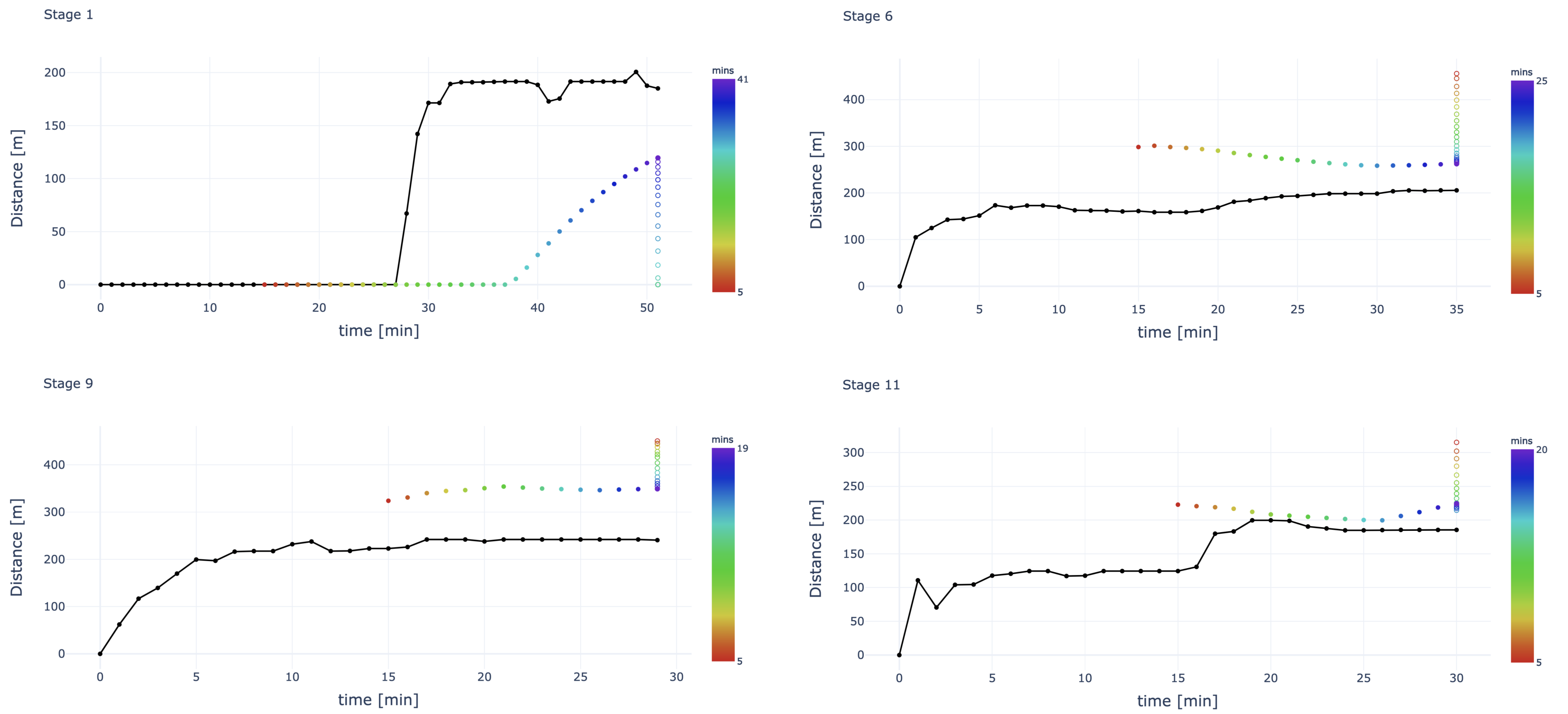


Figure 4.13: Microseismic cloud size forecasting using the diffusion-based approach. Forecasts are done in real-time for the next 10 minutes (filled circles) and at the end of stages 1, 6, 9, and 11 (empty circles). The color bar shows the time in minutes used to fit the triggering front and calculate the hydraulic diffusivity $\hat{D}_t$. Compare to figure 4.10 for the continuous forecasts.

seismic cloud in the middle of the stage, thus underestimating the size during most times. One significant limitation of our CNN methodology is that it can only forecast at a predefined future time sample because it is subject to the data availability over time. Using the CNN-based approach, the final size of the cloud can only be forecasted 10 minutes before the end of the stage.

To investigate the effect of using a different model architecture, we compared the predictions between the proposed model and two other architectures with more free parameters. More specifically, we kept the same number of hidden layers, training epochs, and noise realizations but changed the number of feature maps and units of the hidden layers to 64 and 256, resulting in 6593 and 75521 free parameters respectively, for each of the new models. We found that training more free parameters do not improve predictions considerably than the proposed CNN (Figure 4.15). We argue that this could be due to ignoring the physics of hydraulic fracturing and microseismicity during training. Although the injection history contains relevant information about the behavior of the microseismic clouds, more work is needed to include physical principles into the predictive models.

4.8 Conclusions

We presented an implementation of the random forests regression algorithm to forecast the size of the microseismic cloud during any stage of a hydraulic fracturing treatment. The random forest model was trained with statistical features computed from local time windows of the engineering curves and the microseismic cloud size. No past or future information outside of the current time window is used when making a prediction. Thus, we characterize the local distribution of the engineering curves to predict the microseismic cloud size in the next 5 minutes of the treatment. The main advantage of our RF-based approach is that it only relies on data that is routinely collected during a hydraulic fracturing stimulation. Moreover, once the RF model is trained, it does not need to be updated as more data becomes available later into the stage.

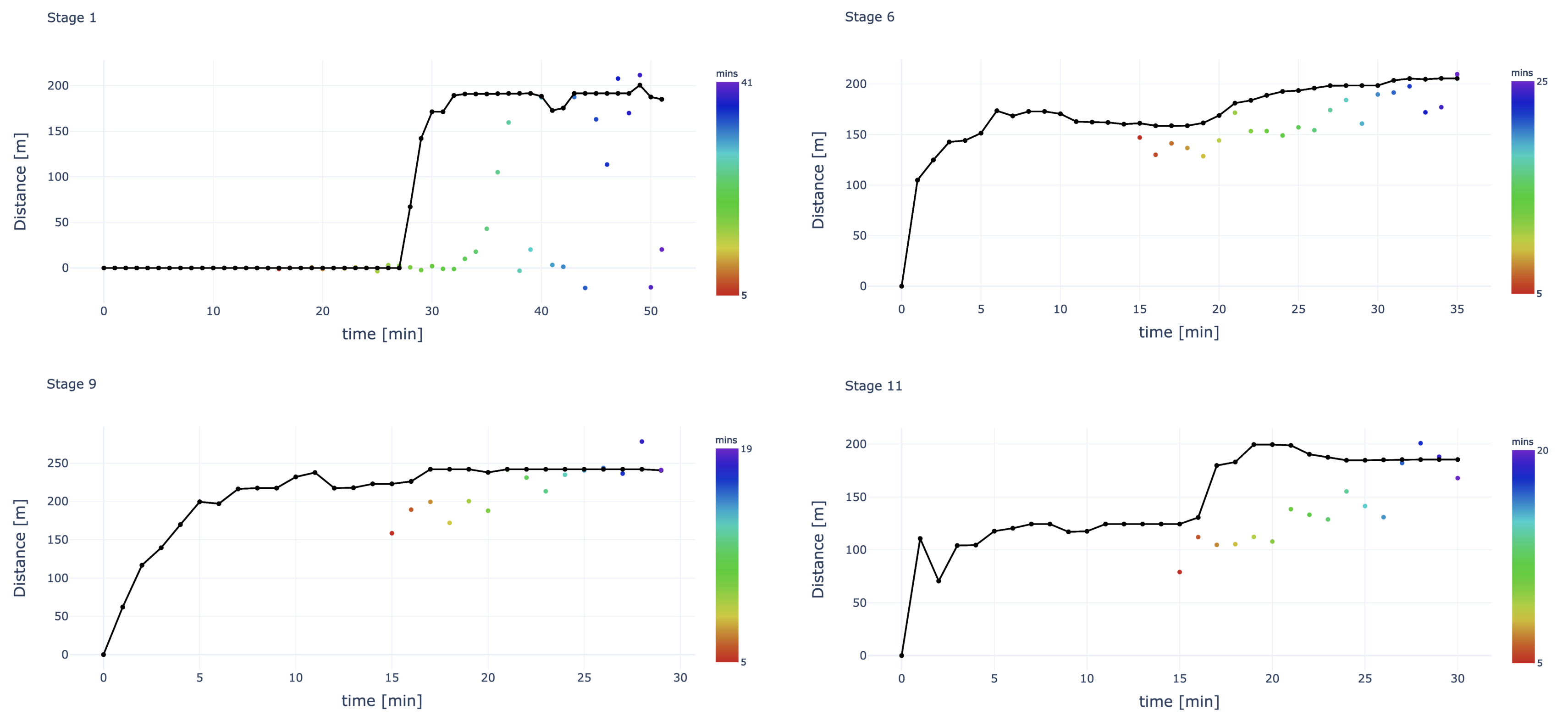


Figure 4.14: Microseismic cloud size forecasting using the CNN-based approach for stages 1, 6, 9, and 11. The color bar shows the time in minutes used to train each CNN model. Compare to the filled circles in figure 4.13.

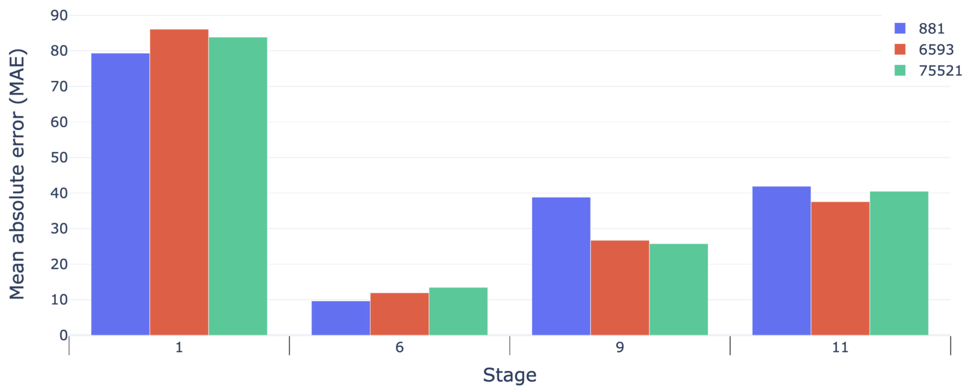


Figure 4.15: Average mean absolute error (MAE) for all forecasts on stages 1, 6, 9, and 11 using 3 CNN model architectures with 881 (model depicted in table 4.2), 6593, and 75521 free parameters each. Legend shows the number of free parameters.

However, we showed that the proposed RF model suffers from overfitting and does not generalize on the training set. The most likely reason for underestimated predictions is the training-testing strategy used. Injection monitoring curves change dramatically at the end of the stage. For instance, the proppant concentration goes to zero, and the injection drops until the next stage resumes. We conclude that the test set engineering curves do not adequately describe the features of interest in the treatment (i.e., it is not representative). Furthermore, the RF implementation limits the applicability for real-time predictions, as the model does not update with newly available data and requires a large proportion of the stage data for training.

We found temporal correlations between the engineering curves and the size of the microseismic cloud. In particular, the bottom-hole proppant concentration and the surface pressure display high absolute correlation values with the cloud size. Our analysis does not imply a one-to-one relationship between the engineering curves and the microseismic cloud but rather shows that the spatial distribution of microseismicity can be directly compared to the injection strategy during specific periods in the stage. Even though the engineering curves contain valuable information to understand fracture growth and microseismicity, hydraulic fracturing involves the interplay of several physical processes, and most likely, a non-linear-fluid-rock interaction (Shapiro, 2015).

To achieve real-time microseismic cloud size forecasts, we presented and compared the results of a physics-based approach and a 1D CNN model. A diffusion-based approach uses past observations of the microseismic cloud to estimate hydraulic diffusivity values. The main advantage of such an approach is that once $\hat{D}_t$ is calculated at the present time step, it can be used to forecast the cloud size at any future time of the stage using the triggering front equation (eq. 4.6). However, results show that this method often incorrectly forecasts the evolution of the microseismic cloud size by 100s of meters, especially for predictions done with early fitted hydraulic diffusivities.

The biggest limitation of a diffusion-based approach is that it only relies on one free parameter to forecast the size of the microseismic cloud, namely the effective hydraulic diffusivity of the medium. The model makes predictions solely based on the observed microseismicity on each stage, thus ignoring routinely measured data like the engineering curves. For this particular treatment, the monitoring curves were kept mostly constant and consistent from stage to stage (Figure 4.2b), justifying their exclusion to some degree. Nonetheless, physics predicts that the engineering parameters play an important role in the hydraulic fracture propagation (Economides and Nolte, 2000; Maxwell, 2014), making their exclusion less than ideal. Moreover, Barthwal and van der Baan, 2019 demonstrate that the apparent diffusivity in low-permeability rocks is predominantly influenced by the hydraulic fracture growth, which is an episodic process (van der Baan et al., 2016), and likely strongly influenced by local material and stress heterogeneities.

Unlike the diffusion-based approach, which is governed by an analytical expression, the machine learning-based methodology can extract and learn complex dependencies between input data and the microseismic cloud size. We showed that training a convolutional neural network with time-series data of the engineering curves can deliver better forecasts of the microseismic cloud than a diffusion-based approach and a RF model. Nevertheless, one disadvantage of the proposed CNN-based approach is that at the present time t, it can only forecast at a maximum predefined horizon of $t + h$ minutes, and cannot make any further prediction beyond this time until the next $t + 1$ minute of data becomes available.

Our results show that the CNN model's performance varies between stages, with better forecasts in situations where the cloud remains constant over time. Furthermore, we showed that training models with more free parameters do not improve accuracy compared to a network with a few parameters, raising the question of whether a deeper or more complex network is needed to improve predictions.

Looking closely at stage 1 (Figure 4.14, top left), it is clear that complex microseismic growth presents a challenge for forecasting. The absence of microseismicity during almost half of the stage could be due to insufficient injected volume to induce shear slip at pre-existing cracks. In other words, the initial delay is caused by the time required to build up sufficient stress and pore-pressure to cause local failure away from the perforation points. On the contrary, it does not take much time to see the first events for the next stages because, most likely, the reservoir is already under sufficient induced stress and pore-pressure perturbations from previous stages. However, our CNN model implementation does not consider rock and fluid changes during injection, nor pressure and stress interactions between stages. Moreover, poor performance during complex microseismic growth periods (e.g., rapid growth spurts) could be attributed to delays between the time that fluid exits the perforation points and when a microseismic event occurs. Such time delays exhibit likely complex patterns because of the complex interaction of injection history, past failure, fluid flow, and in situ stress changes at different locations. Even though we included bottom-hole engineering curves, the model cannot account for these incoherent time delays because they depend on multiple fluid and reservoir parameters unknown to the network.

It is clear that none of the three methods proposed in this work capture the complex physics behind hydraulic fracture growth. Thus, the above discussion suggests that the inclusion of physics of hydraulic fracturing into machine learning models will enhance microseismic prediction. Physics-guided neural networks have been proposed as a way to combine physics-based models with neural networks to improve generalization of physical problems (Karpatne et al., 2017). To achieve that, the proposed CNN model can be changed to use as input the output of simplified hydraulic fracturing physics-based simulations like the Coulomb failure function (CFF) (eq., 2.8) or the diffusion-based approach (section 4.5). Also, the loss function can be modified to guide the learning of the CNN to physically consistent solutions.

Chapter 5

Computer vision for fracture hit detection in low-frequency DAS

5.1 Introduction

A computer vision workflow is introduced to automate the detection of frac-hits in LFDAS. More specifically, we adapt the Oriented FAST and rotated BRIEF (ORB) object detection algorithm (Rublee et al., 2011), which locates and matches keypoints (i.e., interest features) between two or more images of the same object. Furthermore, we subtract the detected frac-hit strain signature using affine image transformations and warping. The subtraction process is applied to aid ORB in detecting multiple frac-hits on the same stage and understanding their underlying strain response. Finally, we compare the results using our proposed methodology with frac-hits detected by integrating the strain data and running an STA/LTA detection as described in section 2.4.1.

5.2 Image matching for object detection

Matching correspondences between two images is a fundamental task for many computer vision problems, including scene recognition and object detection (Lowe, 1999). Some of the most common image matching approaches depend on correlation-based template matching methods (Chen et al., 1994). While very efficient for certain image conditions (i.e., low signal-to-noise ratio and similar sizes between objects), template matching becomes computationally infeasible when object rotation, scale, and illumination strongly differ between images. An alternative to searching all image locations for correlations is to look for distinctive landmarks in both images that are at least partially invariant to noise, image transformations, and changes in illumination.

The use of ”robust” landmarks or *keypoints* for object recognition has been widely studied in the computer vision literature (Bay et al., 2006; Khan and Saleem, 2018; Leutenegger et al., 2011; Lowe, 2004; Rosten and Drummond, 2005; Rublee et al., 2011). The general workflow for identifying local keypoints for object recognition starts with a feature detection algorithm that generates interest keypoints at distinctive image areas. Next, a feature descriptor algorithm encodes the local characteristics at the neighbor pixels surrounding the interest keypoints into vectors. Finally, a matching procedure compares the descriptors from two images and matches those that maximize a similarity criterion (Awad and Hassaballah, 2016). In this section, we describe the most important algorithms in each category.

5.3 Feature detection methods

Feature detection is the process of identifying interest landmarks (e.g., edges, corners, blobs) to characterize salient aspects of digital images (Li et al., 2015). Feature detection methods aim to find highly distinctive, specific keypoints robust to image transformations, changes in illumination, and noise so that a single feature can be correctly matched between two images with high probability. In general, feature detection methods are classified as single-scale detectors and multi-scale detectors. The latter compute several representations of the same object at different scales, thereby producing scale-invariant keypoints.

5.3.1 Multi-scale detectors

Derivative of Gaussian edge detector

An edge could be defined as a set of pixels where the image intensities change abruptly. Classical edge detection tries to find areas with sharp changes in brightness by computing the image gradients in both x and y directions. However, the process of edge detection is highly sensitive to noise (Li et al., 2015). Hence, before computing the derivatives, an image $I(x, y)$ has to be smoothed by a convolution with a Gaussian kernel as

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y), \tag{5.1}$$

where $L(x, y, \sigma)$ is the smoothed image at scale σ (i.e., the scale representation of $I(x, y)$) and $G(x, y, \sigma)$ is the Gaussian kernel defined as

$$G(x, y, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2+y^2}{2\sigma^2}}. \tag{5.2}$$

Once the image has been smooth to reduce noise sensitivity, the first-order derivatives in the x and y directions can be approximated by convolving $L(x, y, \sigma)$ with a mask filter that highlights horizontal and vertical intensity changes as (Pratt, 2007)

$$L_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} * L(x, y, \sigma), \qquad L_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} * L(x, y, \sigma), \tag{5.3}$$

where the 3×3 matrices are the Sobel operators, and L_x and L_y are the derivatives of the Gaussian filter. Next, the gradient magnitude is computed as (Pratt, 2007)

$$|\Delta L| = \sqrt{L_x^2 + L_y^2}, \tag{5.4}$$

while the gradient orientation is calculated as

$$\theta = \arctan\left(L_y, L_x\right). \tag{5.5}$$

If the gradient magnitude $|\Delta L|$ is larger than a predefined threshold T, the (x, y) pixel locations are classified as an edge. The derivative of Gaussian is used by the well known Canny edge detector (Canny, 1986), which locates keypoints based on the optimization of three criteria involving the detection, localization, and uniqueness of an edge.

Laplacian of Gaussian (LoG)

Another type of edge detection relies on second-order spatial derivatives to highlight regions of rapid intensity changes. The Laplacian $\nabla^2 L(x, y, \sigma)$ of a smoothed image is given by

$$\nabla^2 L(x, y, \sigma) = L_{xx}(x, y, \sigma) + L_{yy}(x, y, \sigma), \tag{5.6}$$

which is a summation of second-order derivatives in x and y. Thus, the Laplacian of Gaussian (LoG) is the convolution of a Gaussian kernel and the Laplace operator $\Delta^2 L(x, y, \sigma)$. The 2D LoG centered at 0 has the form (Pratt, 2007)

$$\text{LoG}(x, y, \sigma) = -\frac{1}{\pi\sigma^4}\left[1 - \frac{x^2 + y^2}{2\sigma^2}\right] e^{-\left(x^2+y^2\right)/2\sigma^2}. \tag{5.7}$$

The LoG is circularly symmetric, providing keypoints with both rotation and scale invariance.

Difference of Gaussian (DoG)

Lowe, 1999 introduced the concept of difference of Gaussian (DoG) to approximate the computation of the LoG operator. The DoG can be formulated as

$$\begin{aligned} \text{DoG}(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma), \end{aligned} \tag{5.8}$$

where the first Gaussian $G(x, y, k\sigma)$ is separated in scale from the second one by a constant factor of k. In essence, equation 5.8 represents a very efficient way to approximate LoG without the need for convolution by subtracting the scale representation of the same image at different scales. The local extrema of the DoG are the basis for one of the most widely used scale-invariant keypoint detectors known as the Scale-invariant Feature Transform (SIFT) algorithm (Lowe, 2004).

5.3.2 Single-scale detectors

Harris corner detector

A corner is defined as the point where two dominant and different gradient orientations intersect at a specific image scale. Compared with edges, they are stable and unique in local image regions, making them more robust for image matching applications. Most of the earliest corner detectors relied on the computation of gradients between a local window around a candidate corner and shifted versions of the window in various directions (Moravec, 1981). For example, the Harris corner detector (Harris and Stephens, 1988) calculates the sum of squared difference $E(u,v)$ between an image patch located at (x,y) with intensity $I(x,y)$ and a patch shifted by (u,v) with intensity $I(x+u,y+v)$ as

$$E(u,v) = \sum_{x,y} w(x,y) \left[I(x+u,y+v) - I(x,y)\right]^2 , \tag{5.9}$$

where $w(x,y)$ is a window function like Gaussian (Eq. 5.2). Using a Taylor expansion, equation 5.9 can be approximated as

$$E(u,v) \approx \begin{bmatrix} u & v \end{bmatrix} \mathbf{M} \begin{bmatrix} u \\ v \end{bmatrix} , \tag{5.10}$$

where $\mathbf{M}$ represents the second-moment matrix defined as

$$\mathbf{M} = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} , \tag{5.11}$$

with I_x and I_y being the image gradients in the x and y directions respectively, and λ_1 , λ_2 representing the eigenvalues of matrix $\mathbf{M}$. Using equation 5.11,

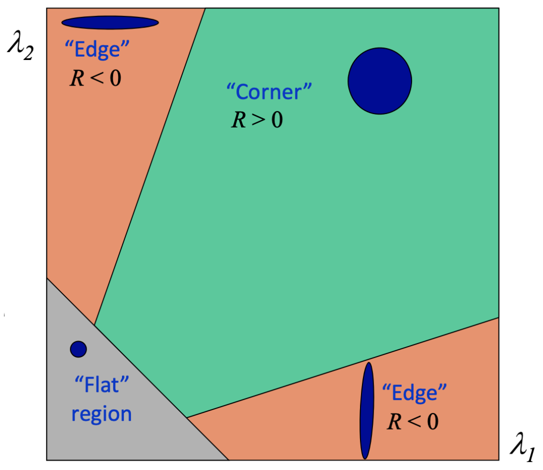


Figure 5.1: Eigenvalue space showing the edge ($R < 0$), corner ($R > 0$), and flat ($|R| \approx 0$) regions used by the Harris corner detector. Modified from Li, 2011.

Harris introduced a cornerness function (R) to determine whether a certain location in the image can be classified as an edge, a corner, or a flat region. To avoid explicit eigenvalue decomposition, the trace and the determinant of $\mathbf{M}$ can be used to approximate R as (Harris and Stephens, 1988)

$$\begin{aligned} R &= \det(\mathbf{M}) - k\,(\mathrm{trace}(\mathbf{M}))^2 \\ R &= (\lambda_1\lambda_2) - k(\lambda_1 + \lambda_2)^2, \end{aligned} \tag{5.12}$$

where $0.04 \le k \le 0.06$. If both eigenvalues are large, then R is large, implying that the candidate region is a corner. If λ_1 is small and λ_2 is large, or vice versa, the region is an edge. Lastly, if both of the eigenvalues are small the region is classified as flat (Figure 5.1).

FAST corner detector

The features from accelerated segment test (FAST) algorithm (Rosten and Drummond, 2006) is a corner detector that relies on pixel intensity comparisons between a predefined template and a candidate corner. FAST keypoints are detected by considering a circular template of 16 pixels and applying a segment test to every image pixel. For a candidate corner p, each pixel on the template

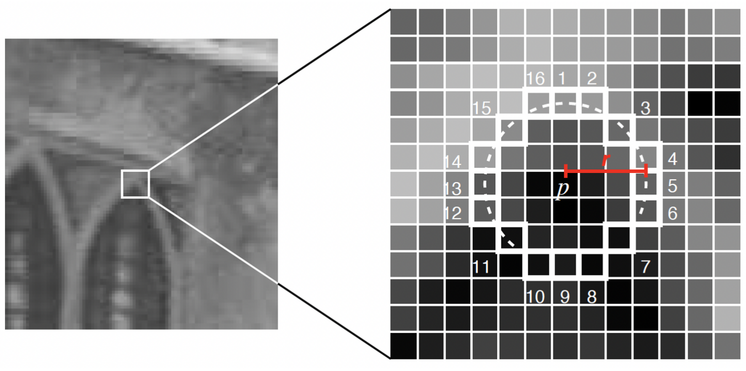


Figure 5.2: Image patch around the pixel at the center of a candidate corner p. The highlighted pixels lying on a circular template with a radius $r = 3.4$ are used in the corner detection. The dashed arc ($n = 12$) indicates the contiguous pixels used to classify p as a corner. Modified from Rosten and Drummond, 2006.

$x \in \{1 \dots 16\}$ has three states $S_{p\to x}$ relative to p defined as

$$S_{p\to x} = \begin{cases} d, & I_{p\to x} \leq I_p - t \qquad \text{(darker)} \\ s, & I_p - t < I_{p\to x} < I_p + t \qquad \text{(similar)} \\ b, & I_p + t \leq I_{p\to x} \qquad \text{(brighter)} \end{cases}, \tag{5.13}$$

where $I_{p\to x}$ is the intensity of the pixel x, I_p is the intensity of the candidate corner, and t is a threshold intensity value. The candidate p is classified as a corner if there exists a set of n contiguous pixels in the circle that are all brighter than the intensity of $p + t$ (i.e., all n pixels have a state $S_{p\to x} = b$), or all darker than $I_p - t$ (i.e., all n pixels have a state $S_{p\to x} = d$) (Figure 5.2). Furthermore, Rosten and Drummond, 2006 adopt a a decision tree classifier to optimize the order in which pixel comparisons in equation 5.13 are performed.

5.4 Feature descriptor methods

Once a set of interest features like edges and corners have been detected in an image, the visual appearance of the pixels surrounding the keypoints has to be encoded into a suitable descriptor for matching between two images (Awad and Hassaballah, 2016). Feature descriptor methods create vectors aligned with the orientation θ and proportional to the scale s of the keypoints, which describe the neighbor pixels' content or image structure. In the last years, feature descriptor algorithms that rely on pairwise pixel comparisons to generate binary descriptors have proven to be highly efficient in computation and storage required for real-time image matching applications (Leutenegger et al., 2011).

5.4.1 Binary descriptors

Binary Robust Independent Elementary Features (BRIEF)

The Binary Robust Independent Elementary Features (BRIEF) (Calonder et al., 2010) builds short descriptors by comparing pair-wise pixel intensities over an image patch defined around keypoints detected using the FAST corner detector (Sec. 5.3.2). BRIEF starts by applying Gaussian smoothing on an image patch centered around the FAST keypoints to reduce the noise sensitivity and increase the stability and repeatability of the descriptors. The algorithm then selects random pairs of pixels within the patch to compare their intensities by applying a logical binary test. The test τ on patch $\mathbf{p}$ of size $S \times S$ is defined as

$$\tau(\mathbf{p}; x, y) = \begin{cases} 1 & \text{if } \mathbf{p}(x) < \mathbf{p}(y) \\ 0 & \text{if } \mathbf{p}(x) \geq \mathbf{p}(y) \end{cases}, \tag{5.14}$$

where $\mathbf{p}(x)$ and $\mathbf{p}(y)$ are the intensities of pixels at locations x and y respectively. Therefore, the n_d-dimensional bit-string BRIEF descriptor on a particular patch $f_{n_d}(\mathbf{p})$ is

$$f_{n_d}(\mathbf{p}) = \sum_{i=1}^{n_d} 2^{i-1} \tau(\mathbf{p}; x_i, y_i). \tag{5.15}$$

The matching performance of BRIEF binary strings increases with uncorrelated tests n_d because in that way each one contributes to the result. Thus, to make the test locations (x_i, y_i) more indiscriminate, pixel pair locations are drawn from an isotropic Gaussian distribution centered at 0 and with standard deviation $\sigma = \frac{S}{5}$.

5.5 Feature matching methods

The last step in image matching for object recognition is to establish correspondences between two images. Feature matching methods use a distance criterion to measure the similarity between descriptors from keypoints computed on different images.

5.5.1 L_p distance matching

For a group of K keypoints detected on an image p, a corresponding set of descriptors can be defined as

$$\Phi(p) = (f_{dk} | k = 1, 2, \ldots K), \tag{5.16}$$

where f_{dk} is the kth feature descriptor. The goal is to find the best correspondence with the descriptors from another image q by calculating the L_p distance between each keypoint in p and q. The latter is defined as

$$d(p, q) = \left(\sum_{i=1}^{K} (|\Phi(p) - \Phi(q)|)^p \right)^{\frac{1}{p}}. \tag{5.17}$$

The L_2 or Euclidean distance (i.e., $p = 2$) and the L_1 norms are commonly used. Once the distance metric is calculated for all keypoints in both images, the pairs from p and q with the smallest $d(p, q)$ are declared as a match.

5.5.2 Hamming distance matching

The Lp distance cannot be computed with binary descriptors like the ones output by BRIEF. In this case, the Hamming distance is usually applied as a fast and efficient alternative to measure the similarity between descriptors (Leutenegger et al., 2011). The Hamming distance $d(u, v)$ of two binary strings

u and v equals the number of positions where the strings have different values. For instance, Calonder et al., 2010 calculates the Hamming distance using a bit-wise XOR operation followed by a bit count.

5.6 Oriented FAST and rotated BRIEF (ORB)

Oriented FAST and rotated BRIEF (ORB) is a local feature detector and descriptor algorithm developed by Rublee et al., 2011. ORB uses the FAST-9 (circular template with radius of 9 pixels) corner feature detector (section 5.3.2) to locate interest keypoints on the input image. However, FAST does not provide a measure of the cornerness of each keypoint, which can be useful to remove low-discriminate points such as those located in the border of the image. To solve this problem, ORB sorts the keypoints by their Harris cornerness measure (Eq. 5.12) and removes those below the top N sorted keypoints. In addition, FAST does not produce multi-scale features. Thus, to achieve scale-invariance, ORB employs scale pyramids to construct the scale space of an image (Lindeberg, 1998). The latter are computed by convolving the original image with a Gaussian filter (Eq. 5.2), followed by a down-sampling operation. The process repeats iteratively, with each level of the pyramid using the previous level as input. In a nutshell, ORB produces FAST keypoints, filtered by their Harris cornerness, at each level of the scale pyramid.

The remaining keypoints should be robust to most possible image transformations. ORB achieves rotation-invariance by computing the intensity centroid on a patch of size 31 pixels $\times$ 31 pixels centered at each keypoint location. The intensity centroid assumes that the corner's centroid C is offset from its center O, such that the vector $\vec{OC}$ can be used to assign an orientation to the keypoints (Rublee et al., 2011). The centroid C is defined as

$$C = \left(\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}} \right), \tag{5.18}$$

where m_{ij} are the moments of the patch, computed as

$$m_{ij} = \sum_{x,y} x^i y^j I\left(x, y\right), \tag{5.19}$$

with $I\left(x, y\right)$ being the intensity of each pixel within the patch. The orientation of the patch θ can be found as

$$\theta = \text{atan2}(m_{01}, m_{10}), \tag{5.20}$$

where atan2 is the quadrant-aware version of arctan. Next, ORB calculates the binary descriptor of the patch using a rotated version of BRIEF-32 (i.e., 32 bytes). For a feature set of n_d binary tests at locations (x_i, y_i) the algorithm defines the $2 \times n_d$ matrix

$$\mathbf{S} = \begin{pmatrix} \mathbf{x}_1 \ldots, \mathbf{x}_{n_d} \\ \mathbf{y}_1, \ldots, \mathbf{y}_{n_d} \end{pmatrix}. \tag{5.21}$$

Using the patch orientation θ computed from equation 5.20, ORB constructs a steered version $\mathbf{S}_\theta$ of $\mathbf{S}$

$$\mathbf{S}_\theta = \mathbf{R}_\theta \mathbf{S}, \tag{5.22}$$

where $\mathbf{R}_\theta$ is the rotation matrix defined as

$$\mathbf{R}_\theta = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}. \tag{5.23}$$

The binary test locations on $\mathbf{S}_\theta$ are used as input to equation 5.15 to produce a rotation-invariant BRIEF descriptor. Lastly, ORB compares the similarity between a set of descriptors coming from 2 images using the Hamming distance (Section 5.5.2) followed by a brute-force matching algorithm. The latter returns only those matches with value (i, j) such that ith descriptor in the first image has jth descriptor in the second image as the best match and vice-versa.

5.7 Image transformations

Matching keypoint pairs can be used to find the optimal linear transformation that relates two images. These transformations are implemented in a wide range of computer vision applications like image registration and image warping (Solem, 2012). In this work, we apply warping to modify the LFDAS strain response on a predefined template to best match a target frac-hit strain signal.

5.7.1 Affine transformations

An affine transformation is a particular class of geometric image transformation that preserves collinearity (i.e., lines remain lines) and distance ratios (i.e., the midpoint on a line remains the same). Consider a point $p = (x, y)^T$ in an image I that relates to a point $p' = (x', y')^T$ on a distorted version of that image I'. Affine transformations are the all transforms that can be written as

$$p' = \mathbf{M}p, \tag{5.24}$$

where $\mathbf{M}$ is composed of linear transformations and translations. Equation 5.24 can be expressed using homogeneous coordinates, which allows us to add a dimension to the working space to represent all possible affine transformations as a multiplication of matrices

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}, \tag{5.25}$$

where a, b, d, e are scalars combining scaling in x (s_x), scaling in y (s_y), rotation angle (θ), shearing in x (h_x), shearing in y (h_y), c is the translation component in x (t_x), and f is the translation component in y (t_y).

The matrix $\mathbf{M}$ can be expressed by six independent parameters or degrees of freedom as h_y can be performed by cascading a 90° rotation, a shearing in x h_x, and again another $-90°$ rotation. Hence, we can solve for its elements using a set of at least $N \geq 3$ corresponding pairs of points between the original image I and the deformed image I'. For i pairs of correspondent points and using homogeneous coordinates, equation 5.25 can be expressed as (Solem, 2012)

$$\begin{pmatrix} x_1 & y_1 & 0 & 0 & 1 & 0 \\ 0 & 0 & x_1 & y_1 & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_i & y_i & 0 & 0 & 1 & 0 \\ 0 & 0 & x_i & y_i & 0 & 1 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \\ d \\ e \\ f \end{pmatrix} = \begin{pmatrix} x'_1 \\ y'_1 \\ \vdots \\ x'_i \\ y'_i \end{pmatrix}. \tag{5.26}$$

For $N > 3$ matching pairs, we define the over-determined linear system of equations as

$$\mathbf{A}\mathbf{x} = \mathbf{b}. \tag{5.27}$$

The least-squares solution for the parameters $\mathbf{x}$ can be determined by solving the corresponding normal equations

$$\mathbf{x} = \left[\mathbf{A}^T\mathbf{A}\right]^{-1}\mathbf{A}^T\mathbf{b}, \tag{5.28}$$

which minimizes the sum of the squared differences between the keypoints locations in I' and I.

5.7.2 Warping

Once the elements of the transformation matrix $\mathbf{M}$ are found, warping can be applied to send each pixel in the original image to its corresponding location in the new image.

Forward mapping

Image warping can be achieved using *forward mapping*, which involves iterating over each pixel in the input image, computing the new coordinates using equation 5.24, and copying its value to the new image (Efford, 2000). However,

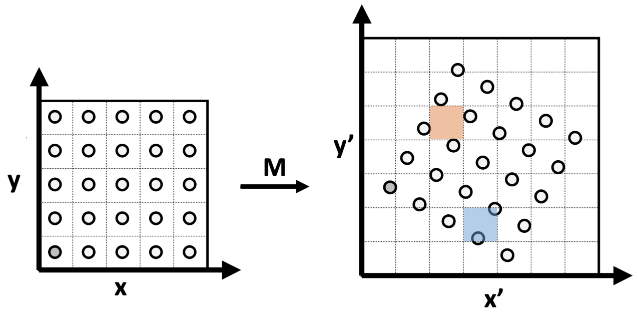


Figure 5.3: Limitations of forward mapping for image warping. A source image with pixel coordinates (x, y) (left) is transformed by a rotation of $-30°$ to produce an output image with coordinates (x', y'). No source pixel was mapped into the orange coordinate, while 2 pixels were mapped into the blue coordinate.

there are several limitations associated with the forward mapping model. For example, consider a pixel $p(x, y)$ located at (50,0) in an input image, which will be transformed by the rotation matrix $\mathbf{M}$ defined as

$$\mathbf{M} = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}, \tag{5.29}$$

with $\theta = 35°$. Using equation 5.24, the location of the pixel in the new image is computed as

$$\begin{aligned} x' &= x\cos\theta - y\sin\theta = 50\cos(35°) = 40.96 \\ y' &= x\sin\theta + y\cos\theta = 50\sin(35°) = 28.68 \end{aligned} . \tag{5.30}$$

Note that output pixel locations $p(x', y')$ are not integers, and therefore cannot be placed in the output image. Figure 5.3 illustrates two other problems caused by forward mapping. For example, several source pixels can map to the same destination pixel, or even worse; some destination pixels may not be covered at all.

Inverse mapping

To guarantee that a single value is generated at every pixel of the output image, we must analyze each transformed pixel and determine the input image's position from where each output pixel must be sampled (Efford, 2000). The latter is called *inverse mapping*, as it relies on the inverse of the transformation matrix **M** in equation 5.24. However, this approach still has the same limitations as forward mapping (i.e., non-integer pixel locations, un-mapped pixels, and over-mapped pixels), thus requiring the application of 2D interpolation strategies.

The most straightforward interpolation in two dimensions is a zero-order interpolation approach like nearest neighbors (Efford, 2000). In this case, a pixel with float coordinates after the inverse mapping is assigned the value of the nearest integer pixel. Although very efficient, this approach degrades the transformed image's appearance as it makes it look very "blocky." A first-order or *bilinear interpolation* can be used to improve results by considering the 4 pixels surrounding the calculated point in the input image. For a calculated point (x, y) in the input image, the first-order interpolation function can be written as (Press et al., 2007)

$$f(x,y) = \sum_{i=0}^{1}\sum_{j=0}^{1} c_{ij}x^i y^j, \tag{5.31}$$

where c_{ij} are obtained by solving a linear system of equations involving $f(x, y)$ and its derivative evaluated at the 4 surrounding pixels. Similarly, a third-order *bicubic interpolation* can be used to obtain even smoother results. In such case, the interpolation uses the 16 pixels surrounding the calculated point (x, y) and solves the third-order function (Press et al., 2007)

$$f(x,y) = \sum_{i=0}^{3}\sum_{j=0}^{3} c_{ij}x^i y^j, , \tag{5.32}$$

which requires knowledge of the first derivative of $f(x, y)$ at the 16 pixels surrounding the calculated point.

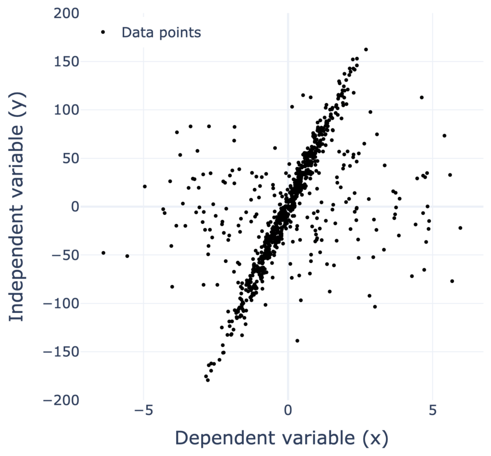
5.7.3 Random sample consensus (RANSAC)

Equation 5.26 assumes a set of correspondent points between two images. In reality, this assumption is not valid because some points are mismatched, leading to outliers in the least-squares solution. The Random Sample Consensus (RANSAC) robust estimation algorithm can be applied to reduce the effect of outliers (Fischler and Bolles, 1981). RANSAC randomly selects a sample from all matching pairs N with the minimum number of points required to determine the model parameters. Then, the least-squares solution (Eq. 5.28) is computed using only the sample subset. The number of points from all N within a tolerance distance ϵ from the fitted model are classified as inliers. If the number of inliers exceeds a τ threshold, the model parameters are recomputed using all the inliers. Otherwise, the random selection and fitting process iteratively repeats. Figure 5.4 compares the linear regression results on a synthetic example with outliers using the RANSAC algorithm and classical least-squares. See how RANSAC performs better than least-squares by only considering inlier data points to fit the model.

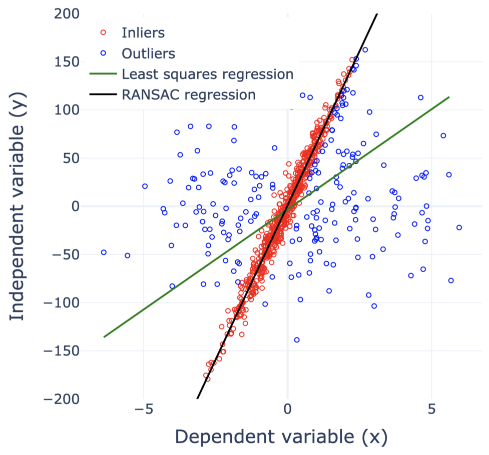
5.8 Data preparation

We use low-frequency DAS (LFDAS) data from a multi-well completion pad in Western Canada. Four horizontal wells were stimulated, and one of them was equipped with a fiber-optic cable used for LFDAS cross-well communication measurements (Section 2.4.2). The DAS data were acquired simultaneously with the fracturing operation and sampled at 20kHz with a spatial sampling of 1.02 m and a gauge length L_G =7.14 m (see equation 2.33).

The operator down-sampled the DAS data from 20 kHz to 1 Hz using an anti-aliasing filter followed by a differentiation in the measured optical phase to obtain a linear-scaled strain rate. We then separated the LFDAS data in



(a)



(b)

Figure 5.4: A synthetic dataset heavily affected by outliers (a). The RANSAC and least-squares regression comparison (b). RANSAC uses the inliers (red) to fit the model and discards the outliers (blue), while classical least-squares considers all points.

a stage-by-stage manner— that is, data within the start and end of injection on each stage—and only considered the depth containing the frac-hits (≈ 350 meters along the measured depth of the cable for most of the stages). Next, we clipped the LFDAS data between -1 radians to +1 radians to aid in visualizing the frac-hits. Clipped data were then converted to red-green-blue (RGB) images used as input to the proposed computer vision workflow. Figure 5.5 shows an LFDAS RGB image for one of the stages in the treatment. Each pixel represents a one-second time and one-meter depth such that the frac-hit location in the images can be directly interpreted on the original LFDAS data domain. Thus images are used as input to locate keypoints at different pixel locations. However, the blue (compression), red (extension), and white (no strain change) color intensities have an associated physical meaning in the strain-rate domain. From now on, figures without a color bar are RGB images defined by a pixel width and height. In contrast, figures with a color bar represent strain rate data measured in radians and with physical dimensions in seconds $\times$ meters.

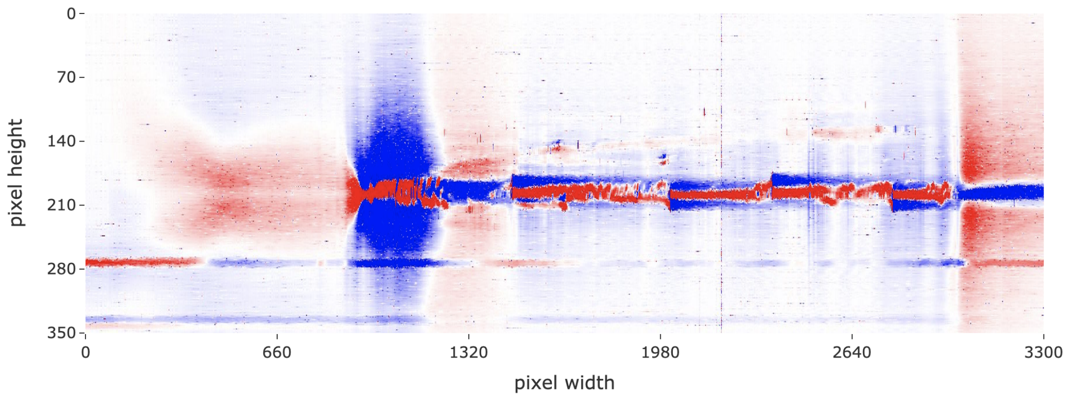


Figure 5.5: RGB image showing the LFDAS recorded at one stage of the treatment. Image dimensions are (350 pixels $\times$ 3300 pixels). Axis ratios have been distorted.

5.9 Methodology

5.9.1 Fracture hit detection

The proposed computer vision workflow is divided into a detection and subtraction procedure. We use the ORB algorithm implementation from OpenCV (Bradski, 2000) which implements the FAST keypoint detection algorithm (Sec. 5.3.2) and the BRIEF descriptor builder methodology (Sec. 5.4.1) [2]. The proposed detection steps are as follows:

1. Select a frac-hit template. A 120 $\times$ 120 pixel image around the frac-hit shown in figure 5.5 was used as a template.

2. Select the LFDAS data from any stage in the 3 monitored wells to locate frac-hits.

3. Smooth the template and stage images. We applied bilateral filtering to smooth the images while preserving its corners and edges (Tomasi and Manduchi, 1998). This technique applies Gaussian smoothing (Eq.5.1) in the spatial and pixel intensity domains. A Gaussian kernel of 9 pixels was used with a standard deviation of $\sigma = 150$ in both space and color domains. The same filter parameters were applied on both images.

[2] OpenCV, an open-source computer vision library. For more details on ORB visit https://docs.opencv.org/3.4/db/d95/classcv_1_1ORB.html.

4. Run the ORB algorithm (keypoint detection, feature descriptor, feature matching) on the smoothed template and smoothed stage images. The ORB parameters applied to both images are:
    - Maximum number of features (keypoints) = 5000.
    - Scale factor = 1.1. Each level of the pyramid is down-sampled by a factor of 1.1.
    - Number of pyramid levels = 8.
    - FAST threshold $t = 20$ (Eq. 5.13).
    - Size of the BRIEF patch $S = 31 \times 31$ (Eq. 5.14).
5. Remove outlier matches on the stage image. We removed all matches with a radial distance larger than 40 pixels from the keypoint location in the stage associated with the lowest Hamming distance.
6. Average the remaining matched locations on the stage, and all the matches on the template, to find the top and bottom left pixel coordinates of the square containing the detected frac-hit. The square has the same dimensions as the template (i.e., 120×120 pixels).

5.9.2 Fracture hit subtraction

We use the ORB algorithm to detect corresponding matching feature pairs between the detected frac-hit and the template. The subtraction procedure starts after a frac-hit is detected. Our proposed subtraction workflow is

1. Run the ORB algorithm on the stage frac-hit and the template (both are 120×120 pixels). Unlike for the detection process, images are not smoothed before applying ORB. The latter allows the matching of more corresponding feature pairs, providing more data to the least-squares inversion (Equation 5.28). We use the same ORB parameters of the detection scheme except for the maximum number of features (= 500) and the pyramid scale factor (= 1.2).

2. Find the least-squares solution of matrix $\mathbf{M}$ (Eq. 5.24). In this case, we do not apply a spatial constraint to remove matches, but rather allow RANSAC to select the inliers for the least-squares fit.

3. Warp the template according to the inversion results using equation 5.24. The warping is applied directly to the LFDAS strain data and not in the images used to detect the ORB matching feature pairs. This is because we aim to subtract the stage frac-hit strain response, which cannot be done with pixel-based image subtraction. To deal with edge effects introduced by size changes after warping, we applied the transformation on a larger template of size 280×280 pixels around the frac-hit in figure 5.5.

4. Subtract the stage frac-hit and the warped template LFDAS data. The warped template can be directly subtracted from the stage frac-hit because both have the same size (i.e., 120×120 pixels), and their values are clipped between -1 radians and +1 radians.

5. Convert the subtracted LFDAS strain signal back to an RGB image and place it on the same square coordinates outputted on the detection workflow.

After running the frac-hit detection and subtraction steps on one fracture hit, the entire workflow can be repeated to locate additional frac-hits in the stage of interest.

5.10 Results

Figure 5.6a shows the template image (I_t) extracted around the frac-hit displayed on figure 5.5, which is used for matching frac-hits at other stages. Likewise, figure 5.7a shows the stage image (I_s) used as input to the detection workflow. We applied bilateral filtering on the template I_t and stage I_s to limit the number of non-relevant keypoints detected (compare figures 5.6b and 5.6c). Smoothing with a bilateral filter reduces the number of keypoints in the template I_t and clusters them in two distinctive groups. The 23 keypoints in figure 5.6c are located in the white region between the extensional frac-hit and the compressional stress shadow, a highly distinctive region seen on most frac-hits across the treatment stages.

The stage keypoints are detected with the same ORB parameters used on the template I_t (Figure 5.7b). Most keypoints appear in areas with large intensity gradients (e.g., the frac-hit, the stress shadow, and strain pulses). Each keypoint on the template I_t and stage I_s has an associated descriptor computed using a rotation-aware version of BRIEF (see equations 5.15 and 5.22). We matched the 23 keypoints in the template with the best keypoints in the stage using the brute-force matching algorithm (Figure 5.8a). To deal with outliers, we removed all matches with a radial distance larger than 40 pixels from the keypoint in I_s with the lowest Hamming metric (Figure 5.8b). The 40-pixel distance criterion was chosen based on the maximum separation between the lower and upper clusters of keypoints in figure 5.6c. Using the average location of the remaining matches on I_s, we located the frac-hit and extracted a square image patch (I_p) the same size of the template around its location for further use in the subtraction process (Figure 5.9).

We subtract a warped version of the template (I_w) from I_p to remove the image patch's frac-hit strain signature. For that, the subtraction steps were applied on the template I_t and frac-hit patch I_p for finding matching feature pairs to invert for the elements of an affine transformation relating both images. However, instead of limiting the number of keypoints as done in the frac-hit detection, we allowed many more in both images to improve the inversion results. Figure 5.10 shows the matches detected on the frac-hit patch I_p. We investigate the subtraction results using different degrees of freedom (DOF) on equation 5.26; three cases were analyzed

- Full affine transformation (6 DOF): That is, s_x, s_y, θ, h_x, t_x, and t_y.
- Limited affine transformation (4 DOF): That is, isotropic scaling in x and y s_{xy}, θ, t_x, and t_y.
- Isometry (3 DOF): That is, isotropic scaling in x and y s_{xy}, t_x, and t_y.

Each affine transformation was computed using the same matched points and the RANSAC-based robust least-squares algorithm. A warped template I_w was calculated from the three affine transformation and subtracted from I_p. To deal with unmapped points resulting from changes in shape after warping, we warped a larger frac-hit template (I_l) (Figure 5.11) using the same affine transformations computed using the original template I_t (Fig. 5.6a). The last column of figure 5.12 shows the subtraction results based on the aforementioned degrees of freedom of the matrix $\mathbf{M}$.

Our goal is to remove the largest amount of fracture hit signal from the frac-hit patch I_p. The cumulative strain of the squared of the subtraction result was compared between the three cases analyzed (Table 5.1). An affine transformation matrix with 6 DOF minimizes the squared of the subtraction, effectively removing the extensional frac-hit and compressional stress shadow signals. We conclude that allowing for a rotation θ and a shearing component h_x, together with the two scalings and translation factors, maximizes the removal of the frac-hit signal.

The subtracted data I_d was converted back to an RGB image and placed at the same location of the image patch I_p. Next, the detection steps were repeated with the subtracted frac-hit (Figure 5.13). After applying the spatial constraint to remove outlier matches (Figure 5.13c), our detection methodology using object detection and image warping locates the second frac-hit (Figure 5.13d). The final output are the coordinates of both frac-hits defined by the center of the yellow and cyan squares (i.e., I_p and I_{p_2} respectively) in figure 5.13d. The first frac-hit was located at (2633 s, 104 m), and the second frac-hit at (3110 s, 95 m).

Table 5.1: Results of the frac-hit subtraction using various DOF to solve for the elements of the affine transformation relating the template I_t and the frac-hit patch I_p. I_w is the warped template.

DOF in equation 5.26	$(I_p - I_w)^2$
6	1291
4	2244
3	3072

5.10.1 Strain rate integration plus STA/LTA detection

We compare the frac-hit detection results obtained using our proposed computer vision workflow with those using a short-time-average over long-term-average (STA/LTA) detector (Eq. 2.21) on the integrated strain rate data. The LFDAS integrated in time can be used to obtain the cumulative strain variation (Jin and Roy, 2017). Positive values indicate expansion with respect to the start of the stage, while negative values indicate contraction. Frac-hits are then commonly identified as the maximum strain change with depth. Furthermore, LFDAS data can be integrated in space to get displacement (Jin and Roy, 2017). In this case, positive values express movement towards the toe of the well, while negative values imply movement towards the heel. Figure 5.14 shows the integrated LFDAS strain data in time and depth for the stage shown in figure 5.7a . The cumulative amplitude in time displays large positive values at the frac-hits depth product of crack-tip propagation and tensile fracture opening. However, from the right panel in figure 5.14 it is not possible to differentiate that frac-hits are at different depths. Moreover, fracture closing signals from previous stages (i.e., 180 m) and rapid extensional-compressional signals (e.g., 225, 260, 320 m) have a strong influence on the cumulative strain measurements in time. The bottom panel of figure 5.14 shows a strong negative response associated with the earliest frac-hit stress shadow. Again, this curve is highly influenced by signals from previous stages.

We applied the STA/LTA detector to the integrated LFDAS curves with the aim of identifying the depth and time of the two frac-hits. The STA/LTA parameters used are:

- Integrated time: STA = 5 m, LTA = 80 m, threshold $\Lambda = 8$.
- Integrated depth: STA = 20 s, LTA = 400 s, threshold $\Lambda = 8$.

Figure 5.15 shows the detection results after running the STA/LTA on both domains. The algorithm detected three frac-hits in the integrated space, but only one in the integrated time. The stress shadows on both frac-hits trigger a high STA/LTA response, thus delivering accurate time detections. However, the closing signal from the previous stage triggers a false positive around 1400

s. On the other hand, the STA/LTA in the right panel of figure 5.15 only detects one frac-hit. The latter could de due to the short separation in space and time of both frac-hits, which cannot be identified by a simple integration of the strain.

We analyzed the proposed computer vision workflow's detection capabilities compared to the STA/LTA detector at other stages of the hydraulic fracturing treatment. Figure 5.16 shows the results at another stage with four fracture hits. Our algorithm located three of the frac-hits present in the stage. The fourth frac-hit—located at the lower part of the magenta square in figure 5.16a)—does not show a clear view of the white region between the extensional signal and the compressional stress shadow; thereby limiting the matching of keypoints in figure 5.6c. For the STA/LTA 5.16b, the algorithm identifies the time of the four frac-hits but fails at locating their depth. We conclude that integrating the strain in time does not provide an accurate frac-hit detection when multiple frac-hits appear within a similar spatiotemporal location in the stage.

Figure 5.17 shows the results at another stage of the treatment containing a single frac-hit. The detection results using our computer vision method are comparable to using an STA/LTA in the integrated LFDAS strain in space and time. Hence, both methods achieve a high detection accuracy for stages with only one frac-hit and little to no strain signal from previous stages.

5.11 Conclusion

The results presented in this chapter prove the effectiveness of using a computer vision workflow based on image matching for the detection and subtraction of multiple frac-hits in LFDAS data. Our results are comparable with a detection done by integrating the strain data and running an STA/LTA in stages with single frac-hits. However, we found that when multiple frac-hits are closely separated in space and time, our proposed methodology achieves better detection results, especially in locating the frac-hits depth.

When using ORB for frac-hits detection, applying bilateral smoothing to the input images results in fewer but highly distinctive keypoints that improve the detection performance. Conversely, to find the appropriate affine transformation that relates a template frac-hit with a stage frac-hit, images should not be bilateral-smoothed, to allow for more matching pairs to use on the inversion. The computational cost of matching more descriptors is compensated by the fact that the template and image patches are only 120×120 pixels in size, while an LFDAS stage can be up to 350×7000 pixels.

Beyond the introduction of a computer vision algorithm for frac-hit detection, our approach also permits removing the frac-hit via affine transformations and warping applied directly to the strain data instead of the image. This has the advantage that it reveals strain information obscured for instance by stress shadows. The latter can be seen in the lower part of the second frac-hit in figure 5.13d, which is hidden under a compressional signal (i.e., negative phase shown in blue). Edge effects introduced by warping are corrected using data from a larger template undergoing the same warping transformation as the 120×120 pixels template.

We calculated three affine transformations with different degrees of freedom using correspondent matching feature pairs between a fracture hit template example and a stage frac-hit. The warping results prove that a full affine transformation with six DOF minimizes the squared difference of the subtraction between the detected frac-hit and a warped template, thereby removing the most strain signal.

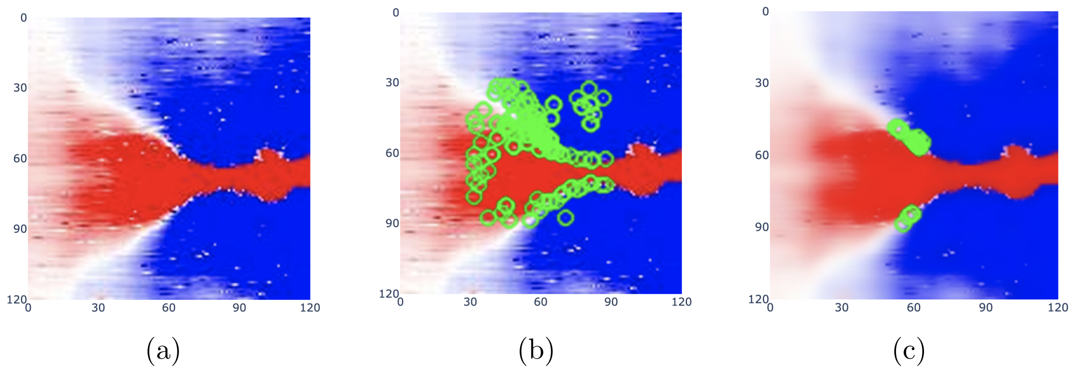


Figure 5.6: The frac-hit template (a), 132 ORB keypoints (green circles) detected on the template (b), 23 ORB keypoints detected on the smoothed template after applying a bilateral filter (c). The x-axis is pixels width and y-axis is pixel height. All images are 120 × 120 pixels.

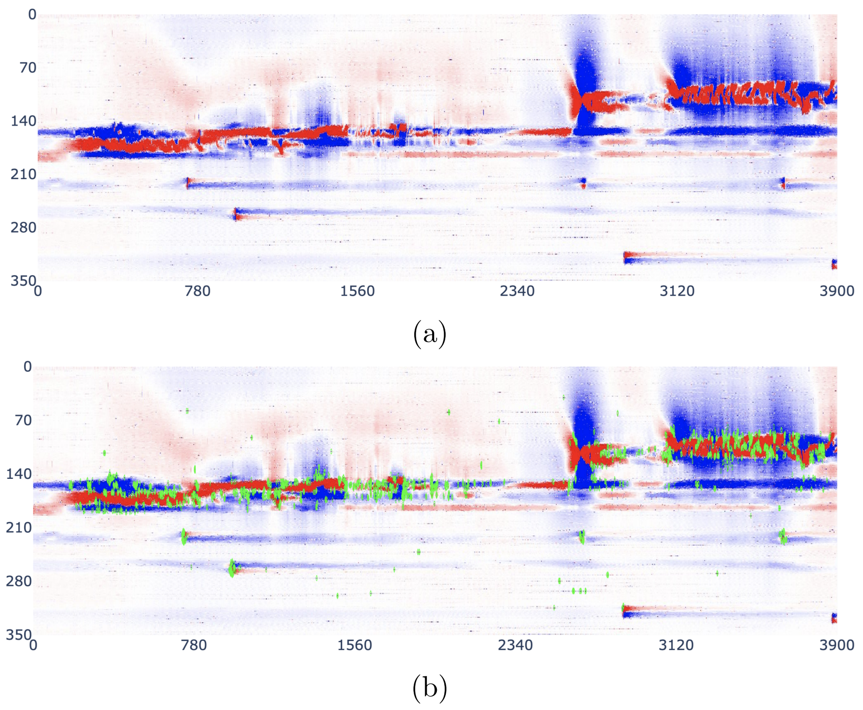
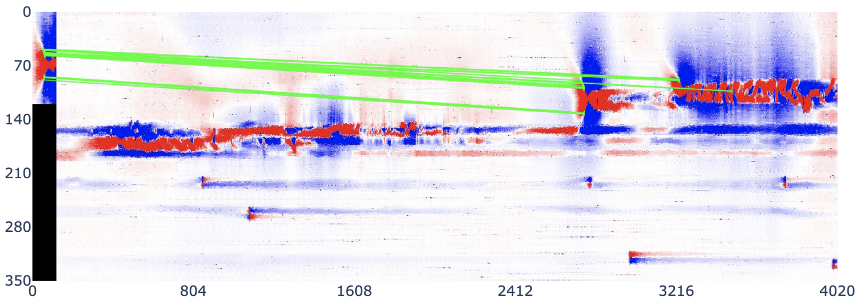
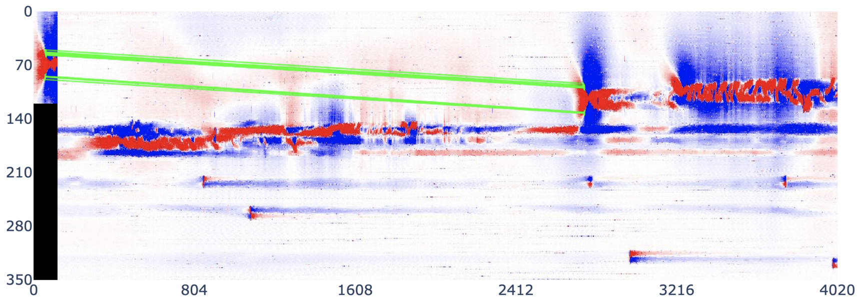


Figure 5.7: The LFDAS RGB image from one stage used as input to the detection workflow (a). 4014 keypoints detected (green) using the ORB algorithm after bilateral smoothing of the image (b).



(a)



(b)

Figure 5.8: Matched descriptors between the template (top left corner) and the stage appear as green lines connecting the 23 keypoints in figure 5.6c with the best 23 keypoints in 5.7b (a). Fifteen matches left after applying a spatial constraint to remove the outlier matches further away from the lowest Hamming distance match (green lines) (b). x-axis is the pixel width of the stage plus pixel width of the template.

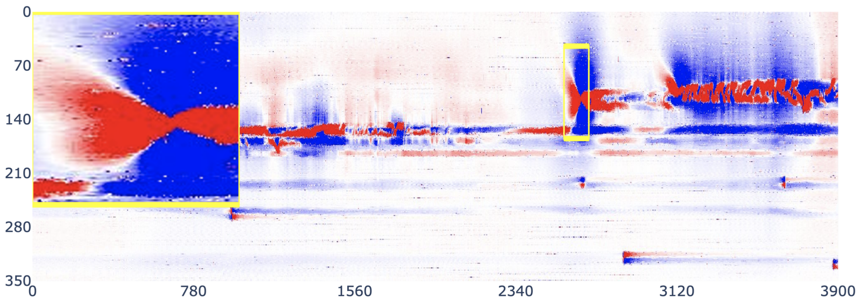


Figure 5.9: Stage with frac-hit detected (yellow) inside a bounding box patch of size 120×120. Inset shows the frac-hit.

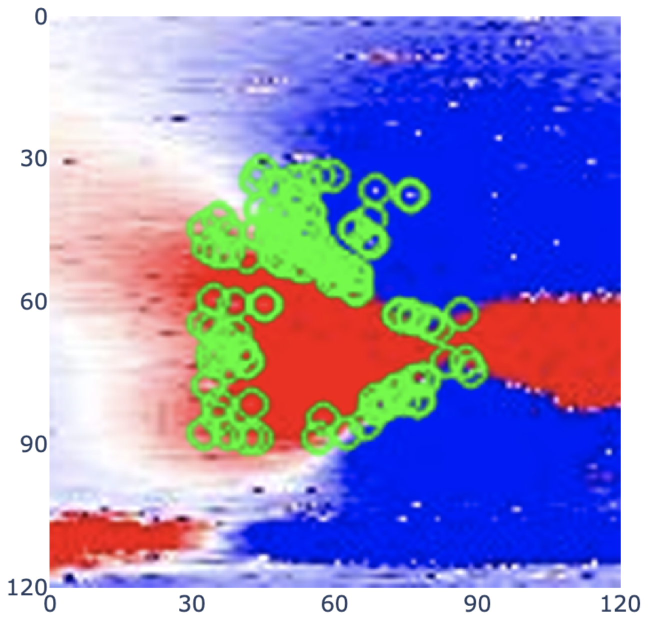


Figure 5.10: 117 keypoints (green circles) detected in the image patch containing the detected frac-hit. The 132 keypoints from image 5.6b are matched to these keypoints.

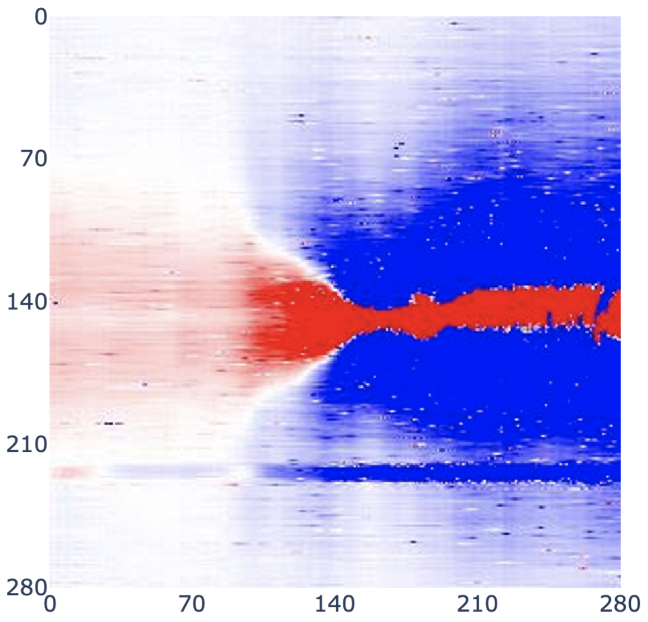
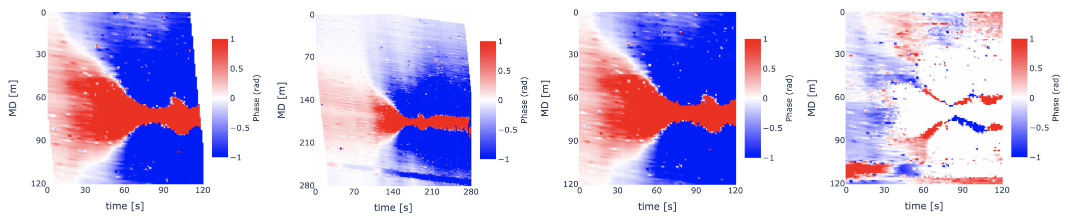
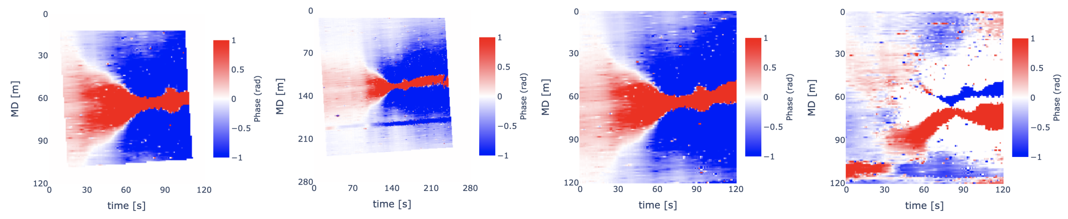


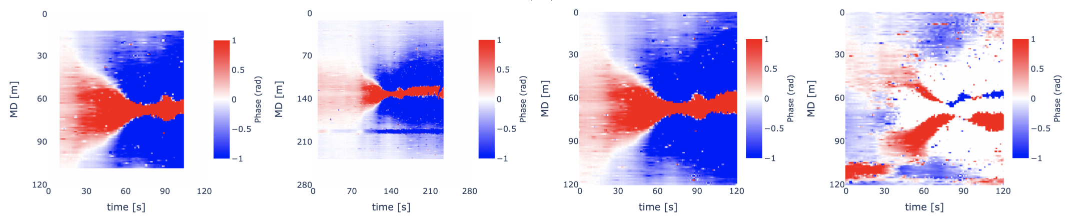
Figure 5.11: Large frac-hit template extracted around the frac-hit on image 5.5. The large template dimensions are 280×280 pixels, which exceed by more than double the original dimensions of the template in figure 5.6a.



(a)

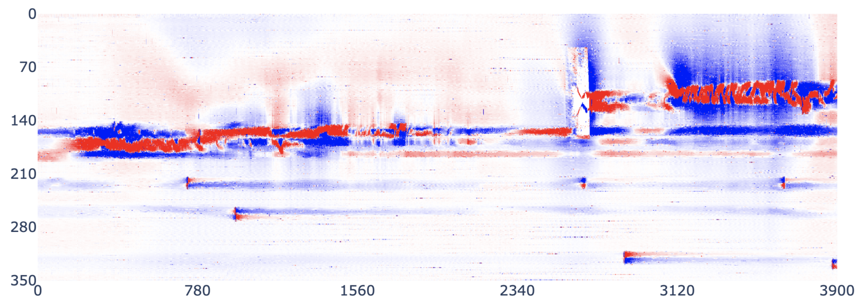


(b)

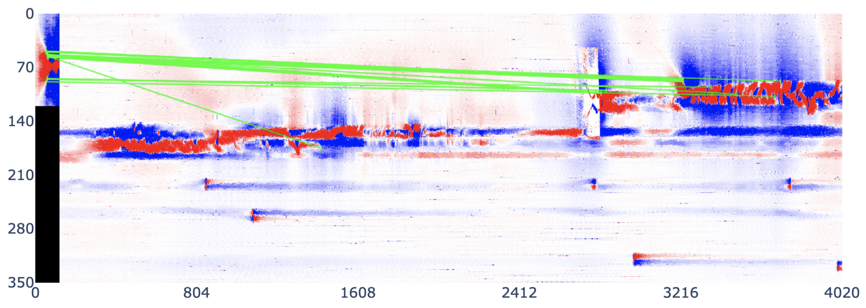


(c)

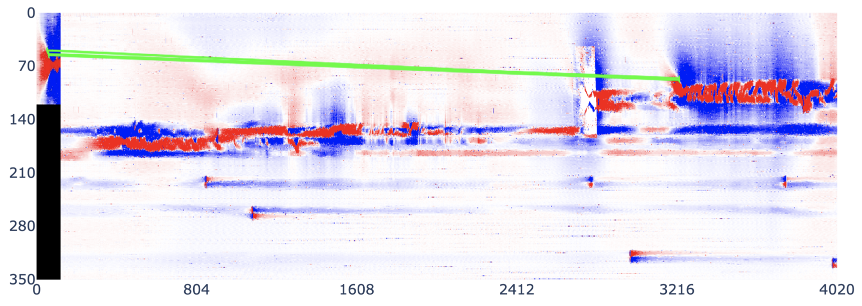
Figure 5.12: LFDAS strain data subtraction between the frac-hit patch (figure 5.9) and a warped version of the frac-hit template on figure 5.10 computed using various DOF on the least-squares solution of equation 5.26. 6 DOF (a), 4 DOF (b), and 3 DOF (c). From left to right in all rows, the warped template, the warped large template, the warped template with missing points filled, and the subtraction between the frac-hit patch (figure 5.9) and the warped template with missing points filled. For all figures, the x-axis is time (s), and the y-axis is depth (m). All the values are clipped between -1 and +1.



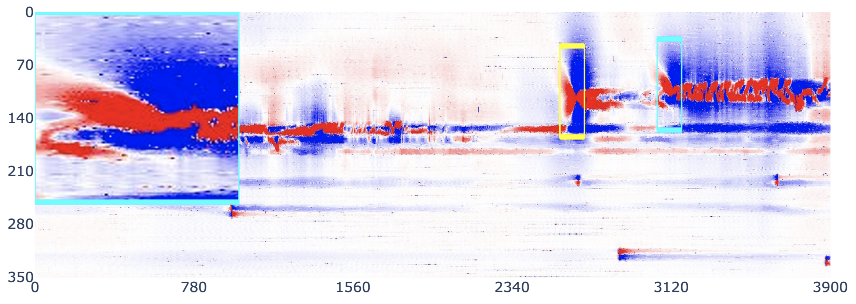
(a)



(b)



(c)



(d)

Figure 5.13: The stage image after subtracting one frac-hit (a). 23 matched keypoints (green) between the template (top left corner) and the stage image after removing the first frac-hit (b). 11 matches left after applying a spatial constraint to remove the outlier matches further away from the lowest Hamming distance match (green) (c).Cyan square showing the second frac-hit detected in the stage analyzed. Inset is a zoom-in view of the second frac-hit (d).

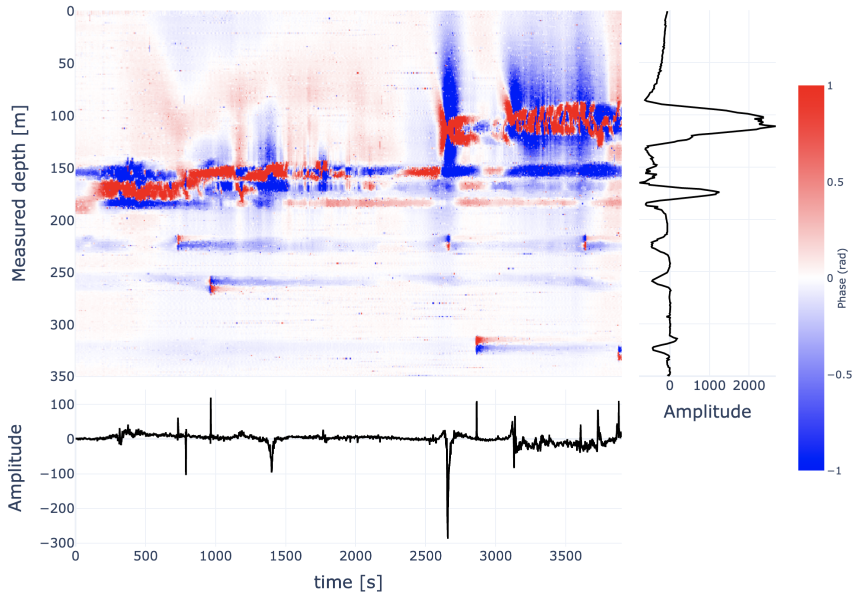


Figure 5.14: LFDAS strain data clipped between -1 and +1 radians. The right panel shows the cumulative strain in time, and the bottom panel shows the cumulative strain in depth.

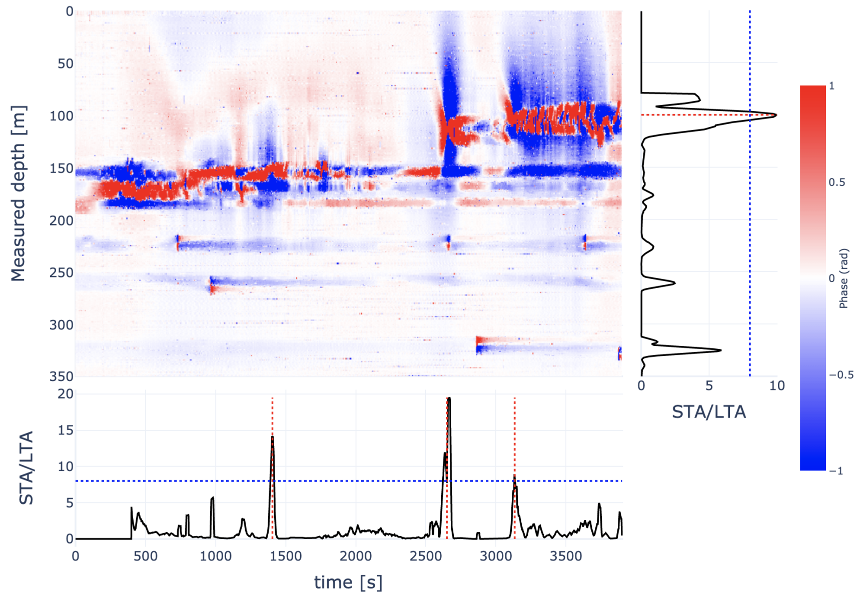
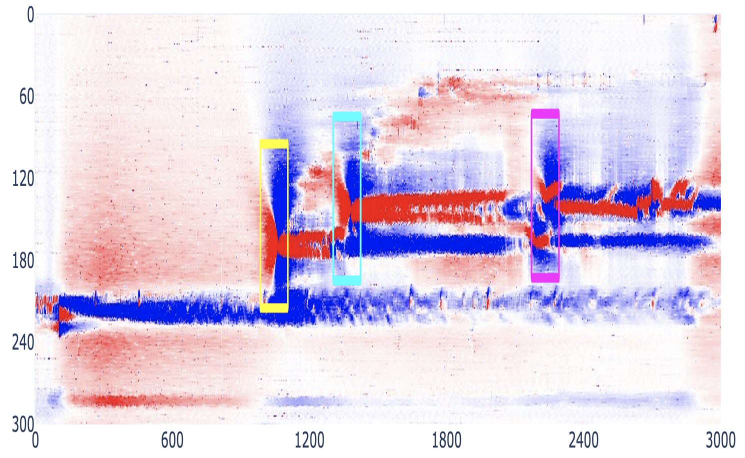
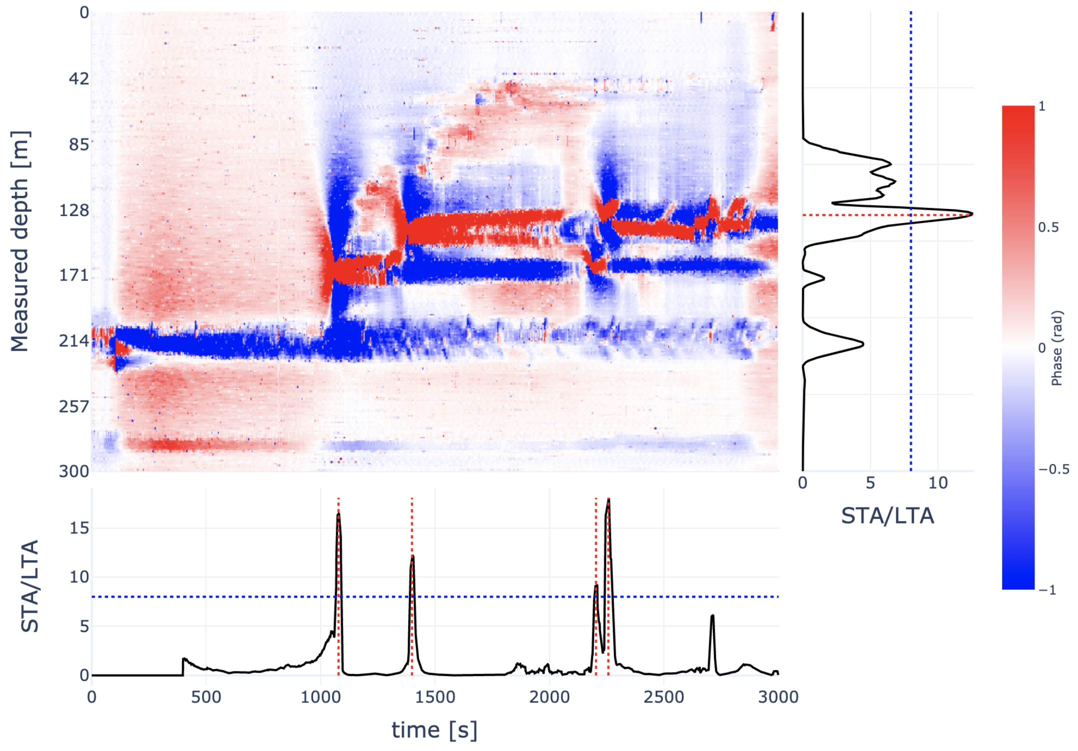


Figure 5.15: LFDAS strain data clipped between -1 and +1 radians. STA/LTA computed on the integrated LFDAS in time (right panel) and the integrated LFDAS in space (bottom panel). Blue dashed lines mark the detection threshold Λ, and red dashed lines are the detected frac-hit locations.

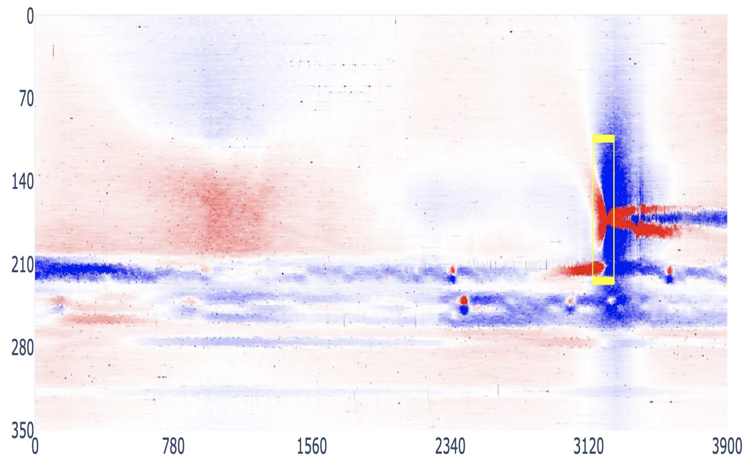


(a)

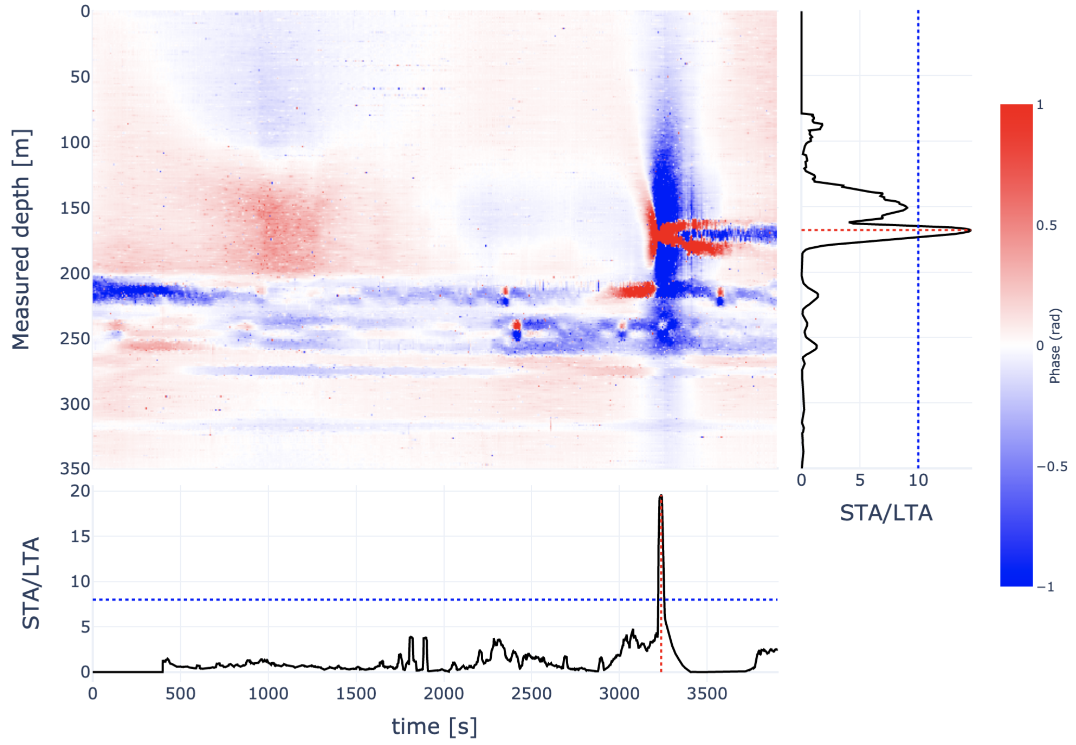


(b)

Figure 5.16: Three frac-hits detected (yellow, cyan, and magenta squares) using the proposed computer vision workflow (i.e., detection and subtraction) on another stage of the treatment (a). The detections (red dotted lines) using an STA/LTA on the integrated strain in time (right panel) and space (bottom panel) (b).



(a)



(b)

Figure 5.17: A frac-hit detected (yellow square) using the proposed computer vision workflow on a stage image of the treatment (a). The detections (red dotted lines) using an STA/LTA on the integrated strain in time (right panel) and space (bottom panel) (b).

Chapter 6

Conclusions

Multi-stage hydraulic fracturing in horizontal wells has enabled the development of vast unconventional reserves and will continue to play a significant role in meeting the increasing energy demand. Microseismic monitoring is a diagnostic technique used in real-time imaging during hydraulic fracturing. The spatio-temporal distribution of the microseismic clouds has proven to be effective in mapping fracture growth and geometry, which are important parameters used to improve the efficiency of the fracturing job. The first motivation of this thesis was to predict in real-time the microseismic cloud size during an hydraulic fracturing stimulation using routinely recorded engineering curves. We introduced three methodologies, and two of them can achieve real-time predictions of the microseismic cloud size.

The first method uses a random forests model trained with statistical features derived from the engineering curves and past cloud observations. Our initial approach to solving the microseismic cloud size prediction problem was to use data up to a particular time sample as the training set and leave the remaining data as the test set. However, by employing this training-testing strategy, the RF model achieves a poor generalization, forecasting microseismic cloud sizes that decrease at the end of the stages. Low accuracy in the test set is likely attributed to the surface and bottom-hole proppant concentrations, which drop to zero during the test time. Besides, the proposed RF model is not suitable for real-time applications. Predictions can only be made after more than half of the stage injection has passed, leaving only the end of the stage available to forecast.

We aim to develop tools that can improve the hydraulic fracturing treatments in real-time, for example, by allowing engineers to modify the completion strategy on the fly. The second methodology introduced in this work is based on a physical model that assumes a microseismic cloud growth proportional to the square root of time. We update our forecasts with every additional minute of data by computing the apparent hydraulic diffusivity from the present microseismic cloud size observations. The main advantage of this method is that it allows forecasting the microseismic cloud size at any time lag. More specifically, we can predict the cloud size at the end of the stage 5 minutes into the stimulation. However, prediction accuracy varies depending on the microseismic cloud behavior and the amount of time used to fit the hydraulic diffusivity. Thus, the main limitation of this method is that it only considers the microseismicity to make predictions and ignores every other data recorded during treatment.

The third predictive method uses a convolutional neural network trained in real-time with the engineering curves and historical values of the microseismic cloud size. We update the model parameters every minute, so forecasts can only be done at a predefined time lag subject to data availability. Predictions are better when the cloud remains flat compared to when the cloud experiences rapid growth spurts. Moreover, training a CNN with more parameters does not considerably improve the predictions during complex microseismic cloud growth. We believe that the data available to the CNN cannot account for the interplay of several physical processes affecting hydraulic fracture growth and microseismicity. We conclude that adding more physics into the CNN models can improve predictions, for example, by using the output of the diffusion-based approach as input to the network. Nevertheless, the CNN achieves the best results from the three methods presented in this thesis, providing an integrated methodology to utilize routinely recorded data like the engineering curves to predict microseismicity.

Another diagnostic technique used in real-time hydraulic fracture imaging is low-frequency DAS. The time and depth at which a fracture intersects an offset fibered well (frac-hit) can be used to constrain fracture geometry and optimize the treatment. The second motivation in this thesis was to automate the frac-hit detection in LFDAS using a computer vision workflow based on keypoints matching. We developed an integrated method that uses a frac-hit template to match frac-hits at any hydraulic fracturing treatment stage. Furthermore, our implementation allows removing the strain signal to reveal parts of the frac-hits obscured by overlapping linear strain variations.

The proposed computer vision workflow exceeds the performance of an STA/LTA run in the integrated strain, especially for locating the depth of multiple frac-hits in the same stage. This is most likely because the cumulative strain is influenced by signals generated during fracture opening and closing that persist across stages. To conclude, the proposed workflow can help interpret LF-DAS data, reduce human bias due to manual picking, and complement existing methods that rely on the cumulative strain to detect fracture hits.

6.1 Suggested future research

In this work, we only used the engineering curves as the input data for our predictive models. However, the use of physics-based machine learning can improve microseismic prediction. The challenge will be to find ways of integrating physics into the predictive models as data are not always accessible or in the format required by the specific application (e.g., time-series data).

Future work includes a more in-depth analysis of the frac-hits after subtraction to get additional insights into the underlying strain information. Furthermore, we focused solely on the detection and extraction of frac-hits. However, the proposed image matching workflow can be applied to identify other interest features in the LFDAS data, like rapid strain pulses.

Bibliography

Abe, H., Mura, T., & Keer, L. M. (1976). Growth rate of a penny-shaped crack in hydraulic fracturing of rocks. *J Geophys Res*, *81*(29), 5335–5340. doi:10.1029/JB081i029p05335

Adachi, J., Siebrits, E., Peirce, A., & Desroches, J. (2007). Computer simulation of hydraulic fractures. *International Journal of Rock Mechanics and Mining Sciences*, *44*(5), 739–757. doi:10.1016/j.ijrmms.2006.11.006

Akram, J. (2020). *Understanding Downhole Microseismic Data Analysis*. doi:10.1007/978-3-030-34017-9_1

Akram, J., & Eaton, D. W. (2016). A review and appraisal of arrival-time picking methods for downhole microseismic data. *Geophysics*, *81*(2), 67–87. doi:10.1190/GEO2014-0500.1

Albright, J. N., & Pearson, C. F. (1982). Acoustic Emissions as a Tool for Hydraulic Fracture Location: Experience at the Fenton Hill Hot Dry Rock Site. In *Society of Petroleum Engineers journal* (Vol. 22, *4*, pp. 523–530). doi:10.2118/9509-PA

Allen, R. V. (1978). Automatic earthquake recognition and timing from single traces. *Bulletin of the Seismological Society of America*, *68*(5), 1521–1532.

Arrowsmith, S. J., & Eisner, L. (2006). A technique for identifying microseismic multiplets and application to the Valhall field, North Sea. *Geophysics*, *71*(2), V31–V40. doi:10.1190/1.2187804

Awad, A. I., & Hassaballah, M. (2016). *Image Feature Detectors and Descriptors* (1st ed.). doi:10.1007/978-3-319-28854-3_1

Baig, A. M., Urbancic, T., & Prince, M. (2010). Microseismic moment tensors: A path to understanding growth of hydraulic fractures. In *Society of Petroleum Engineers - Canadian Unconventional Resources and International Petroleum Conference 2010* (Vol. 3, pp. 2133–2140). doi:10.2118/137771-ms

Bakku, S. K. (2015). *Fracture Characterization from Seismic Measurements in a Borehole* (Doctoral dissertation, Massachusetts Institute of Technology).

Barnoski, M. K., & Jensen, S. M. (1976). Fiber waveguides: a novel technique for investigating attenuation characteristics. *Applied Optics*, *15*(9), 2112. doi:10.1364/ao.15.002112

Barthwal, H., & van der Baan, M. (2017). Stress Perturbations and Microseismicity Induced by Hydraulic Fracturing. In *51st U.S. Rock Mechanics/-Geomechanics Symposium.* doi:ARMA-2017-0136

Barthwal, H., & van der Baan, M. (2019). Role of fracture opening in triggering microseismicity observed during hydraulic fracturing. *Geophysics*, *84*(3), KS105–KS118. doi:10.1190/geo2018-0425.1

Bay, H., Tuytelaars, T., & Van Gool, L. (2006). SURF: Speeded up robust features. In *Lecture Notes in Computer Science* (Vol. 3951 LNCS, pp. 404–417). doi:10.1007/11744023_32

Belyadi, H., Fathi, E., & Belyadi, F. (2017). Fracture Treatment Design. In *Hydraulic Fracturing in Unconventional Reservoirs* (pp. 143–167). doi:10.1016/B978-0-12-849871-2.00010-1

Ben, Y., James, C., & Cao, D. (2019). Development and application of a real-time drilling state classification algorithm with machine learning. In *SPE/AAPG/SEG Unconventional Resources Technology Conference.* doi:10.15530/urtec-2019-253

Ben, Y., Perrotte, M., Ezzatabadipour, M., Ali, I., Sankaran, S., Harlin, C., & Cao, D. (2020). Real time hydraulic fracturing pressure prediction with machine learning. In *SPE Hydraulic Fracturing Technology Conference and Exhibition*. doi:10.2118/199699-ms

Biot, M. A. (1962). Mechanics of deformation and acoustic propagation in porous media. *Journal of Applied Physics*, *33*(4), 1482–1498. doi:10.1063/1.1728759

Bishop, C. (2006). *Pattern Recognition and Machine Learning* (1st ed.). Springer-Verlag New York.

Boone, K., Crickmore, R., Werdeg, Z., Laing, C., & Molenaar, M. (2015). Monitoring Hydraulic Fracturing Operations Using Fiber-Optic Distributed Acoustic Sensing. In *SPE/AAPG/SEG Unconventional Resources Technology Conference*. doi:10.15530/urtec-2015-2158449

Boroumand, N., & Eaton, D. W. (2015). Energy-based hydraulic fracture numerical simulation: Parameter selection and model validation using microseismicity. *Geophysics*, *80*(5), W33–W44. doi:10.1190/GEO2014-0091.1

Box, G. E., Jenkins, G. M., Reinsel, G. C., & Ljung G. M. (2015). *Time Series Analysis: Forecasting and Control* (5th ed.). John Wiley & Sons.

Bradski, G. (2000). The OpenCV Library. *Dr. Dobb's Journal of Software Tools*.

Breiman, L. (1996). Bagging Predictors. *24*, 123–140. doi:10.1023/A:101805431

Breiman, L. (2001). Random Forests. *45*, 5–32. doi:10.1023/A:101093340

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). *Classification and regression trees*. Monterey, CA: Wadsworth & Brooks/Cole Advanced Books & Software.

Brune, J. N. (1970). Tectonic stress and the spectra of seismic shear waves from earthquakes. *J Geophys Res*, *75*(26), 4997–5009. doi:10.1029/jb075i026p04997

Byerlee, J. (1978). Friction of rocks. *Pure and Applied Geophysics*, *116*(4-5), 615–626. doi:10.1007/BF00876528

Caffagni, E., Eaton, D. W., Jones, J. P., & van der Baan, M. (2016). Detection and analysis of microseismic events using a Matched Filtering Algorithm (MFA). *Geophysical Journal International*, *206*(1), ggw168. doi:10.1093/gji/ggw168

Calonder, M., Lepetit, V., Strecha, C., & Fua, P. (2010). BRIEF: Binary Robust Independent Elementary Features. In *Lecture Notes in Computer Science* (Vol. 6314 LNCS, *PART 4*, pp. 778–792). doi:10.1007/978-3-642-15561-1_56

Canny, J. (1986). A Computational Approach to Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *PAMI-8*(6), 679–698. doi:10.1109/TPAMI.1986.4767851

Cao, Q., Banerjee, R., Gupta, S., Li, J., Zhou, W., & Jeyachandra, B. (2016). Data Driven Production Forecasting Using Machine Learning. In *SPE Argentina Exploration and Production of Unconventional Resources Symposium*. doi:10.2118/180984-MS

Carter, E. D. (1957). Optimum Fluid Characteristics for Fracture Extension. In *Drilling and Production Practices (ed. Howard, G. C. & Fast, C. R.)* (pp. 261–270). doi:API-57-261

Chen, Q., Defrise, M., & Deconinck, F. (1994). Symmetric phase-only matched filtering of Fourier-Mellin transforms for image registration and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *16*(12), 1156–1168. doi:10.1109/34.387491

Chiang, K. K. (1984). The Giant Hoadley Gas Field, South-Central Alberta. In *Elmworth: Case Study of a Deep Basin Gas Field* (Vol. 38, pp. 297–313). doi:10.1306/m38441c13

Cipolla, C. L., Williams, M. J., Weng, X., Mack, M., & Maxwell, S. (2010). Hydraulic fracture monitoring to reservoir simulation: Maximizing value.

In *Society of Petroleum Engineers - SPE* (pp. 1–26). doi:10.2118/133877-ms

Cipolla, C., Maxwell, S., Mack, M., & Downie, R. (2011). A practical guide to interpreting microseismic measurements. In *Society of Petroleum Engineers - SPE/EAGE European Unconventional Resources Conference and Exhibition 2012* (pp. 49–76). doi:10.2118/144067-ms

Cipolla, C., & Wright, C. (2000). State-of-the-Art in Hydraulic Fracture Diagnostics. doi:10.2118/64434-ms

Cole, S., Karrenbach, M., Boone, K., Ridge, A., Kahn, D., Rich, J., … Langton, D. (2017). Effective Diffusivity Estimates from Distributed Fiber-optic Strain and Microseismic Measurements. In *SEG Technical Program Expanded Abstracts* (pp. 2757–2761). doi:10.1190/segam2017-17680716.1

De Meersman, K., van der Baan, M., & Kendall, J. M. (2006). Signal extraction and automated polarization analysis of multicomponent array data. *Bulletin of the Seismological Society of America*, *96*(6), 2415–2430. doi:10.1785/0120050235

Dou, S., Lindsey, N., Wagner, A. M., Daley, T. M., Freifeld, B., Robertson, M., … Ajo-Franklin, J. B. (2017). Distributed Acoustic Sensing for Seismic Monitoring of the Near Surface: A Traffic-Noise Interferometry Case Study. *Scientific Reports*, *7*(1), 1–12. doi:10.1038/s41598-017-11986-4

Drew, J., White, R., & Wolfe, J. (2008). Microseismic event azimuth estimation: Establishing a relationship between hodogram linearity and uncertainty in event azimuth. In *SEG Technical Program Expanded Abstracts* (Vol. 27, *1*, pp. 1446–1450). doi:10.1190/1.3059186

Duncan, P. M., & Eisner, L. (2010). Reservoir characterization using surface microseismic monitoring. *Geophysics*, *75*(5). doi:10.1190/1.3467760

Earle, P. S., & Shearer, P. M. (1994). Characterization of global seismograms using an automatic-picking algorithm. *Bulletin of the Seismological Society of America*, *84*(2), 366–376.

Eaton, D. W. (2018). *Passive Seismic Monitoring of Induced Seismicity: Fundamental Principles and Application to Energy Technologies.* doi:10.1017/9781316535547

Eaton, D., van der Baan, M., Matthews, L., & Caffagni, E. (2014). Passive Seismic Monitoring and Integrated Geomechanical Analysis of a Tight-Sand Reservoir During Hydraulic-Fracture Treatment, Flowback and Production. In *Proceedings of the 2nd Unconventional Resources Technology Conference.* doi:10.15530/urtec-2014-1929223

Economides, M., & Nolte, K. (2000). *Reservoir Stimulation* (3rd ed.). Wiley.

Efford, N. (2000). *Digital Image Processing: A Practical Introduction using Java* (1st ed.). Addison-Wesley.

EIA. (2021). *Annual Energy Outlook 2021.* U.S. Energy Information Administration. Washington, DC.

Eisner, L., Duncan, P. M., Heigl, W. M., & Keller, W. R. (2009). Uncertainties in passive seismic monitoring. *Leading Edge (Tulsa, OK)*, *28*(6), 648–655. doi:10.1190/1.3148403

Eisner, L., Thornton, M., & Griffin, J. (2011). Challenges for microseismic monitoring. *SEG Technical Program Expanded Abstracts*, *30*(1), 1519–1523. doi:10.1190/1.3627491

Fischer, T., & Guest, A. (2011). Shear and tensile earthquakes caused by fluid injection. *Geophysical Research Letters*, *38*(5). doi:10.1029/2010GL045447

Fischler, M. A., & Bolles, R. C. (1981). Random sample consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Communications of the ACM*, *24*(6), 381–395. doi:10.1145/358669.358692

Fisher, K., & Warpinski, N. (2012). Hydraulic-fracture-height growth: Real data. In *SPE Production and Operations* (Vol. 27, *1*, pp. 8–19). doi:10.2118/145949-pa

Fisher, M., Heinze, J., Harris, C., Davidson, B., Wright, C., & Dunn, K. (2004). Optimizing Horizontal Completion Techniques in the Barnett Shale Using Microseismic Fracture Mapping. In *SPE Annual Technical Conference and Exhibition*. doi:10.2118/90051-MS

Fisher, M., Wright, C., Davidson, B., Goodwin, A., Fielder, E., Buckler, W., & Steinsberger, N. (2002). Integrating Fracture Mapping Technologies to Optimize Stimulations in the Barnett Shale. doi:10.2118/77441-ms

Fossen, H. (2016). *Structural Geology* (2nd ed.). Cambridge University Press.

Geertsma, J., & de Klerk, F. (1969). A Rapid Method of Predicting Width and Extent of Hydraulically Induced Fractures. *Journal of Petroleum Technology*, *21*(12), 1571–1581. doi:10.2118/2458-PA

Geiger, L. (1912). Probability method for the determination of earthquake epicenters from the arrival time only. *Bull. St. Louis Univ*, (8), 60–71.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.

Grattan, L., & Meggitt, B. (2000). *Optical Fiber Sensor Technology: Advanced Applications - Bragg Gratings and Distributed Sensors* (1st ed.). Springer US.

Griffith, A. (1924). The theory of rupture. In *First Int. Cong. Appl. Mech* (pp. 55–63).

Griffith, A. A. (1921). The Phenomena of Rupture and Flow in Solids, 163–198. doi:10.1098/rsta.1921.0006

Grigoli, F., Cesca, S., Krieger, L., Kriegerowski, M., Gammaldi, S., Horalek, J., … Dahm, T. (2016). Automated microseismic event location using Master-Event Waveform Stacking. *Scientific Reports*, *6*(1), 1–13. doi:10.1038/srep25744

Grob, M., & van der Baan, M. (2011). Inferring in-situ stress changes by statistical analysis of microseismic event characteristics. *Leading Edge (Tulsa, OK)*, *30*(11), 1296–1301. doi:10.1190/1.3663403

Grömping, U. (2009). Variable importance assessment in regression: Linear regression versus random forest. *American Statistician*, *63*(4), 308–319. doi:10.1198/tast.2009.08199

Gutenberg, B., & Richter, C. F. (1944). Frequency of earthquakes in California. *Bulletin of the Seismological Society of America*, *34*(4), 185–188.

Han, J., & van der Baan, M. (2015). Microseismic and seismic denoising via ensemble empirical mode decomposition and adaptive thresholding. *Geophysics*, *80*(6), KS69–KS80. doi:10.1190/GEO2014-0423.1

Harris, C., & Stephens, M. (1988). A combined corner and edge detector. In *Proceedings of Fourth Alvey Vision Conference* (pp. 147–151). doi:10.1.1.434.4816

Harris, D. B. (2006). *Subspace Detectors: Theory*. Lawrence Livermore National Laboratory (LLNL). doi:10.2172/900081

Hartog, A. H. (2017). *An introduction to distributed optical fibre sensors*. doi:10.1201/9781315119014

Hayes, B., Christopher, J., Los, G., McKercher, B., Minken, D., Tremblay, Y., … Smith, D. (1994). Cretaceous Mannville Group of the western Canada sedimentary basin. In *Geological Atlas of the Western Canada sedimentary basin* (pp. 317–334). Canadian Society of Petroleum Geologists and Alberta Research Council.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (Vol. 2016-Decem, pp. 770–778). doi:10.1109/CVPR.2016.90. arXiv: 1512.03385

Healey, P. (1984). Fading in heterodyne OTDR. *Electronics Letters*, *20*(1), 30–32. doi:10.1049/el:19840022

Healy, D., Jones, R. R., & Holdsworth, R. E. (2006). Three-dimensional brittle shear fracturing by tensile crack interaction. *Nature*, *439*(7072), 64–67. doi:10.1038/nature04346

Holland, A. A. (2013). Earthquakes triggered by hydraulic fracturing in south-central Oklahoma. *Bulletin of the Seismological Society of America*, *103*(3), 1784–1792. doi:10.1785/0120120109

Holley, E., & Kalia, N. (2015). Fiber-Optic Monitoring: Stimulation Results From Unconventional Reservoirs. doi:10.15530/urtec-2015-2151906

Hyndman, R. J., & Athanasopoulos, G. (2018). *Forecasting: principles and practice* (3rd ed.). OTexts.

Ichikawa, M., Kurosawa, I., & Uchida, S. (2020). Case study of hydraulic fracture monitoring using low-frequency components of DAS data. *SEG International Exposition and Annual Meeting 2019*, 948–952. doi:10.1190/segam2019-3214251.1

Ito, T., & Zoback, M. D. (2000). Fracture permeability and in situ stress to 7 km depth in the KTB scientific drillhole. *Geophysical Research Letters*, *27*(7), 1045–1048. doi:10.1029/1999GL011068

Jaeger, J. C., Cook, G. W., & Zimmerman, R. (2007). *Fundamentals of rock mechanics*. Blackwell Pub.

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An introduction to Statistical Learning*. doi:10.1007/978-1-4614-7138-7

Jiang, Y., Zur, R. M., Pesce, L. L., & Drukker, K. (2009). A study of the effect of noise injection on the training of artificial neural networks. In *Proceedings of the International Joint Conference on Neural Networks* (pp. 1428–1432). doi:10.1109/IJCNN.2009.5178981

Jin, G., & Roy, B. (2017). Hydraulic-fracture geometry characterization using low-frequency DAS signal. *Leading Edge*, *36*(12), 975–980. doi:10.1190/tle36120975.1

Jurvekys, A. (1988). Polarization analysis of three-component array data. *Bulletin of the Seismological Society of America*, *78*(5), 1725–1743.

Kanamori, H. (1977). The energy release in great earthquakes. *Journal of Geophysical Research*, *82*(20), 2981–2987. doi:10.1029/jb082i020p02981

Karpatne, A., Watkins, W., Read, J., & Kumar, V. (2017). Physics-guided Neural Networks (PGNN): An Application in Lake Temperature Modeling. *arXiv preprint arXiv:1710.11431*. arXiv: 1710.11431

Karrenbach, M., Ridge, A., Cole, S., Boone, K., Kahn, D., Rich, J., . . . Langton, D. (2017). DAS microseismic monitoring and integration with strain measurements in hydraulic fracture profiling. In *SPE/AAPG/SEG Unconventional Resources Technology Conference 2017*. doi:10.15530/urtec-2017-2670716

Khan, S. A., & Saleem, Z. (2018). A comparative analysis of SIFT, SURF, KAZE, AKAZE, ORB, and BRISK. In *2018 International Conference on Computing, Mathematics and Engineering Technologies: Invent, Innovate and Integrate for Socioeconomic Development, iCoMET 2018 - Proceedings* (Vol. 2018-Janua, pp. 1–10). doi:10.1109/ICOMET.2018.8346440

King, G. E. (2010). Thirty Years of Gas Shale Fracturing: What Have We Learned? In *SPE Annual Technical Conference and Exhibition*. doi:10.2118/133456-MS

Kingma, D. P., & Ba, J. L. (2015). Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, International Conference on Learning Representations, ICLR. arXiv: 1412.6980

Kiranyaz, S., Avci, O., Abdeljaber, O., Ince, T., Gabbouj, M., & Inman, D. J. (2019). 1D Convolutional Neural Networks and Applications: A Survey. *Mechanical Systems and Signal Processing*, *151*. arXiv: 1905.03554

Kong, Q., Trugman, D., Ross, Z. E., Bianco, M. J., Meade, B. J., & Gerstoft, P. (2019). Machine learning in seismology: Turning data into insights. *Seismological Research Letters*, *90*(1), 3–14. doi:10.1785/0220180259

Korjani, M., Popa, A., Grijalva, E., Cassidy, S., & Ershaghi, I. (2016). A new approach to reservoir characterization using deep learning neural networks. In *Society of Petroleum Engineers - SPE Western Regional Meeting*. doi:10.2118/180359-ms

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems* (Vol. 25). doi:10.1145/3065386

Krohn, D. A., MacDougall, T. W., & Mendez, A. (2014). *Fiber Optic Sensors: Fundamentals and Applications*. doi:10.1117/3.1002910

Le Calvez, J. H., Bennett, L., Tanner, K. V., Grant, W. D., Nutt, L., Jochen, V., . . . Drew, J. (2005). Monitoring microseismic fracture development to optimize stimulation and production in aging fields. *The Leading Edge*, *24*(1), 72–75. doi:10.1190/1.1859705

Le Calvez, J., Malpani, R., Xu, J., Stokes, J., & Williams, M. (2016). Hydraulic Fracturing Insights from Microseismic Monitoring. *Oilfield Review*, *28*(2), 16–33.

LeCun, Y., & Bengio, Y. (1998). Convolutional networks for images, speech, and time-series. In *The handbook of brain theory and neural networks* (pp. 255–258). doi:10.5555/303568.303704

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. doi:10.1038/nature14539

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2323. doi:10.1109/5.726791

Lehtinen, J., Munkberg, J., Hasselgren, J., Laine, S., Karras, T., Aittala, M., & Aila, T. (2018). Noise2Noise: Learning Image Restoration without Clean Data. *35th International Conference on Machine Learning, ICML 2018*, *7*, 4620–4631. arXiv: 1803.04189

Leutenegger, S., Chli, M., & Siegwart, R. Y. (2011). BRISK: Binary Robust invariant scalable keypoints. In *Proceedings of the IEEE International*

Conference on Computer Vision (pp. 2548–2555). doi:10.1109/ICCV.2011.6126542

Li, F., Rich, J., Marfurt, K. J., & Zhou, H. (2014). Automatic event detection on noisy microseismograms. In *SEG Technical Program Expanded Abstracts* (Vol. 33, pp. 2363–2367). doi:10.1190/segam2014-1605.1

Li, F. F. (2011). *Detectors and descriptors.* Stanford Vision Lab.

Li, X., Zhang, J., Grubert, M., Laing, C., Chavarria, A., Cole, S., & Oukaci, Y. (2020). Distributed acoustic and temperature sensing applications for hydraulic fracture diagnostics. *Society of Petroleum Engineers - SPE Hydraulic Fracturing Technology Conference and Exhibition 2020, HFTC 2020.* doi:10.2118/199759-ms

Li, Y., Wang, S., Tian, Q., & Ding, X. (2015). A survey of recent advances in visual feature detection. *Neurocomputing*, *149*(PB), 736–751. doi:10.1016/j.neucom.2014.08.003

Liaw, A., & Wiener, M. (2002). *Classification and Regression by RandomForest.*

Lindeberg, T. (1998). Feature Detection with Automatic Scale Selection. *International Journal of Computer Vision*, *30*(2), 79–116. doi:10.1023/A:1008045108935

Lindsey, N. J., Rademacher, H., & Ajo-Franklin, J. B. (2020). On the Broadband Instrument Response of Fiber-Optic DAS Arrays. *Journal of Geophysical Research: Solid Earth*, *125*(2). doi:10.1029/2019JB018145

Louppe, G., Wehenkel, L., Sutera, A., & Geurts, P. (2013). Understanding variable importances in forests of randomized trees. In *Advances in Neural Information Processing Systems* (Vol. 26, pp. 431–439). doi:10.5555/2999611.2999660

Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *Proceedings of the IEEE International Conference on Computer Vision* (Vol. 2, pp. 1150–1157). doi:10.1109/iccv.1999.790410

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, *60*(2), 91–110. doi:10.1023/B:VISI.0000029664.99615.94

Lubbers, N., Bolton, D., Mohd-Yusof, J., Marone, C., Barros, K., & Johnson, P. (2018). Earthquake Catalog-Based Machine Learning Identification of Laboratory Fault States and the Effects of Magnitude of Completeness. *Geophysical Research Letters*, *45*(24), 13, 269–13, 276. doi:10.1029/2018GL079712

Luo, G., Tian, Y., Bychina, M., & Ehlig-Economides, C. (2018). Production optimization using machine learning in bakken shale. *SPE/AAPG/SEG Unconventional Resources Technology Conference 2018, URTC 2018*, (2011). doi:10.15530/urtec-2018-2902505

Ma, Z., Davani, E., Ma, X., Lee, H., Arslan, I., Zhai, X., … Castineira, D. (2020). Finding a trend out of chaos, a machine learning approach for well spacing optimization. In *Proceedings - SPE Annual Technical Conference and Exhibition* (Vol. 2020-Octob). doi:10.2118/201698-ms

Mack, M., & Warpinski, N. (2000). Mechanics of Hydraulic Fracturing. In *Reservoir Stimulation* (Chap. 9).

Maghsoudi, S., Eaton, D., & Davidsen, J. (2016). Nontrivial clustering of microseismicity induced by hydraulic fracturing. *Geophysical Research Letters*, *43*(20), 10, 672–10, 679. doi:10.1002/2016GL070983

Maity, D., Aminzadeh, F., & Karrenbach, M. (2014). Novel hybrid artificial neural network based autopicking workflow for passive seismic data. *Geophysical Prospecting*, *62*(4), 834–847. doi:10.1111/1365-2478.12125

Mateeva, A., Lopez, J., Potters, H., Mestayer, J., Cox, B., Kiyashchenko, D., … Detomo, R. (2014). Distributed acoustic sensing for reservoir monitoring with vertical seismic profiling. *Geophysical Prospecting*, *62*(4), 679–692. doi:10.1111/1365-2478.12116

Maucec, M., & Garni, S. (2019). Application of Automated Machine Learning for Multi-Variate Prediction of Well Production. In *SPE Middle East Oil and Gas Show and Conference*. doi:10.2118/195022-MS

Maxwell, S. (2014). *Microseismic Imaging of Hydraulic Fracturing* (1st ed.). doi:10.1190/1.9781560803164

Maxwell, S. C., Rutledge, J., Jones, R., & Fehler, M. (2010). Petroleum reservoir characterization using downhole microseismic monitoring. *Geophysics*, *75*(5). doi:10.1190/1.3477966

Maxwell, S., Jones, M., Parker, R., Miong, S., Leaney, S., Dorval, D., ... Hammermaster, K. (2009a). Fault activation during hydraulic fracturing. In *SEG Technical Program Expanded Abstracts 2009* (pp. 1552–1556). doi:10.1190/1.3255145

Maxwell, S., Shemeta, J., Campbell, E., & Quirk, D. (2008). Microseismic Deformation Rate Monitoring. *SPE Annual Technical Conference and Exhibition*. doi:10.2118/116596-MS

Maxwell, S. C., Waltman, C. K., Warpinski, N. R., Mayerhofer, M. J., & Baroumand, N. (2009b). Imaging seismic deformation induced by hydraulic fracture complexity. *SPE Reservoir Evaluation and Engineering*, *12*(1), 48–52. doi:10.2118/102801-pa

Mayerhofer, M. J., Lolon, E. P., Rightmire, C., Walser, D., Cipolla, C. L., & Warplnskl, N. R. (2010). What is stimulated reservoir volume? *SPE Production and Operations*, *25*(1), 89–98. doi:10.2118/119890-PA

McKean, S. H., Priest, J. A., Dettmer, J., & Eaton, D. W. (2019). Quantifying Fracture Networks Inferred From Microseismic Point Clouds by a Gaussian Mixture Model With Physical Constraints. *Geophysical Research Letters*, *46*(20), 11008–11017. doi:10.1029/2019GL083406

Mestayer, J., Cox, B., Wills, P., & Kiyashchenko, D. (2011). Field trials of distributed acoustic sensing for geophysical monitoring. *SEG Technical Program Expanded Abstracts*, *30*(1), 4253–4257. doi:10.1190/1.3628095

Mohri, M., Rostamizadeh, A., & Talwalkar, A. (2012). *Foundations of machine learning*. MIT Press.

Molenaar, M. M., Hill, D. J., Webster, P., Fidan, E., & Birch, B. (2011). First downhole application of Distributed Acoustic Sensing (DAS) for hydraulic fracturing monitoring and diagnostics. In *Society of Petroleum Engineers - SPE Hydraulic Fracturing Technology Conference 2011* (pp. 751–759). doi:10.2118/140561-ms

Molenaar, M. M., & Cox, B. E. (2013). Field cases of hydraulic fracture stimulation diagnostics using fiber optic distributed acoustic sensing (DAS) measurements and analyses. In *Society of Petroleum Engineers - SPE Middle East Unconventional Gas Conference and Exhibition 2013, UGAS 2013 - Unconventional and Tight Gas: Bridging the Gaps for Sustainable Economic Development* (pp. 754–763). doi:10.2118/164030-ms

Montgomery, C. T., & Smith, M. B. (2010). Hydraulic fracturing: History of an enduring technology. *JPT, Journal of Petroleum Technology*, *62*(12), 26–32. doi:10.2118/1210-0026-jpt

Moravec, H. (1981). *Rover Visual Obstacle Avoidance.*

Mousa, W. A., Al-Shuhail, A. A., & Al-Lehyani, A. (2011). A new technique for first-arrival picking of refracted seismic data based on digital image segmentation. *Geophysics*, *76*(5), V79–V89. doi:10.1190/geo2010-0322.1

Mousavi, S. M., Langston, C. A., & Horton, S. P. (2016). Automatic microseismic denoising and onset detection using the synchrosqueezed continuous wavelet transform. *Geophysics*, *81*(4), V341–V355. doi:10.1190/GEO2015-0598.1

Nair, V., & Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning* (Vol. 1, pp. 807–814). doi:10.1109/CVPR.2005.202

Nakata, N., & Beroza, G. C. (2016). Reverse time migration for microseismic sources using the geometric mean as an imaging condition. *Geophysics*, *81*(2), KS51–KS60. doi:10.1190/GEO2015-0278.1

Newbert, J. D., & Trick, M. D. (1987). A systematic deliverability analysis for the Hoadley gas condensate field. doi:10.2118/16938-ms

Nocedal, J., & Wright, S. (2006). *Numerical optimization* (2nd ed.). doi:10.1007/978-0-387-40065-5

Nordgren, R. (1972). Propagation of a Vertical Hydraulic Fracture. *Society of Petroleum Engineers Journal*, *12*(04), 306–314. doi:10.2118/3009-pa

Oye, V., & Roth, M. (2003). Automated seismic event location for hydrocarbon reservoirs. *Computers and Geosciences*, *29*(7), 851–863. doi:10.1016/S0098-3004(03)00088-8

Parker, T., Shatalin, S., & Farhadiroushan, M. (2014). Distributed Acoustic Sensing - A new tool for seismic applications. *First Break*, *32*(2), 61–69. doi:10.3997/1365-2397.2013034

Pedregosa, F., Michel, V., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., ... Perrot, M. (2011). *Scikit-learn: Machine Learning in Python.*

Perkins, T., & Kern, L. (1961). Widths of Hydraulic Fractures. *Journal of Petroleum Technology*, *13*(09), 937–949. doi:10.2118/89-pa

Pinnegar, C. R. (2006). Polarization analysis and polarization filtering of three-component signals with the time-frequency. *Geophysical Journal International*, *165*(2), 596–606. doi:10.1111/j.1365-246X.2006.02937.x

Pratt, W. K. (2007). *Digital Image Processing.* doi:10.1002/0470097434

Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. (2007). *Numerical recipes The Art of Scientific Computing* (3rd ed.). Cambridge University Press.

Raab, T., Reinsch, T., Aldaz Cifuentes, S. R., & Henninges, J. (2019). Real-Time Well-Integrity Monitoring Using Fiber-Optic Distributed Acoustic Sensing. *SPE Journal*, *24*(05), 1997–2009. doi:10.2118/195678-pa

Rafiq, A., Eaton, D. W., Mcdougall, A., & Pedersen, K. (2016). Reservoir characterization using microseismic facies analysis integrated with surface seismic attributes. *The Leading Edge*. doi:10.1190/INT-2015-0109.1

Reynolds, M., Thomson, S., Peyman, F., Hung, A., Quirk, D., & Chen, S. (2012). A direct comparison of hydraulic fracture geometry and well performance between cemented liner and open hole packer completed horizontal wells in a tight gas reservoir. In *Society of Petroleum Engineers - SPE Hydraulic Fracturing Technology Conference 2012* (pp. 431–449). doi:10.2118/152185-ms

Richter, P., Parker, T., Woerpel, C., Wu, W., Rufino, R., & Farhadiroushan, M. (2019). High-resolution Distributed Acoustic Sensor using engineered fiber for hydraulic fracture monitoring and optimization in unconventional completions. In *SEG International Exposition and Annual Meeting 2019* (pp. 4874–4878). doi:10.1190/segam2019-3215860.1

Rodi, W. (2006). Grid-search event location with non-Gaussian error models. *Physics of the Earth and Planetary Interiors*, *158*(1), 55–66. doi:10.1016/j.pepi.2006.03.010

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *Lecture Notes in Computer Science* (Vol. 9351, pp. 234–241). doi:10.1007/978-3-319-24574-4_28

Rosten, E., & Drummond, T. (2005). Fusing points and lines for high performance tracking. In *Proceedings of the IEEE International Conference on Computer Vision* (Vol. 2, pp. 1508–1515). doi:10.1109/ICCV.2005.104

Rosten, E., & Drummond, T. (2006). Machine learning for high-speed corner detection. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (Vol. 3951 LNCS, pp. 430–443). doi:10.1007/11744023_34

Rothert, E., & Shapiro, S. A. (2003). Microseismic monitoring of borehole fluid injections: Data modeling and inversion for hydraulic properties of rocks. *Geophysics*, *68*(2), 685–689. doi:10.1190/1.1567239

Rouet-Leduc, B., Hulbert, C., Lubbers, N., Barros, K., Humphreys, C., & Johnson, P. (2017). Machine Learning Predicts Laboratory Earthquakes. *Geophysical Research Letters*, *44*(18), 9276–9282. doi:10.1002/2017GL074677

Rozhko, A. Y. (2010). Role of seepage forces on seismicity triggering. *Journal of Geophysical Research: Solid Earth*, *115*(11), 1–12. doi:10.1029/2009JB007182

Rublee, E., Rabaud, V., Konolige, K., & Bradski, G. (2011). ORB: An efficient alternative to SIFT or SURF. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 2564–2571). doi:10.1109/ICCV.2011.6126544

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, *323*(6088), 533–536. doi:10.1038/323533a0

Rutledge, J. T., & Phillips, W. S. (2003). Hydraulic stimulation of natural fractures as revealed by induced microearthquakes, Carthage Cotton Valley gas field, east Texas. doi:10.1190/1.1567214

Scholz, C. H. (2019). *The Mechanics of Earthquakes and Faulting.* doi:10.1017/9781316681473

Schorlemmer, D., Wiemer, S., & Wyss, M. (2005). Variations in earthquake-size distribution across different stress regimes. *Nature*, *437*(7058), 539–542. doi:10.1038/nature04094

Scornet, E., Biau, G., & Vert, J. P. (2015). Consistency of random forests. *Annals of Statistics*, *43*(4), 1716–1741. doi:10.1214/15-AOS1321. arXiv:1405.2881

Segall, P., & Lu, S. (2015). Journal of Geophysical Research : Solid Earth Injection-induced seismicity : Poroelastic and earthquake nucleation effects. *Journal of Geophysical Research: Solid Earth*, *120*, 5082–5103. doi:10.1002/2015JB012060.Received

Shapiro, S. (2015). *Fluid-induced seismicity*. doi:10.1017/CBO9781139051132

Shapiro, S. A., & Dinske, C. (2009a). Fluid-induced seismicity: Pressure diffusion and hydraulic fracturing. In *Geophysical Prospecting* (Vol. 57, *2*, pp. 301–310). doi:10.1111/j.1365-2478.2008.00770.x

Shapiro, S. A., Dinske, C., & Rothert, E. (2006). Hydraulic-fracturing controlled dynamics of microseismic clouds. *Geophysical Research Letters*, *33*(14), L14312. doi:10.1029/2006GL026365

Shapiro, S., & Dinske, C. (2009b). Scaling of seismicity induced by nonlinear fluid-rock interaction. *J. Geophys. Res*, *114*, 9307. doi:10.1029/2008JB006145

Shapiro, S., Rothert, E., Rath, V., & Rindschwentner, J. (2002). Characterization of fluid transport properties of reservoirs using induced microseismicity. *Geophysics*, *67*(1), 212–220. doi:10.1190/1.1451597

Shapiro, S. A., Huenges, E., & Borm, G. (1997). Estimating the crust permeability from fluid-injection-induced seismic emission at the KTB site. *Geophysical Journal International*, *131*(2), F15–F18. doi:10.1111/j.1365-246X.1997.tb01215.x

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, International Conference on Learning Representations, ICLR. arXiv: 1409.1556

Smith, M. B., & Montgomery, C. (2015). *Hydraulic Fracturing* (1st ed.). CRC Press.

Sneddon, I., & Elliot, H. (1946). The opening of a Griffith crack under internal pressure. *Quarterly of Applied Mathematics*, *4*(3), 262–267.

Solem, J. (2012). *Programming Computer Vision with Python*. O'Reilly Media, Inc.

Stork, A., Verdon, J., & Kendall, J.-M. (2014). The robustness of seismic moment and magnitudes estimated using spectral analysis. *Geophysical Prospecting*, *62*(4), 862–878. doi:10.1111/1365-2478.12134

Tomasi, C., & Manduchi, R. (1998). Bilateral filtering for gray and color images. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 839–846). doi:10.1109/iccv.1998.710815

Twiss, R. J., & Moores, E. M. (2006). *Structural Geology* (2nd ed.). W. H. Freeman.

Ugueto, G. A., Ehiwario, M., Grae, A., Molenaar, M., McCoy, K., Huckabee, P., & Barree, B. (2014). Application of integrated advanced diagnostics and modeling to improve hydraulic fracture stimulation analysis and optimization. In *Society of Petroleum Engineers - SPE Hydraulic Fracturing Technology Conference 2014* (pp. 347–360). doi:10.2118/168603-ms

Ugueto, G. A., Todea, F., Daredia, T., & Wojtaszek, M. (2019). Can you feel the strain? DAS strain fronts for fracture geometry in the BC Montney, Groundbirch. In *Proceedings - SPE Annual Technical Conference and Exhibition* (Vol. 2019-Septe). doi:10.2118/195943-ms

Ugueto, G. A., Wojtaszek, M., Huckabee, P. T., Reynolds, A., Brewer, J., & Acosta, L. (2018). Accelerated Stimulation Optimization via Permanent and Continuous Production Monitoring Using Fiber Optic. doi:10.15530/urtec-2018-2901897

Vaezi, Y., & van der Baan, M. (2015). Comparison of the STA/LTA and power spectral density methods for microseismic event detection. *Geophysical Journal International*, *203*(3), 1896–1908. doi:10.1093/gji/ggv419

van der Baan, M., Eaton, D. W., & Preisig, G. (2016). Stick-split mechanism for anthropogenic fluid-induced tensile rock failure. *Geology*, *44*(7), 503–506. doi:10.1130/G37826.1

van der Baan, M., Eaton, D., & Dusseault, M. (2013). Microseismic Monitoring Developments in Hydraulic Fracture Stimulation. In *ISRM Inter-*

national Conference for Effective and Sustainable Hydraulic Fracturing 2013. doi:10.5772/56444

van der Baan, M., & Jutten, C. (2000). Neural networks in geophysical applications. *Geophysics*, *65*(4), 1032–1047. doi:10.1190/1.1444797

Vavryčuk, V. (2001). Inversion for parameters of tensile earthquakes. *Journal of Geophysical Research: Solid Earth*, *106*(B8), 16339–16355. doi:10.1029/2001jb000372

Vera Rodriguez, I., Bonar, D., & Sacchi, M. D. (2011). Microseismic record denoising using a sparse time-frequency transform. *SEG Technical Program Expanded Abstracts*, *30*(1), 1693–1698. doi:10.1190/1.3627530

Wang, K., Li, K., Zhou, L., Hu, Y., Cheng, Z., Liu, J., & Chen, C. (2019). Multiple convolutional neural networks for multivariate time series prediction. *Neurocomputing*, *360*, 107–119. doi:10.1016/j.neucom.2019.05.023

Wang, S., & Chen, S. (2019). Insights to fracture stimulation design in unconventional reservoirs based on machine learning modeling. *Journal of Petroleum Science and Engineering*, *174*, 682–695. doi:10.1016/j.petrol.2018.11.076

Warpinski, N. R. (2009a). Integrating microseismic monitoring with well completions, reservoir behavior, and rock mechanics. In *Tight Gas Completions Conference 2009* (pp. 26–38). doi:10.2118/125239-ms

Warpinski, N. R., Wolhart, S. L., & Wright, C. A. (2004). Analysis and prediction of microseismicity induced by hydraulic fracturing. *SPE Journal*, *9*(1), 24–33. doi:10.2118/87673-PA

Warpinski, N., Branagan, P., Peterson, R., & Wolhart, S. (1998). An Interpretation of M-Site Hydraulic Fracture Diagnostic Results. doi:10.2118/39950-ms

Warpinski, N. (2009b). Microseismic monitoring: Inside and out. *JPT, Journal of Petroleum Technology*, *61*(11), 80–85. doi:10.2118/118537-JPT

Warpinski, N. R., Sullivan, R. B., Uhl, J. E., Waltman, C. K., & Machovoe, S. R. (2005). Improved microseismic fracture mapping using perforation timing measurements for velocity calibration. *SPE Journal*, *10*(1), 14–23. doi:10.2118/84488-PA

Warpinski, N. R. (1996). Hydraulic fracture diagnostics. *JPT, Journal of Petroleum Technology*, *48*(10), 907–910. doi:10.2118/36361-JPT

Webster, P., Wall, J., Perkins, C., & Molenaar, M. (2013a). Micro-seismic detection using distributed acoustic sensing. In *Society of Exploration Geophysicists International Exposition and 83rd Annual Meeting, SEG 2013: Expanding Geophysical Frontiers* (pp. 2459–2463). doi:10.1190/segam2013-0182.1

Webster, P., Cox, B., & Molenaar, M. (2013b). Developments in diagnostic tools for hydraulic fracture geometry analysis. In *Unconventional Resources Technology Conference 2013, URTC 2013*. doi:10.1190/urtec2013-025

Webster, P., Molenaar, M., & Perkins, C. (2016). DAS Microseismic. *CSEG Recorder*, *41*(6).

Wu, Y., Richter, P., Hull, R., & Farhadiroushan, M. (2020). Hydraulic frac-hit corridor (FHC) monitoring and analysis with high-resolution distributed acoustic sensing (DAS) and far-field strain (FFS) measurements. *First Break*, *38*(June), 65–70. doi:10.3997/1365-2397.fb2020045

Zhang, C., & van der Baan, M. (2021). Complete and representative training of neural networks: A generalization study using double noise injection and natural images. *Geophysics*, 1–43. doi:10.1190/geo2020-0193.1

Zoback, M. D. (2007). *Reservoir Geomechanics*. Cambridge University Press.

Zoback, M. D., & Harjes, H. P. (1997). Injection-induced earthquakes and crustal stress at 9 km depth at the KTB deep drilling site, Germany. *Journal of Geophysical Research B: Solid Earth*, *102*(B8), 18477–18491. doi:10.1029/96jb02814