Conditioning reservoir models on vast data sets

Ricardo Vasconcellos Soares

Thesis for the degree of Philosophiae Doctor (PhD)
University of Bergen, Norway
2021

UNIVERSITY OF BERGEN

Conditioning reservoir models on vast data sets

Ricardo Vasconcellos Soares

Thesis for the degree of Philosophiae Doctor (PhD)
at the University of Bergen

© Copyright Ricardo Vasconcellos Soares

The material in this publication is covered by the provisions of the Copyright Act.

Year: 2021

Title: Conditioning reservoir models on vast data sets

Name: Ricardo Vasconcellos Soares

Print: Skipnes Kommunikasjon / University of Bergen

Preface

This thesis is submitted as a partial fulfillment of the requirements for the degree of Philosophiae Doctor (Ph.D.) at the University of Bergen. The supervising committee has consisted of Guttorm Alendal (University of Bergen), Xiaodong Luo (NORCE), and Geir Evensen (NORCE).

The Ph.D. project has been supported by NORCE Norwegian Research Centre AS through the Research Council of Norway (RCN no. 280473) and the industrial partners AkerBP, Wintershall DEA, Vår Energi, Petrobras, Equinor, Lundin, and Neptune Energy.

Acknowledgements

I am very grateful for all the people I have met during my Ph.D. It has been a long and enjoyable journey. I will take everything I learned during this period for the rest of my personal and professional life.

First, I would like to thank my supervisors Guttorm Alendal, Xiaodong Luo, and Geir Evensen, for the opportunity to pursue my Ph.D. A special thanks to Xiaodong and Geir for the fruitful discussions and all the help and support. I am also grateful for all my colleagues at NORCE that made the journey easier and smoother, especially those in the DIGIRES project and the other Ph.D.s and post-docs. A special thanks to Tuhin and Gilson for the interesting discussions and the collaboration in the papers of this thesis.

I cannot forget about my family in Brazil who has always support me in all challenges I have faced in my life. Mãe, pai e Lela, obrigado por todo amor e apoio. Essa jornada não seria possível sem vocês. Família Vasconcellos e Soares, não tenho palavras pra descrever e agradecer tudo que vocês fazem e fizeram por mim. Muito obrigado.

Many thanks also to my friends. Those in Brazil and different parts of the world who have always been by my side and supporting me, and those in Bergen who have become my family in Norway. Thank you very much.

Ricardo Vasconcellos Soares
Bergen, April 2021

Abstract

History matching is a valuable process to obtain better models and a more reliable forecast in oil reservoir applications. In such processes, one conditions the simulation models on the observed data from the field. One can formulate history matching as an inverse problem, where one has the final answer (i.e., the observed data from the field) and wants to update the model parameters that lead to this answer. There are different methods to perform this update, and among them, ensemble-based methods are state-of-the-art.

Many different research fields use ensemble-based methods, such as oil reservoirs, oceanography, and meteorology. They can represent the models' uncertainties and solve the history matching problem through an ensemble of models. Iterative ensemble-based methods have been more commonly used in oil reservoir applications because of the problem's strong nonlinearity.

The development of new sensors and new technologies have enabled conditioning models on distinct data types. Moreover, the increase in computational power has allowed dealing with larger data sets. Nevertheless, the use of these vast data sets still faces some problems. Depending on how one implements the ensemble methods, they can generate enormous matrices that are impractical to handle and manipulate.

Furthermore, ensemble-based methods might underestimate the variability after history matching because of a limited ensemble size and simplifications in the measurement errors. Therefore, one could use a localization technique to compensate for this problem. However, depending on the type of data used to condition the models, localization can be challenging.

One common data set that has gained much importance lately is the 4D seismic. On the one hand, such a data set brings information about pressure and saturation changes throughout the reservoir. On the other hand, 4D data sets can be immense and incur additional issues to the history-matching process. This thesis focuses on conditioning oil-reservoir models on such vast data sets.

Firstly, we investigate a Dictionary Learning method, the K-SVD, to represent the 4D seismic data set sparsely. In this part, we focus on reducing the data-set size without losing its main information. Besides, we also investigate a way of optimizing the computational cost of the data reduction algorithm.

Secondly, we use the sparsely represented data in an ensemble-based history matching process. In this work, we develop a new implementation of the Iterative Ensemble Smoother Regularized Levenberg–Marquardt. We also compare two ways of dealing with the vast size of the 4D seismic, one with the 4D seismic after applying the K-SVD algorithm and one with the complete data set projected into the ensemble subspace. Using Amplitude versus Angle as the seismic data and a benchmark case, the results show the importance and improvement obtained when using the sparse data.

Furthermore, we also investigate a correlation-based local analysis scheme which allows us to introduce more information from the measurement into the model and avoid variability underestimation. An advantage of local analysis is the ability to partition the models and data sets and update the model using smaller parts, which is suitable for large models and data sets. We also discuss local analysis in a 4D-seismic history matching using a different ensemble method, the Subspace Ensemble Randomized Maximum Likelihood.

Finally, Dictionary Learning's use to sparsely represent a vast data set, and the correlation-based local analysis scheme helps to deal with the big size of the observed data and generate a good representation of the reservoir.

Outline

This thesis consists of two parts. Part I introduces the scientific background, followed by the scientific results presented in Part II.

Part I consists of six chapters. Chapter 1 introduces history matching and its challenges regarding the use of vast data sets and provides the motivation and the main contribution of this thesis. Chapter 2 gives an overview of the history matching process in the oil industry and details two iterative ensemble smoother methods, the Iterative Ensemble Smoother Regularized Levenberg–Marquardt (IES-RLM) and the Subspace Ensemble Randomized Maximum Likelihood (SEnRML), with a focus on how to handle vast data sets. In Chapter 3, we describe different localization methods used in history matching. Chapter 4 provides the background theory on sparse representation methods and Dictionary Learning. Chapter 5 gives a brief description of 4D seismic data and how to use it in history matching. Finally, Chapter 6 summarizes the scientific results in Part II and presents an outlook for future research.

Part II presents the scientific results through a collection of four papers:

A Soares, R.V., Luo, X., Evensen, G., *Sparse Representation of 4D Seismic Signal Bassed on Dictionary Learning*, in SPE Norway One Day Seminar, 2019. SPE-495599-MS.

B Soares, R.V., Luo, X., Evensen, G., Bhakta, T., *4D Seismic History Matching: Assessing the use of a Dictionary Learning Based Sparse Representation Method*, Journal of Petroleum Science and Engineering. 195 (2020).

C Soares, R.V., Luo, X., Evensen, G., Bhakta, T., *Handling Big Models and Big Data Sets in History Matching Through Local Analysis*, SPE Journal (2020). SPE-204221-PA.

D Neto, G.M.S., Soares, R.V., Evensen, G., Davolio, A., Schiozer, D.J., *Subspace Ensemble Randomized Maximum Likelihood with Local Analysis for Time-Lapse-Seismic-Data Assimilation*, SPE Journal (2021). SPE-205029-PA.

Contents

Part I

Scientific background

Chapter 1

Introduction

Oil reservoir production forecasts play a vital role in the oil and gas industry. With the correct production forecast, companies can have a better knowledge of how much the reservoir will produce and the company's future economy. Consequently, they can be better prepared for future challenges.

The production forecasts are usually based on simulation models that can provide oil, gas, and water production. Nevertheless, there are many uncertainties and simplifications involved in the reservoir models and their simulations, which generates an uncertain forecast. These uncertainties can have a big impact on how one would develop and manage the reservoir. Therefore, there has been a lot of effort from the industry and academia to handle them. In this chapter, we first start with state of the art in reservoir development and management, followed by the motivation and this work's main contribution.

1.1 State of the art in reservoir development and management

Oil reservoir simulation models are used to predict oil, gas and water production from a field. However, the process of constructing the models and run the simulations involves a large amount of uncertainties. Among them, it is possible to point out porosity and permeability values throughout the reservoir, faults location and transmissibility, relative permeability curves, to name a few. Besides, the phenomena that occurs in the reservoir and the reservoir characteristics are complex. Consequently, there are some simplifications to make the process more feasible. For instance, we can highlight the use of coarse models, fluid characterization considering only one component (black-oil simulators), and not considering more complex phenomena such as hysteresis.

A common practice within the industry to develop and manage an oil field is to apply the closed-loop workflow [54] as shown in Fig. 1.1 and use system models to simulate the response of a field. System models are also referred to simulation models that are constructed using different types of data, such as geology, seismic, and well logs and tests. Note that system models include reservoir models and other parts of the system, such as production facilities and wells. However, in this work, we focus only on the reservoir simulation models as a part of the system.

In the scheme presented by Fig. 1.1, one can get the observed data (measured output) from the field (system), including well production rates and pressure and seismic, through sensors. Then, one should use them together with history-matching algorithms to condition the system models on those data [78, 1, 3, 19, 30]. The main goal of

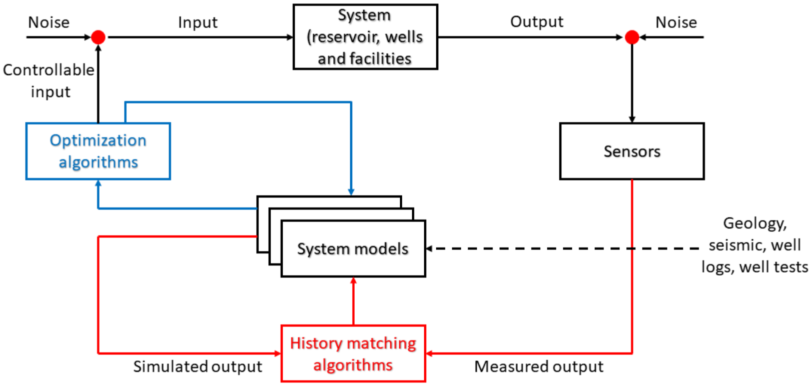


Figure 1.1: Closed-Loop Reservoir Development and Management [54].

history matching is to update the models so that the simulated output can be as close as possible to the measured (observed) output. Hence, one can use the information in the observed data to reduce the system models' uncertainties and obtain a better forecast and representation of the system.

Afterward, one can use these better models to optimize field development and management. The optimization can be based on several different functions, such as oil production or net present value, and there are many variables that one can control to obtain an optimized solution, such as drilling schedule, well position, and injection rate [112, 12, 46]. For the optimization process, it is crucial to have a good representation of the reservoir characteristics and behavior since one might need to define where to place a well and how to control it to avoid early water breakthroughs. Thus, one should also take into consideration the geological features of the reservoir.

Over time, the reservoir continues to produce, and more data become available. As a result, one can perform history matching again to generate better models and a better optimization strategy. As the name suggests, this process is a closed-loop where one will continuously update the reservoir models and the selected strategy [101, 13, 54].

This work focus on history matching, which can be formulated as an inverse problem where one approximately knows the answer (well rates, for instance) and wants to find the parameters' values that lead to this answer. In other words, one wants to obtain uncertain parameter values (porosity maps, fault transmissibility) that lead to simulated results as close as possible to the observed data from the field.

Once the reservoir starts to produce, usually simulated data obtained from the reservoir simulation do not follow the observed data, such as oil rates from the producing wells. Therefore, one performs history matching to condition the models on the observed data. Nevertheless, it is not only the match between simulated and observed data one should be interested in. One should also search for reservoir models that are geologically consistent to obtain the reservoir's best possible representation. Achieving both a match between simulated and observed data and consistent reservoir geology is a challenging task but essential for a good field development [85, 100, 11].

Ensemble methods have become a robust and powerful tool to solve history matching problems, mainly because of their ability to characterize and represent the uncertainties and also their relatively easiness to be implemented [37, 2]. Through a prior ensemble of models, ensemble methods allows us to consider different realizations with distinct values of permeability and porosity, for instance. Consequently, one will not have only one model representing the reservoir behavior; one will have an ensemble of models and the uncertainty quantification. Among the existing algorithms, iterative ensemble-based methods can be considered state of the art, and they have been extensively used within academia and industry [33, 18, 75, 42].

In this work, the focus is on ensemble methods to perform history matching and obtain more reliable reservoir models to be used for optimization. Note that different challenges may arise when using this type of algorithms. Among them, the type and size of the observed data used to condition the models are evident nowadays.

1.2 Motivation

Traditionally, most works about history matching have used well production rates and pressure as observed data [104, 30, 93, 103]. Production well data are relatively easy to gather and they are more frequent in time and less frequent in space, i.e., well data are from one fixed point, and one can get them at different time instances (daily or monthly). More recently, the development of modern sensors allowed us to monitor reservoirs through different and more complex types of field data. One example of such data is the subsurface information derived from seismic surveys. Seismic data are acquired through acoustic waves generated in the surface that travels into the subsurface. At each interface, part of this wave reflects, and it is recorded at the surface. Hence, 3D Seismic (3DS) can help to find the most likely locations of the reservoirs and their main structures (faults and cap rocks), and it has been extensively used for the constructions of reservoir simulation models [27, 50, 45]

Furthermore, seismic data are also capable of distinguishing media filled with different types of fluids and under distinct pressure conditions. Therefore, if one takes two seismic surveys of the reservoir at two consecutive times, one might be able to identify saturation and pressure changes in the reservoir due to production of oil and gas and injection of water and gas. The inclusion of the time-lapse effect in seismic is also known as 4D seismic (4DS) [55, 67]. Permanent-reservoir-monitoring (PRM) is one example of such sensor systems that have gained attention [105]. They are installed on the seabed (in an offshore case), and through frequent acquisitions of seismic signals, they can capture dynamic information about the reservoir that occurs with the production from the wells. Besides, PRM generates vast data sets due to several acquisitions along the time.

Different from well data, seismic data are not frequent in time, but they are much denser in space [55]. Therefore, the combination of these two types of data can assist the history-matching process to obtain more reliable models. Nevertheless, the use of 4DS in history matching brings additional issues. Firstly, seismic data sets are much bigger compared to well rates. As a result, ensemble-based methods might face some challenges because, to directly use such methods, one needs to compute matrices that depend on the sizes of the seismic data sets. Thus, their manipulation and storage might either be impractical, or take a long time. Secondly, one can use different types of seismic data, such as Acoustic Impedance (AI) [64, 31] or Amplitude versus Angle (AVA) [70]. In the case of AVA, for instance, some parts of the data sets correspond to non-reservoir sections (under- and overburden), and they would not be useful for the history-matching process.

To deal with these characteristics, sparse data representation methods used for dimensionality-reduction problems have gained much attraction recently [73, 63]. Many research areas are developing effective methods to reduce the dimension of big data sets while still preserving the main information of the data. Among these methods, one can highlight those from the machine learning community, such as clustering [10] and deep learning [63], and also dictionary learning [4].

Dictionary learning methods use linear combinations of pre-selected basis functions to provide a compact representation of the original data set [76]. Besides, one can learn a better dictionary from the data to further reduce the sparsely represented data as well as retaining more information about the data. The K-SVD [4] is one of the dictionary learning algorithms that have been used for many different applications, including image denoising [29], face recognition [114], and parameterization of reservoir facies [62].

Nevertheless, the use of a sparsely represented data brings another challenge. Most of the successful applications of ensemble-based methods reported the use of a localization technique to circumvent the problem of underestimation of the variability and spurious correlation due to the limited size of the ensemble [19, 30, 69, 104, 2]. The most common method of localization is distance-dependent, in which one needs to select a region around each observed data that should be influenced by the model parameters. On the one hand, this process works fine with well data, as they are in the same physical domain as the simulation model. On the other hand, for sparsely represented data, the distance-dependent approach based on physical distance cannot be applied because the measured data are in a different domain now. Hence, an alternative for distance-based localization, such as a correlation-based approach [69, 37], should be used.

Therefore, this thesis aims to develop a methodology capable of dealing with these issues arising in big data history matching problems. In other words, this work's objective is to reduce the observed data's size to help manipulate and store the matrices formed during history matching while keeping the main information about the signal. It is worth mentioning that the focus of this work is on developing a generic workflow to handle vast data sets in the history-matching process, while the main application of the developed workflow herein is to 4D seismic history matching problems.

In Chapters 2 and 3, we address history matching, ensemble methods and localization. Besides, we show how vast data sets can impact the computation of such methods and some alternatives to deal with it. In Chapter 4 we introduce another alternative to deal with the large size of the observed data sets through sparse representation methods. Chapter 5 presents one example of vast data sets in history matching processes, 4D seismic, and how one can use it. Finally, we present an overview and an introduction to the Scientific results in the second part of this work in Chapter 6.

1.3 Main contributions

The main contributions of the thesis are:

Developing an optimal approach for sparse representation of seismic data using dictionary learning. Paper A presents detailed information about the K-SVD algorithm. In this work, we evaluated the best way to apply the algorithm to a seismic data set with three main objectives: reduction of the data size; preserving the main characteristics of the signal; and reducing the computational time to apply the algorithm. By evaluating how the main parameters of the K-SVD algorithm affect the data set compression, the quality of the retained information, and the computational time spent, we developed an optimal application of the dictionary learning method for seismic data sets.

Development of computationally efficient ensemble methods for conditioning on vast data sets. In Paper B, we reformulated the Iterative Ensemble Smoother Regularized Levenberg-Marquardt (IES-RML) [75] method to avoid forming and inverting big matrices. Moreover, we compared two approaches for history matching, one using the sparsely represented data obtained in Paper A as the measured data and one with the full data set projected onto the ensemble subspace. We also developed and compared an approach where we denoised the original seismic data by using the same K-SVD algorithm from Paper A. However, as for this case the objective is denoising and not reducing the data set size, the way we applied it was not the same.

Developing localization approaches in ensemble-based methods to deal with both big data sets and big models. In Paper C and D, we developed a new local analysis scheme, a type of localization technique, that deals with big models and big data sets much more efficiently by reducing the sizes of the matrices during the history matching. We used two different ensemble-based methods (IES-RML [75] and the Subspace Ensemble Randomized Maximum Likelihood - SEnRML [92, 42]) that required different methodologies for the application of the local analysis. Paper C presents the case for the IES-RML, and Paper D shows the case for the SEnRML.

Chapter 2

History matching and ensemble methods

Numerical reservoir simulation is a fundamental topic for the reservoir community. Most oil companies use simulations to predict the reservoir behavior and production rates, which are inputs to decision-making processes concerning the management and development of petroleum fields.

To perform reservoir model simulations, one needs to know some reservoir characteristics to construct the models and simulate them. The process of building the models is better known in the literature as reservoir characterization. Nevertheless, reservoir models contain many associated uncertainties and, consequently, the predictions from these models are also uncertain.

As a reservoir starts to produce, one can use dynamic data from it, such as production rates and 4D seismic data, to update the model parameters and their uncertainties, such that one can have a more realistic representation of the reservoir and a better production forecast for the field. This process is better known as history matching in the literature. Oliver and Chen [86] presented a review on history matching, and they pointed out different algorithms to solve this problem, such as evolutionary algorithms, neighborhood-based approaches, and adjoint-/gradient-based methods. The focus here is on the ensemble-based methods, which can be considered as certain approximate gradient-based methods. They have gained much attention from researchers and practitioners in the oil industry, especially with the recent developments of iterative versions of the ensemble methods [33, 17, 18, 75].

This chapter starts from introducing basic concepts about reservoir characterization and history matching, followed by the introduction of ensemble-based methods and their iterative forms. Afterwards, we present the two different ensemble-based methods used in this work, focusing on how vast data sets can impact the computation of the methods.

2.1 An overview of reservoir characterization and history matching

The reservoir characterization process starts with the construction of geological models, which has a detailed description of the reservoir geology. It is necessary to acquire data that contain information about the reservoir characteristics to build the models. These data can include well logging, depositional models, outcrops, cores, 3D seismic, among others. They have a crucial role in the characterization since it is possible to infer the most important reservoir parameters from them, such as reservoir structure, rock type, porosity, permeability, and fluid type. Some of these data, such as cores and well logging, come from specific wells that are apart from each other. Hence, one also uses geostatistical methods and well testing to infer the characteristics of the reservoir in the regions that one does not have direct information.

Geological models are usually very refined and big models that capture the critical features of the reservoir. However, it is unfeasible to simulate these models to predict oil production because it would take significant computational time. Hence, a common practice is to upscale the geological model into a coarser model to run the simulations. From this point, the models are referred to as simulation models instead of geological models.

Commonly, simulation models do not follow the dynamic behavior of the reservoir due to the uncertainties and simplifications. Thus, one should condition the simulation models on the observed data from the field. This model conditioning process is also known as history matching, and it allows the update of the reservoir simulation models according to the observed data.

History matching is an inverse problem where one has the final answer and wishes to find the parameters that lead to this answer. In reservoir applications, the final answer is usually represented by production and injected rates from the wells and seismic, while the parameters are porosity and permeability maps, relative permeability curves, fault multipliers, among others. Note that the simulation models have many parameters, but one should update only those that are uncertain and have a significant impact on the simulation response.

If one considers $\mathbf{d}$ as a vector containing the observed data from the field and $\mathbf{g}$ as the function representing the forward reservoir simulation, the history-matching goal is to find the uncertain model parameters $\mathbf{x}$ that lead to $\mathbf{d}^{\mathrm{sim}}$, as in

$$\mathbf{d}^{\mathrm{sim}} = \mathbf{g}(\mathbf{x}), \tag{2.1}$$

as close as possible to $\mathbf{d}$.

The most common type of data used as observed data is the production rates from the wells [104, 30, 93, 103]. On the one hand, these data are frequent in time. On the other hand, they are not dense in space. Another type of data that is becoming more and more popular within history-matching problems is the seismic data [31, 64, 23]. Through seismic acquisitions at different times, one can observe the effect of reservoir pressure and saturation changes. Different from well data, seismic data are not frequent in time, but they are dense in space. Hence, production and seismic data can complement each other in history-matching processes.

Nevertheless, there are some challenges when one uses seismic data, such as data set size, processing, and interpretability. We provide more information about seismic in Chapter 5. For now, it suffices to say that seismic data sets are much bigger than well production data sets. In [73], the authors conducted a history matching experiment using both well and seismic data in the Brugge benchmark case [91]. In their study, the production data included 1 400 data points, while the seismic contained more than 7 million data points. Thus, this could bring computational issues regarding both storage and CPU time.

One important aspect of history-matching problems is its ill-poseness, i.e., the number of independent observed data is much smaller than the number of model uncertain parameters. Therefore, there are infinite combinations of the parameters that can generate the same final answer. By inserting a regularization term in the cost function through prior information from the reservoir, it is possible to restrict the space in which one is looking for the solution. Besides, the introduction of prior information about the reservoir can assist in generating models more geologically consistent.

Traditionally, manual history matching approaches were used to update the parameters in the reservoir models [3, 113]. However, manual history matching requires much experience from those who perform it. Also, it can destroy the models' geological consistency, and it is not easy to match all of the production data.

In addition, history matching problems have many solutions, and one will never know which one is the true one. Hence, it is desirable to consider several models to represent the reservoir by using a probabilistic approach where it is possible to quantify the uncertainties. As mentioned previously, there are different methods to solve history matching problems, and, in this work, the focus is on the ensemble-based methods. Such methods are popular because of their ability to quantify the uncertainties, easiness of implementation and its interesting results reported by several works [79, 19, 42, 30, 66, 102].

2.2 Ensemble-based methods

The development of ensemble-based methods started with Evensen [35] when he proposed the Ensemble Kalman Filter (EnKF), which was first used in petroleum applications in [79]. EnKF considers an ensemble of realizations to describe the uncertainties in the model and updates these uncertainties sequentially in time as new observations become available. However, there is a need to restart the simulations at each update step, and this can increase the computational time significantly. Alternatively, van Leeuwen and Evensen [111] proposed a variation of the EnKF that uses all data simultaneously, also known as the Ensemble Smoother (ES). Skjervheim et al. [103] used ES in a reservoir application for the first time.

Before introducing the formulations of the ES, we first present some basic concepts. As mentioned in Eq. (2.1), the simulated data can be obtained through the simulator and input model variables. Nevertheless, there are some errors related to the data and also to the forward simulator. Hence, one can define the observed data $\mathbf{d}$ as

$$\mathbf{d} = \mathbf{g}^{\text{true}}(\mathbf{x}^{\text{true}}) + \delta. \tag{2.2}$$

Here, δ is measurement error where $\delta \sim \mathcal{N}(0, \mathbf{C}_{\text{d}})$, $\mathbf{C}_{\text{d}}$ is the measurement-error covariance matrix, and the superscript true indicates a perfect forward simulator and the true value for the uncertainties in $\mathbf{x}^{\text{true}}$. After performing simulations of the ensemble, one would achieve the simulated data as in Eq. (2.1). Note that now $\mathbf{g}$ may be an imperfect forward simulator, such that one needs to consider model errors in the formulation. However, we will first focus on the case where one has a perfect forward simulator, and we will provide more discussions on model errors later in this chapter.

In order to derive the ES formulation, one should consider Bayes' formulation

$$p(\mathbf{x}|\mathbf{d}) \propto p(\mathbf{x})p(\mathbf{d}|\mathbf{x}), \tag{2.3}$$

where $p(\mathbf{x}|\mathbf{d})$ is the posterior distribution of $\mathbf{x}$ given $\mathbf{d}$, $p(\mathbf{x})$ is the prior distribution of the uncertain model parameters, and $p(\mathbf{d}|\mathbf{x})$ is the likelihood of $\mathbf{d}$ given $\mathbf{x}$. If one considers a Gaussian distribution of both prior and likelihood, one can approximately sample $p(\mathbf{x}|\mathbf{d})$ by defining the following cost function

$$\text{argmin}_{\{\mathbf{x}_j\}}\left\{\left(\mathbf{g}(\mathbf{x}_j)-\mathbf{d}_j\right)^{\mathrm{T}}\mathbf{C}_{\mathrm{d}}^{-1}\left(\mathbf{g}(\mathbf{x}_j)-\mathbf{d}_j\right)+\left(\mathbf{x}_j-\mathbf{x}_j^0\right)^{\mathrm{T}}\mathbf{C}_{\mathrm{x}}^{-1}\left(\mathbf{x}_j-\mathbf{x}_j^0\right)\right\}. \tag{2.4}$$

Here j is the index of ensemble member, $\mathbf{x}^0$ is the prior distribution of the uncertain model parameters, and $\mathbf{C}_{\mathrm{x}}$ is the error covariance matrix of $\mathbf{x}$. Note that $\mathbf{x}_j^0$ is sampled from $\mathcal{N}(\mathbf{x}^0, \mathbf{C}_{\mathrm{x}})$ and $\mathbf{d}_j$ from $\mathcal{N}(\mathbf{d}, \mathbf{C}_{\mathrm{d}})$.

Evensen [42, 40] showed a detailed derivation of the ES formulation, and we follow a similar procedure in this work. Firstly, one should compute the gradient of the cost function to minimize it,

$$\frac{\partial J(\mathbf{x}_j)}{\partial \mathbf{x}_j}=\mathbf{G}_j^{\mathrm{T}}\mathbf{C}_{\mathrm{d}}^{-1}\left(\mathbf{g}(\mathbf{x}_j)-\mathbf{d}_j\right)+\mathbf{C}_{\mathrm{x}}^{-1}\left(\mathbf{x}_j-\mathbf{x}_j^0\right), \tag{2.5}$$

where J is the cost function and $\mathbf{G}_j \in \mathbb{R}^{N_{\mathrm{d}}\times N_{\mathrm{x}}}$ is the Jacobian matrix. Here N_{x} and N_{d} represent the number of uncertain model parameters and observed data. $\mathbf{G}_j$ can be defined as

$$\mathbf{G}_j=\left(\nabla_{\mathbf{x}}\mathbf{g}(\mathbf{x})|_{\mathbf{x}=\mathbf{x}_j}\right)^{\mathrm{T}} \tag{2.6}$$

If one equals Eq. (2.5) to zero, one obtains

$$\mathbf{G}_j^{\mathrm{T}}\mathbf{C}_{\mathrm{d}}^{-1}\left(\mathbf{g}(\mathbf{x}_j)-\mathbf{d}_j\right)+\mathbf{C}_{\mathrm{x}}^{-1}\left(\mathbf{x}_j-\mathbf{x}_j^0\right)=0. \tag{2.7}$$

Eq. (2.7) does not have an explicit solution. One alternative is to linearize $\mathbf{g}(\cdot)$ around $\mathbf{x}^0$

$$\mathbf{g}(\mathbf{x}_j)\approx\mathbf{g}(\mathbf{x}_j^0)+\mathbf{G}_j^0(\mathbf{x}_j-\mathbf{x}_j^0), \tag{2.8}$$

$$\mathbf{G}_j\approx\mathbf{G}_j^0+\mathbf{H}_j^0(\mathbf{x}_j-\mathbf{x}_j^0). \tag{2.9}$$

$\mathbf{G}_j^0$ is the Jacobian evaluated at $\mathbf{x}_j^0$ and $\mathbf{H}_j^0$ is the Hessian matrix. A common procedure is to disregard the Hessian. Following, if one substitutes Eq. (2.8) and (2.9) in Eq. (2.7), one will get

$$(\mathbf{G}_j^0)^{\mathrm{T}}\mathbf{C}_{\mathrm{d}}^{-1}\left(\mathbf{g}(\mathbf{x}_j^0)+\mathbf{G}_j^0(\mathbf{x}_j-\mathbf{x}_j^0)-\mathbf{d}_j\right)+\mathbf{C}_{\mathrm{x}}^{-1}\left(\mathbf{x}_j-\mathbf{x}_j^0\right)=0. \tag{2.10}$$

Also, one can rewrite Eq. (2.10) as

$$\left(\mathbf{C}_{\mathrm{x}}^{-1}+(\mathbf{G}_j^0)^{\mathrm{T}}\mathbf{C}_{\mathrm{d}}^{-1}\mathbf{G}_j^0\right)\left(\mathbf{x}_j-\mathbf{x}_j^0\right)=(\mathbf{G}_j^0)^{\mathrm{T}}\mathbf{C}_{\mathrm{d}}^{-1}\left(\mathbf{d}_j-\mathbf{g}(\mathbf{x}_j^0)\right). \tag{2.11}$$

Finally, one can use the following identity matrix [40]

$$\left(\mathbf{C}^{-1}+\mathbf{G}^{\mathrm{T}}\mathbf{D}^{-1}\mathbf{G}\right)\mathbf{G}^{\mathrm{T}}\mathbf{D}^{-1}=\mathbf{C}\mathbf{G}^{\mathrm{T}}\left(\mathbf{G}\mathbf{C}\mathbf{G}^{\mathrm{T}}+\mathbf{D}\right)^{-1}, \tag{2.12}$$

and update the uncertain model parameter for the realization j as in

$$\mathbf{x}_j=\mathbf{x}_j^0+\mathbf{C}_{\mathrm{x}}(\mathbf{G}_j^0)^{\mathrm{T}}\left(\mathbf{G}_j^0\mathbf{C}_{\mathrm{x}}(\mathbf{G}_j^0)^{\mathrm{T}}+\mathbf{C}_{\mathrm{d}}\right)^{-1}\left(\mathbf{d}_j-\mathbf{g}(\mathbf{x}_j^0)\right). \tag{2.13}$$

Eq. (2.13) is the analytical solution of ES [111]. If $\mathbf{g}$ is linear, the equation samples the correct posterior distribution. For a nonlinear case, this solution is an approximation that would be valid only for small updates.

In addition, depending on the format of $\mathbf{g}$, the computation of the gradient might not be straightforward, especially in cases of a very non-linear function such as a reservoir simulator. Therefore, Evensen [35, 39, 40] considered an ensemble of realizations to compute the gradient by determining the covariances $\mathbf{AS}^\mathrm{T}$ and $\mathbf{SS}^\mathrm{T}$

$$\mathbf{AS}^\mathrm{T} = \mathbf{C}_\mathrm{x}(\mathbf{G}_j^0)^\mathrm{T}, \tag{2.14}$$

$$\mathbf{SS}^\mathrm{T} = \mathbf{G}_j^0\mathbf{C}_\mathrm{x}(\mathbf{G}_j^0)^T, \tag{2.15}$$

. To compute $\mathbf{S}$ and $\mathbf{A}$, one should use the following equations

$$\mathbf{S} = \frac{1}{\sqrt{N_\mathrm{e} - 1}}\Big(\mathbf{g}\big(\mathbf{x}_1^0\big) - \overline{\mathbf{g}\big(\mathbf{x}^0\big)}, ..., \mathbf{g}\big(\mathbf{x}_{N_\mathrm{e}}^0\big) - \overline{\mathbf{g}\big(\mathbf{x}^0\big)}\Big), \tag{2.16}$$

and

$$\mathbf{A} = \frac{1}{\sqrt{N_\mathrm{e} - 1}}\Big(\mathbf{x}_1^0 - \overline{\mathbf{x}^0}, ..., \mathbf{x}_{N_\mathrm{e}}^0 - \overline{\mathbf{x}^0}\Big), \tag{2.17}$$

where N_e is the total number of realizations in the ensemble and the overbar represents the mean value regarding the ensemble. Evensen [40] pointed out that these equations are valid for most of the history matching applications, where the rank of $\mathbf{A}$ is $N_\mathrm{e} - 1$ (i.e., $N_\mathrm{x} > N_\mathrm{e} - 1$).

Finally, one can write the formula to update the uncertain parameters in the ES as

$$\mathbf{x}_j = \mathbf{x}_j^0 + \mathbf{AS}^\mathrm{T}\Big(\mathbf{SS}^\mathrm{T} + \mathbf{C}_\mathrm{d}\Big)^{-1}\Big(\mathbf{d}_j - \mathbf{g}(\mathbf{x}_j^0)\Big). \tag{2.18}$$

Note that in Eq. (2.13), one computes the model sensitivity $\mathbf{C}_\mathrm{x}\big(\mathbf{G}_j^0\big)^\mathrm{T}$ for each model, and in Eq. (2.18), one computes the average model sensitivity $\mathbf{AS}^\mathrm{T}$ related to the ensemble instead.

If $\mathbf{g}$ is linear and N_e tends to infinity, Eq. (2.18) will sample the correct posterior distribution. However, if these conditions are not satisfied, as in reservoir applications, one will have an approximate solution. Such approximations are due to the linearization in Eqs. (2.8) and (2.9), the introduction of the ensemble gradient, and the limit size of the ensemble. For weakly nonlinear problems, the linearization is the most severe approximation. However, one alternative to deal with it is to use iterative forms of the ensemble smoother [39].

2.3 Iterative ensemble smoothers

Iterative ensemble smoothers (IES) have became very popular in the reservoir community. Among the different methods, it is possible to point out the Ensemble Randomized Maximum Likelihood (EnRML) [17, 18], the Ensemble Smoother with Multiple Data Assimilation (ESMDA) [33], the Subspace EnRML (SEnRML) [92, 42], and the Iterative Ensemble Smoother Regularized Levenberg–Marquardt (IES-RLM) [75].

All the different methods solve slightly different problems, but the main idea behind them is to perform small updates at each iteration instead of performing a big update at one time as in the ES. In the following, we discuss the IES-RLM and the SEnRML.

2.3.1 IES-RLM

The cost function used in the IES-RLM [75] is a minimum-average-cost function, and it can be defined as

$$\begin{aligned}\operatorname{argmin}_{\{\mathbf{x}_j^{i+1}\}} \frac{1}{N_\mathrm{e}} \sum_{j=1}^{N_\mathrm{e}} \bigg\{ & \left(\mathbf{d}_j - \mathbf{g}(\mathbf{x}_j^{i+1})\right)^\mathrm{T} \mathbf{C}_\mathrm{d}^{-1} \left(\mathbf{d}_j - \mathbf{g}(\mathbf{x}_j^{i+1})\right) + \\ & \alpha^i \left(\mathbf{x}_j^{i+1} - \mathbf{x}_j^i\right)^\mathrm{T} \mathbf{C}_\mathrm{x}^{-1} \left(\mathbf{x}_j^{i+1} - \mathbf{x}_j^i\right) \bigg\},\end{aligned} \tag{2.19}$$

where i is the current iteration number and N_i is the maximum number of iterations. The cost function here follows the same idea as the one presented in Eq. (2.4), in which the first term calculates the difference between the observed and simulated data, and now the second term corresponds to the difference between the uncertain model parameters at two consecutive iterations. Besides, the authors also added the coefficient α, which assigns a weight to the second term of the equation.

It is important to mention that, different from the other IESs, this formulation does not sample the posterior distribution from Bayes' theorem. This approach is an optimization problem with a regularization term, and one solves this cost-function several times until a stopping criterion is reached.

To solve this problem, Luo et al. [75] used a similar approach as in the development of the ES. But instead of linearizing $\mathbf{g}(\cdot)$ around the initial ensemble, they did it around the mean $\overline{\mathbf{x}^i}$ as in

$$\mathbf{g}\left(\mathbf{x}_j^{i+1}\right) \approx \mathbf{g}\left(\overline{\mathbf{x}^i}\right) + \mathbf{G}_j\left(\mathbf{x}_j^{i+1} - \overline{\mathbf{x}^i}\right). \tag{2.20}$$

Note that now, $\mathbf{G}_j$ is the jacobian of $\mathbf{g}$ at $\overline{\mathbf{x}^i}$.

After applying a similar algebraic procedure as in the previous section, the authors achieved the following update formula for the uncertain model parameters

$$\mathbf{x}_j^{i+1} = \mathbf{x}_j^i + \mathbf{A}^i(\hat{\mathbf{S}}^i)^\mathrm{T}\left(\hat{\mathbf{S}}^i(\hat{\mathbf{S}}^i)^\mathrm{T} + \alpha^i\mathbf{C}_\mathrm{d}\right)^{-1}\left(\mathbf{d}_j - \mathbf{g}(\mathbf{x}_j^i)\right), \tag{2.21}$$

where $\mathbf{A}^i$ is calculated as in Eq. (2.17) at each iteration i and $\hat{\mathbf{S}}^i$ is calculated as

$$\hat{\mathbf{S}}^i = \frac{1}{\sqrt{N_\mathrm{e} - 1}}\left(\mathbf{g}(\mathbf{x}_1^i) - \mathbf{g}(\overline{\mathbf{x}^i}), \ldots, \mathbf{g}(\mathbf{x}_{N_\mathrm{e}}^i) - \mathbf{g}(\overline{\mathbf{x}^i})\right). \tag{2.22}$$

α is the step length of each iteration, and it can be calculated as follows

$$\alpha^i = \beta^i \times \operatorname{trace}\left(\hat{\mathbf{S}}^i(\hat{\mathbf{S}}^i)^\mathrm{T}\right) / \operatorname{trace}\left(\mathbf{C}_\mathrm{d}\right). \tag{2.23}$$

Here β is a term that should decrease if the data mismatch reduces with the iterations or increase otherwise. Luo et al. [75] suggested to start with $\beta^0 = 1$ and multiply by 0.9 (if the mismatch decreases) or 2 (if the mismatch increases).

If one considers $\mathbf{X}$ as a matrix containing all realizations (1,...,N_e), and $\mathbf{D}$ as the difference between the observed and simulated data for all realizations, Eq. (2.21) becomes

$$\mathbf{X}^{i+1} = \mathbf{X}^i + \mathbf{A}^i(\hat{\mathbf{S}}^i)^\mathrm{T}\left(\hat{\mathbf{S}}^i(\hat{\mathbf{S}}^i)^\mathrm{T} + \alpha^i\mathbf{C}_\mathrm{d}\right)^{-1}\mathbf{D}^i, \tag{2.24}$$

Table 2.1: Matrices dimension IES-RLM.

Matrix	Size
$\mathbf{X}$	$N_\mathrm{x} \times N_\mathrm{e}$
$\mathbf{A}$	$N_\mathrm{x} \times N_\mathrm{e}$
$\tilde{\mathbf{S}}$	$N_\mathrm{d} \times N_\mathrm{e}$
$\tilde{\mathbf{D}}$	$N_\mathrm{d} \times N_\mathrm{e}$
$\mathbf{C}_\mathrm{d}$	$N_\mathrm{d} \times N_\mathrm{d}$

In addition, considering that

$$\tilde{\mathbf{S}}^i = \mathbf{C}_\mathrm{d}^{-1/2}\hat{\mathbf{S}}^i \tag{2.25}$$

and

$$\tilde{\mathbf{D}}^i = \mathbf{C}_\mathrm{d}^{-1/2}\mathbf{D}^i, \tag{2.26}$$

one can obtain

$$\mathbf{X}^{i+1} = \mathbf{X}^i + \mathbf{A}^i\left(\tilde{\mathbf{S}}^i\right)^\mathrm{T}\left(\tilde{\mathbf{S}}^i\left(\tilde{\mathbf{S}}^i\right)^\mathrm{T} + \alpha^i\mathbf{I}\right)^{-1}\tilde{\mathbf{D}}^i, \tag{2.27}$$

The computation of Eq. (2.27) can be challenging because of the size that the matrices can have. Table 2.1 shows the size of each matrix one deals with during the application of the IES-RLM.

As mentioned before, simulation models are getting bigger and more observed data are becoming available. In such cases N_x and N_d are much larger than N_e. For instance, in ensemble-based history matching, the number of realizations is around 100, and N_x and N_d can reach thousand or even million. Before computing Eq. (2.27), one needs to compute Eqs. (2.25) and (2.26). The cost of computing the product between $\mathbf{C}_\mathrm{d}$ and $\mathbf{S}$ and $\mathbf{D}$ is about $N_\mathrm{d}^2 N_\mathrm{e}$, which are quadratic in the number of measurement. Hence, it can cause a big issue in the case of a large observed data set. Nevertheless, one can neglect the correlation between the measurement, and use a diagonal $\mathbf{C}_\mathrm{d}$ to avoid this problem [104, 70, 65]. In addition, the inversion of the matrix in Eq. (2.27) can be a challenge since the matrix product $\tilde{\mathbf{S}}^i\left(\tilde{\mathbf{S}}^i\right)^\mathrm{T}$ has dimension of $N_\mathrm{d} \times N_\mathrm{d}$. One alternative is to compute the singular value decomposition of $\tilde{\mathbf{S}}^i$ and truncate it, so one can use only the leading singular values. Evensen [38] suggested to keep between 90% and 99.9% of the leading coefficients.

Considering the truncated singular value decomposition (TSVD) of $\tilde{\mathbf{S}}^i$ as

$$\tilde{\mathbf{S}}^i \approx \hat{\mathbf{U}}^i\hat{\mathbf{\Sigma}}^i(\hat{\mathbf{V}}^i)^\mathrm{T}, \tag{2.28}$$

it is possible to rewrite Eq. (2.27) as

$$\mathbf{X}^{i+1} = \mathbf{X}^i + \mathbf{A}^i\hat{\mathbf{V}}^i(\hat{\mathbf{\Sigma}}^i)^\mathrm{T}\left(\hat{\mathbf{\Sigma}}^i(\hat{\mathbf{\Sigma}}^i)^\mathrm{T} + \alpha^i\mathbf{I}\right)^{-1}(\hat{\mathbf{U}}^i)^\mathrm{T}\tilde{\mathbf{D}}^i. \tag{2.29}$$

There are different ways of computing Eq. (2.29), and they can have distinct impacts concerning computational costs. In one approach, one can define the Kalman gain matrix $\mathbf{K}^i$ and write Eq. (2.29) as

$$\mathbf{X}^{i+1} = \mathbf{X}^i + \mathbf{K}^i\tilde{\mathbf{D}}^i \tag{2.30}$$

However, one drawback of this computation is the fact that $\mathbf{K}$ has a dimension of $N_\mathrm{x} \times N_\mathrm{d}$. Thus, if one is dealing with big models and big data sets, it might be

Table 2.2: Matrices dimension.

Matrix	Computational cost	Brugge example
$\mathbf{K}_{\text{eff}}$	$K + N_{\text{sv}}N_{\text{e}}(N_{\text{d}} + N_{\text{x}})$	$9.1 * 10^3 N_{\text{d}} + 1.6 * 10^9 + K$
$\mathbf{K}$	$K + N_{\text{d}}N_{\text{x}}(N_{\text{sv}} + N_{\text{e}})$	$3.4 * 10^7 N_{\text{d}} + K$

impractical to handle and manipulate this matrix. For instance, Luo et al. [73] used the Brugge benchmark, and they considered 178 200 uncertain model parameters and about 7 million observed data. Therefore, $\mathbf{K}$ would be very big, and one would need about 9 TB to store it.

One alternative is to compute the effective Kalman Gain [36, 65, 69, 20] by reordering the sequence of the matrix products. Thus, one can transform Eq. (2.29) in

$$\mathbf{X}^{i+1} = \mathbf{X}^i + \mathbf{K}^i_{\text{eff}}\Big((\hat{\mathbf{U}}^i)^{\text{T}}\tilde{\mathbf{D}}^i\Big). \tag{2.31}$$

In such cases, by first computing the product between the last two matrices, one would achieve a matrix with dimension $N_{\text{sv}} \times N_{\text{e}}$, where N_{sv} is the number of retained singular values. Note that this product is a projection of $\mathbf{D}$ (also known as the observation innovations) into the space spanned by $\mathbf{U}$, and it is a lower-dimensional representation of the observation innovations, which can help to deal with the problem of the vast data sets as in the case of seismic data. In this case, $\mathbf{K}_{\text{eff}}$ would have the size of $N_{\text{x}} \times N_{\text{sv}}$, and since $N_{\text{sv}} \ll N_{\text{d}}$, the effective Kalman Gain is easier to store and manipulate.

It is worth mentioning that one does not need to explicitly compute $\mathbf{K}$ or $\mathbf{K}_{\text{eff}}$. Instead, one can compute each row of these matrices separately [31], and consequently, avoid the big dimension of the Kalman gain matrix.

Another fundamental aspect concerning how to compute the analysis equation is about computational costs. Table 2.2 shows the cost for both types of computation ($\mathbf{K}$ and $\mathbf{K}_{\text{eff}}$). Supposing that the cost for computing $\mathbf{K}_{\text{eff}}$ is K. The cost for computing $(\hat{\mathbf{U}}^i)^{\text{T}}\tilde{\mathbf{D}}^i$ is approximately $N_{\text{sv}}N_{\text{d}}N_{\text{e}}$, and the cost of multiplying $\mathbf{K}_{\text{eff}}$ by $(\hat{\mathbf{U}}^i)^{\text{T}}\tilde{\mathbf{D}}^i$ is $N_{\text{x}}N_{\text{sv}}N_{\text{e}}$. By summing these terms up, one can achieve the final expression in the middle column in Table 2.2. On the other hand, to compute $\mathbf{K}$, one needs to first multiply $\mathbf{K}_{\text{eff}}$ by $(\hat{\mathbf{U}}^i)^{\text{T}}$, which would give $N_{\text{x}}N_{\text{sv}}N_{\text{d}}$. Subsequently, it is necessary to multiply this term by $\underset{\sim}{\mathbf{D}}^i$, resulting in $N_{\text{x}}N_{\text{d}}N_{\text{e}}$. Summing up these parts with K during the calculation of $\mathbf{K}_{\text{eff}}$, one would achieve what is shown in the table above. Note that we are not considering previous computations such as in the Eqs. (2.25) and (2.26).

To illustrate this, we will consider the Brugge benchmark case used in [73], where $N_{\text{x}} = 178\,200$, $N_{\text{e}} = 103$, and $N_{\text{sv}} = 88$, which corresponds to a 90% of the energy retained. The third column of Table 2.2 shows computational cost for this example. The computational cost for calculating $\mathbf{K}$ is larger than calculating $\mathbf{K}_{\text{eff}}$ when N_{d} is approximately larger than 47. The left plot in Fig. 2.1 illustrates the behavior of the computational cost for different numbers of observed data, and as one can notice, the case for $\mathbf{K}_{\text{eff}}$ almost does not change compared to $\mathbf{K}$, while the case for $\mathbf{K}$ increases the computational cost substantially with the increase in N_{d}.

In addition to the computational cost, the right plot in Fig. 2.1 shows that the memory requirement for $\mathbf{K}$ increases linearly and it can reach almost 150GB for 100 000 data points, while for $\mathbf{K}_{\text{eff}}$, the requirement remains low and almost constant (0.13 GB) within the considered interval. Note that $\mathbf{K}$ does not need to be fully computed, one can compute each row of it separately. However, one would need to do it several times,

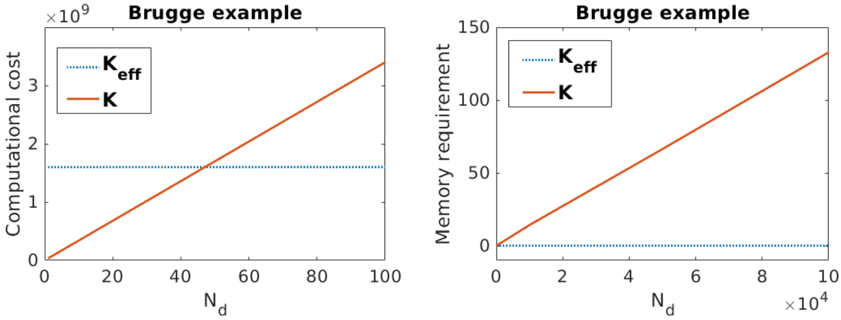


Figure 2.1: Computational cost (left) and memory requirement (right) for the Brugge benchmark case.

which can increase the computational time.

Therefore, if one is facing a history-matching problem, the approach using $\mathbf{K}_{\text{eff}}$ seems to be more effective in handling large data sets. An alternative is to reduce N_{d} through sparse representation or machine learning methods.

2.3.2 SEnRML

Differently from [75], Chen and Oliver [17, 18] minimize the same cost function of the ES method (Eq. (2.4)). The difference is that instead of solving through a linearization of g, one now uses Gauss-Newton iterations.

In [42, 92], the authors developed a new approach for the EnRML, the Subspace EnRML (SEnRML), in which they simplified the algorithm computationally. To do this, they considered that the solution to the problem is confined in the subspace spanned by the initial ensemble. Hence, they wrote the solution as

$$\mathbf{X} = \mathbf{X}^0 + \mathbf{A}\mathbf{W}. \tag{2.32}$$

In the SEnRML, the authors solved the following minimization problem

$$\text{argmin}_{\{\mathbf{w}_j\}}\left\{\frac{1}{2}\mathbf{w}_j^{\text{T}}\mathbf{w}_j + \frac{1}{2}\Big(\mathbf{g}(\mathbf{x}_j^0 + \mathbf{A}\mathbf{w}_j) - \mathbf{d}_j\Big)^{\text{T}}\mathbf{C}_{\text{d}}^{-1}\Big(\mathbf{g}(\mathbf{x}_j^0 + \mathbf{A}\mathbf{w}_j) - \mathbf{d}_j\Big)\right\}, \tag{2.33}$$

where the first part concerns the update of the model parameters and the second to the data mismatch. The minimization of Eq. (2.33) provides the exactly same solution as Eq. (2.4). The difference is that with the incorporation of Eq. (2.32) into the cost function, the solution is contained in the space spanned by the initial ensemble, and one does not need to introduce the linearization approximation.

The first and second derivative of the cost function J shown in Eq. (2.33) are

$$\frac{\partial J(\mathbf{w}_j)}{\partial \mathbf{w}_j} = \mathbf{w}_j + \big(\mathbf{G}_j\mathbf{A}\big)^{\text{T}}\mathbf{C}_{\text{d}}^{-1}(\mathbf{g}(\mathbf{x}_j^0 + \mathbf{A}\mathbf{w}_j) - \mathbf{d}_j), \tag{2.34}$$

and

$$\frac{\partial^2 J(\mathbf{x}_j)}{\partial \mathbf{w}_j^2} = \mathbf{I} + (\mathbf{G}_j\mathbf{A})^{\text{T}}\mathbf{C}_{\text{d}}^{-1}(\mathbf{G}_j\mathbf{A}), \tag{2.35}$$

respectively, where in Eq. (2.35) the term containing the second order derivatives is not considered, and $\mathbf{G}_j$ is now

$$\mathbf{G}_j = \left(\nabla_{\mathbf{x}}\mathbf{g}(\mathbf{x})|_{\mathbf{x}=\mathbf{x}_j^0+\mathbf{A}\mathbf{w}_j}\right)^{\mathrm{T}} \tag{2.36}$$

Then, if one uses the Gauss-Newton scheme as in

$$\mathbf{w}_j^{i+1} = \mathbf{w}_j^i - \gamma \frac{\frac{\partial J(\mathbf{w}_j)}{\partial \mathbf{w}_j}}{\frac{\partial^2 J(\mathbf{x}_j)}{\partial \mathbf{x}_j^2}}, \tag{2.37}$$

substitutes Eq. (2.34) and (2.35) into Eq. (2.37), includes the ensemble sensitivity matrix to replace the gradient $\mathbf{G}_j$, and apply some algebra, one can reach the following analysis equation to update the uncertain model parameters

$$\mathbf{W}^{i+1} = \mathbf{W}^i - \gamma\Big(\mathbf{W}^i - (\mathbf{P}^i)^{\mathrm{T}}\left((\mathbf{P}^i)(\mathbf{P}^i)^{\mathrm{T}} + \mathbf{C}_{\mathrm{d}}\right)^{-1}\mathbf{H}^i\Big). \tag{2.38}$$

γ is the step length and $\mathbf{P}^i$ is defined as

$$\mathbf{P}^i = \mathbf{S}^i(\mathbf{I} + \hat{\mathbf{W}}^i)^{-1}, \tag{2.39}$$

$\hat{\mathbf{W}}^i$ as

$$\hat{\mathbf{W}}^i = \frac{1}{\sqrt{N_{\mathrm{e}} - 1}}\Big(\mathbf{w}_1^i - \overline{\mathbf{w}^i}, ..., \mathbf{w}_{N_{\mathrm{e}}}^i - \overline{\mathbf{w}^i}\Big), \tag{2.40}$$

and $\mathbf{H}^i$

$$\mathbf{H}^i = \mathbf{P}^i\mathbf{W}^i + \mathbf{D}^i, \tag{2.41}$$

The readers are referred to [42, 92] for more information on the derivation.

Nevertheless, the matrix to be inverted has a dimension of $N_{\mathrm{d}} \times N_{\mathrm{d}}$, which can be impractical. One approach to deal with this is to use the Woodbury corollary [42, 92]

$$(\mathbf{I} + \mathbf{B}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{B})^{-1}\mathbf{B}^{\mathrm{T}}\mathbf{R}^{-1} = \mathbf{B}^{\mathrm{T}}(\mathbf{B}\mathbf{B}^{\mathrm{T}} + \mathbf{R})^{-1}. \tag{2.42}$$

Hence, Eq. (2.38) can be rewritten as

$$\mathbf{W}^{i+1} = \mathbf{W}^i - \gamma\Big(\mathbf{W}^i - \left(\mathbf{I} + (\mathbf{P}^i)^{\mathrm{T}}\mathbf{C}_{\mathrm{d}}^{-1}(\mathbf{P}^i)\right)^{-1}(\mathbf{P}^i)^{\mathrm{T}}\mathbf{C}_{\mathrm{d}}^{-1}\mathbf{H}^i\Big). \tag{2.43}$$

Note that now the matrix to be inverted has dimension of $N_{\mathrm{e}} \times N_{\mathrm{e}}$, which is easier than in the previous formulation.

Another alternative proposed in [37] is to use Eq. (2.38) together with the subspace algorithm. In this formulation, one needs to project the measurement error matrix onto the subspace defined by $\mathbf{P}$ as in

$$\left(\mathbf{P}\mathbf{P}^{\mathrm{T}} + \mathbf{C}_{\mathrm{d}}\right) \approx \mathbf{P}\mathbf{P}^{\mathrm{T}} + \left(\mathbf{P}\mathbf{P}^{+}\right)\mathbf{C}_{\mathrm{d}}\left(\mathbf{P}\mathbf{P}^{+}\right)^{\mathrm{T}}. \tag{2.44}$$

Note that we dropped the superscript i for simplification. By computing the singular value decomposition (SVD) of $\mathbf{P}$,

$$\mathbf{P} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\mathrm{T}} \tag{2.45}$$

and substituting in Eq. (2.44), one achieves

$$\mathbf{P}\mathbf{P}^{\mathrm{T}} + (\mathbf{P}\mathbf{P}^{+})\mathbf{C}_{\mathrm{d}}(\mathbf{P}\mathbf{P}^{+})^{\mathrm{T}} = \mathbf{U}\mathbf{\Sigma}\left(\mathbf{I} + \mathbf{\Sigma}^{+}\mathbf{U}^{\mathrm{T}}\mathbf{C}_{\mathrm{d}}\mathbf{U}(\Sigma^{+})^{\mathrm{T}}\right)\mathbf{\Sigma}^{\mathrm{T}}\mathbf{U}^{\mathrm{T}}, \tag{2.46}$$

where $\mathbf{P}^{+} = \mathbf{V}\boldsymbol{\Sigma}^{+}\mathbf{U}^{\mathrm{T}}$ [37]. Additionally, if one considers the following eigenvalue decomposition

$$\boldsymbol{\Sigma}^{+}\mathbf{U}^{\mathrm{T}}\mathbf{C}_{\mathrm{d}}\mathbf{U}(\boldsymbol{\Sigma}^{+})^{\mathrm{T}} = \mathbf{Z}\boldsymbol{\Lambda}\mathbf{Z}^{\mathrm{T}}, \tag{2.47}$$

it is possible to perform the inversion as in

$$\left(\mathbf{P}\mathbf{P}^{\mathrm{T}} + \mathbf{C}_{\mathrm{d}}\right)^{-1} \approx \left(\mathbf{U}(\boldsymbol{\Sigma}^{+})^{\mathrm{T}}\mathbf{Z}\right)\left(\mathbf{I} + \boldsymbol{\Lambda}\right)^{-1}\left(\mathbf{U}(\boldsymbol{\Sigma}^{+})^{\mathrm{T}}\mathbf{Z}\right)^{\mathrm{T}} \tag{2.48}$$

Another important point mentioned in [37] is the need to manipulate and store $\mathbf{C}_{\mathrm{d}}$, which can be very big depending on the application. Therefore, it is proposed to use a low-rank version of $\mathbf{C}_{\mathrm{d}}$ as in

$$\mathbf{C}_{\mathrm{d}} \approx \boldsymbol{\Delta}\boldsymbol{\Delta}^{\mathrm{T}}, \tag{2.49}$$

where $\boldsymbol{\Delta}$ is a matrix in the dimension of $N_{\mathrm{d}} \times N$ that contains the measurement perturbations and N is a value defined by the user. Note that it is possible to increase N to a number even higher than the number of realizations per ensemble to reduce sampling errors [41].

By using this representation, Eq. (2.46) and (2.47) becomes

$$\mathbf{P}\mathbf{P}^{\mathrm{T}} + (\mathbf{P}\mathbf{P}^{+})\mathbf{C}_{\mathrm{d}}(\mathbf{P}\mathbf{P}^{+})^{\mathrm{T}} = \mathbf{U}\boldsymbol{\Sigma}\left(\mathbf{I} + \boldsymbol{\Sigma}^{+}\mathbf{U}^{\mathrm{T}}\boldsymbol{\Delta}\boldsymbol{\Delta}^{\mathrm{T}}\mathbf{U}(\boldsymbol{\Sigma}^{+})^{\mathrm{T}}\right)\boldsymbol{\Sigma}^{\mathrm{T}}\mathbf{U}^{\mathrm{T}}, \tag{2.50}$$

and

$$\boldsymbol{\Sigma}^{+}\mathbf{U}^{\mathrm{T}}\boldsymbol{\Delta}\boldsymbol{\Delta}^{\mathrm{T}}\mathbf{U}(\boldsymbol{\Sigma}^{+})^{\mathrm{T}} = \mathbf{Z}\boldsymbol{\Lambda}\mathbf{Z}^{\mathrm{T}}. \tag{2.51}$$

Considering that $\mathbf{U}$ and $\mathbf{C}_{\mathrm{d}}$ have the dimension of $N_{\mathrm{d}} \times N_{\mathrm{e}}$ and $N_{\mathrm{d}} \times N_{\mathrm{d}}$, respectively, the multiplication $\mathbf{U}^{\mathrm{T}}\mathbf{C}_{\mathrm{d}}$ in Eq. (2.46) would require about $N_{\mathrm{d}}^{2}N_{\mathrm{e}}$ operations. One the other hand, assuming that N and N_{e} have the same order of magnitude, the multiplication $\mathbf{U}^{\mathrm{T}}\boldsymbol{\Delta}$ in Eq. (2.50) would require about $N_{\mathrm{e}}^{2}N_{\mathrm{d}}$ operations. Therefore, since N_{e} is usually much smaller than N_{d}, the application of the ensemble subspace inversion is more suitable for vast data sets.

2.3.3 Model errors and Correlated measurement errors

In most of the ensemble history matching applications, the measurement errors are considered uncorrelated, i.e., the matrix $\mathbf{C}_{\mathrm{d}}$ is diagonal [104, 73]. On the one hand, the use of a diagonal $\mathbf{C}_{\mathrm{d}}$ becomes important when one is using Eq. (2.30), (2.31), and (2.43), because the matrix inversion is much simpler. On the other hand, the correlated errors' negligence can underestimate the ensemble variance [41]. Therefore, the low-rank version of $\mathbf{C}_{\mathrm{d}}$ as in Eq. (2.49) avoids the construction of the measurement error matrix and can better represent the statistics of it. Additionally, augmenting $\boldsymbol{\Delta}$ with new realizations helps to reduce the sampling errors. For instance, Evensen [41] used N as 10 times N_{e}. He also provided a consistent formulation of history matching problems taking into account a proper error statistics of the measurement error.

The impact of model errors is another relevant point related to history matching. It is known that the forward models are not perfect, and they contain a series of simplifications and approximations. Alfonzo and Oliver [5] mentioned that ignoring model errors can result in the underestimation of the ensemble variability and biased production forecast. There are different methodologies in the reservoir community to tackle this problem [40, 5, 68]. For instance, Evensen [40] incorporated model errors as an uncertain parameter and estimated them during the application of the ensemble methods. He also forced the model simulations with the uncertain well rate data to account for these errors. Alfonzo and Oliver [5] updated the matrix $\mathbf{\Delta}$ at each iteration and developed a method to calculate the mismatch taking into consideration a non-diagonal $\mathbf{C}_{\mathrm{d}}$. By doing that, the authors could evaluate if the models achieved the desired mismatch level and, consequently, check if model parameters should be further updated. More differently, Luo [68] used a machine learning procedure to deal with model errors. The readers are referred to as the mentioned works for more information about each method.

Chapter 3

Localization

The finite ensemble size used in ensemble methods makes the history-matching problems feasible to compute. On the other hand, a limited ensemble size constrains the degrees of freedom of the problem to search for the best solution contained in the space spanned by the prior ensemble. Also, using a limited ensemble size results in sampling errors that reveal themselves through spurious correlations. A spurious correlation means that two uncorrelated variables would correlate when we use a finite ensemble to compute the correlation. Such spurious correlations might lead to unphysical updates of the model's uncertain parameters and contribute to underestimating the ensemble variance [37].

In practical applications of reservoir history matching, the number of realizations in the ensemble N_e is usually around 100 [19, 70, 32] and can get to 500 in some applications [104]. Thus, spurious correlations do appear. To illustrate this point, let us suppose one has a reservoir field with a sealing fault in the middle. Due to the limited size of the ensemble, one might find a correlation between porosity from one side of the fault and the produced oil rate from a well on the other side of the fault. There should be no correlation across the fault, but in this case, the spurious correlations can cause the oil rate measurement from this well to update the porosity on the other side of the fault. Consequently, this spurious update leads to an unphysical estimate.

To deal with these problems, one can increase the ensemble size or apply a localization technique [51, 88, 9]. To avoid spurious correlations and underestimates of the variability, the number of realizations in the ensemble would need to be very large, making the history-matching impractical. Therefore, localization is preferred as it extends the solution space and reduces the impact of spurious correlation.

The main idea behind localization is to select only a subset of the observed data with a type of relationship with the uncertain parameters to update them. There are different ways to implement it, such as the Kalman Gain localization [104, 31, 74], the covariance localization [30, 16], and local analysis [9, 88, 20, 99]. Moreover, there are different approaches on how to define the localization values, such as the distance-dependent approach [104, 30, 99] or the correlation-based method [64, 74].

This chapter presents the three most popular localization methods, i.e., Kalman gain localization, covariance localization, and local analysis. After that, we discuss the distance-based and correlation-based tapering approaches for determining which measurements influence which localized update variables.

3.1 Kalman-Gain localization

The Kalman-gain matrix $\mathbf{K}$ (Eq.(2.30)) has a dimension of $N_\mathrm{x} \times N_\mathrm{d}$, and each column represents a scaled covariance function of an observation. Furthermore, in ensemble methods, $\mathbf{K}$ is of low-rank and contains much redundant information, especially in the case with a vast observed data set [2]. Hence, the Kalman-Gain localization reduces the measurement's impact when updating uncertain model parameters with a strong relationship with the observation. It is worth mentioning that it is only possible to use the Kalman-Gain localization in approaches in which the Kalman-Gain matrix appears. Therefore, it is not possible to use it with the SEnRML.

If one considers Eq. (2.30), one can insert the Kalman-Gain localization matrix $\mathbf{C}$ through an element-wise multiplication (Schur product) as in

$$\mathbf{X}^{i+1} = \mathbf{X}^{i} + \mathbf{C} \circ \mathbf{K}^{i}\tilde{\mathbf{D}}^{i}. \tag{3.1}$$

$\mathbf{C}$ has the same dimension as $\mathbf{K}$, i.e., $N_\mathrm{x} \times N_\mathrm{d}$, and it represents the relation between the model uncertain parameters and the observed data. Its values vary between zero and one. Here, zero means that there is no relationship between the uncertain parameter and the measured data. By assigning values between zero and one, one gives different weights to this relationship.

The localization matrix $\mathbf{C}$ is also known as the tapering matrix. Besides eliminating spurious correlation, its use increases the degrees of freedom available to history match the observed data [30], and one computes each parameter's update in a much larger space than the ensemble space spanned by the prior realizations [2].

There are different ways to calculate the values in the localization matrix $\mathbf{C}$. See the presentation of the distance-dependent [51] and the correlation-based [70] methods at the end of this chapter.

3.2 Covariance localization

Another alternative is to localize the covariance matrices as in [16, 30, 99]. If one takes the IES-RML (Eq. (2.27)) as an example, the covariance localization can be computed from

$$\mathbf{X}^{i+1} = \mathbf{X}^{i} + \mathbf{C}_\mathrm{as} \circ \left(\mathbf{A}^{i}(\tilde{\mathbf{S}}^{i})^\mathrm{T}\right)\left(\mathbf{C}_\mathrm{ss} \circ \left(\tilde{\mathbf{S}}^{i}(\tilde{\mathbf{S}}^{i})^\mathrm{T}\right) + \alpha^{i}\mathbf{I}\right)^{-1}\tilde{\mathbf{D}}^{i}. \tag{3.2}$$

$\mathbf{C}_\mathrm{as}$ can be calculated as mentioned previously, i.e., by the distance-dependent or the correlation-based approach, and $\mathbf{C}_\mathrm{ss}$ can be defined as in [16]

$$\mathbf{C}_\mathrm{ss} = \mathbf{C}_\mathrm{as}^\mathrm{T}\mathbf{C}_\mathrm{as}. \tag{3.3}$$

The difference between the Kalman gain and covariance localization is the computation of $\mathbf{C}_\mathrm{ss}$, which is only performed in the latter case. It is worth mentioning that $\mathbf{C}_\mathrm{ss}$ has a dimension of $N_\mathrm{d} \times N_\mathrm{d}$, which can be very big; consequently, its computation and storage might be prohibitive. Moreover, it is not possible to use the effective Kalman Gain (Eq. (2.31)) with the covariance localization case [20]. Finally, Chen and Oliver [16] mentioned that the Kalman gain localization could generate an increase in the ensemble variability after the assimilation process compared to the covariance localization.

3.3 Local Analysis

Another localization method is the local analysis, firstly introduced by Brusdal et al. [9]. This method is more frequently used in atmosphere or ocean data-assimilation problems [99, 88, 84, 81], but less exploited in history-matching problems [20, 43]. In the local analysis, one selects smaller local groups of uncertain model parameters and updates each group by conditioning on a subset of the observations. Since the IES-RML and SEnRML differ from each other, the way of applying local analysis is different. Firstly, we will show how to perform local analysis using the IES-RML followed by the SEnRML case.

It is possible to rewrite Eq. (2.30) considering the local groups l

$$\mathbf{X}_l^{i+1} = \mathbf{X}_l^i + \mathbf{K}_l^i \tilde{\mathbf{D}}_l^i, \tag{3.4}$$

where $\mathbf{X}_l$ consists of some rows of $\mathbf{X}$, representing the local group. Chen and Oliver [20], for instance, selected individual columns of the 3D reservoir simulation model as groups. $\tilde{\mathbf{D}}_l^i$ consists of the rows of $\tilde{\mathbf{D}}^i$ that corresponds to the subset of the observed data related to the local group. $\mathbf{K}_l$ follows the same logic, and to compute it, one should use a subset of the ensemble anomalies $\mathbf{A}_l$ (rows of $\mathbf{A}$ that corresponds to the local groups), the predicted anomalies $\tilde{\mathbf{S}}_l$ (rows of $\tilde{\mathbf{S}}$ that corresponds to the data related to the local group), and the measurement error-covariance $\mathbf{C}_{\mathrm{d}l}$ (rows and columns of $\mathbf{C}_{\mathrm{d}}$ that corresponds to the data related to the local group).

If one deals with observed data with a physical location associated with the reservoir simulation models, such as well rates, the data selection is more straight-forward. One can use the same idea as in the distance-dependent approach and assign a radius or region where the data will be used to update the uncertain model parameter [20, 43, 99]. Then, one can select the rows in $\tilde{\mathbf{S}}$, $\tilde{\mathbf{D}}$, and $\mathbf{C}_{\mathrm{d}}$ using the approach from Sec. 3.4. However, if the observed data does not have such characteristics, one should use the correlation-based approach discussed in Sec. 3.5 to select the data.

It is worth mentioning that one can also taper the local Kalman gain as in

$$\mathbf{X}_l^{i+1} = \mathbf{X}_l^i + \mathbf{C}_l \circ \mathbf{K}_l^i \tilde{\mathbf{D}}_l^i. \tag{3.5}$$

Here $\mathbf{C}_l$ is part of the localization matrix $\mathbf{C}$ considering only the uncertain local model parameters (rows) and the selected observed data (columns). Note that one can apply the tapering or not, but its application might bring some benefits as we mention in Sec. 3.5

One advantage of local analysis is the smaller number of uncertain model parameters and observed data, making the manipulation and storage of the matrix $\mathbf{C}$ and $\mathbf{K}$ much easier. Consequently, local analysis is recommended for applications with big models and big data sets. Besides, it allows one to use a different linear combination of realizations for each uncertain model parameter. On the other hand, one must perform several local updates, i.e., one time for each local group, increasing the computational time. However, since the updates are independent, one can parallelize the process [37].

To illustrate this point, if one considers one column of a simulation model from the Brugge benchmark as the local group (9 gridblocks) and four uncertainties (the porosity value and permeability in the x, y, and z directions), one would have the number of uncertain model parameters $N_{\mathrm{x}} = 36$. Fig. 3.1 compares the memory consumption of $\mathbf{K}$

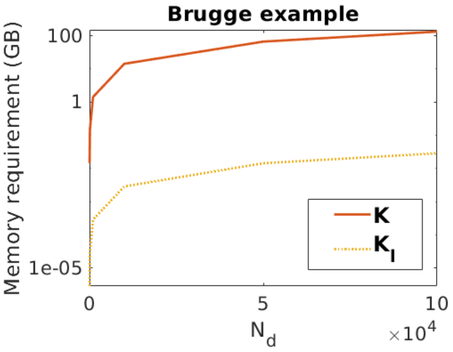


Figure 3.1: Memory requirement for the Brugge benchmark case for global ($\mathbf{K}$) *and local* ($\mathbf{K}_l$) *analysis (log scale).*

and $\mathbf{K}_l$ as a function of N_d for the Brugge case. As expected, the memory consumption of $\mathbf{K}$ is much higher than its local version. For instance, for 100 000 data points, the global Kalman gain requires about 133 GB, while the local one requires only 0.03 GB. However, one would need to perform the update 4 950 times (once for each local group). If one considers larger local groups, the memory requirement will increase while the number of updates decreases.

As mentioned previously, Emerick [31] proposed to compute each row of the Kalman Gain at a time. Note that he used all of the measurements, while in local analysis, one should select only the data related to the local group instead.

It is also possible to apply local analysis with SEnRML. By considering a local group, one should compute the local transition matrix $\mathbf{W}_l$ as in

$$\mathbf{W}_l^{i+1} = \mathbf{W}_l^i - \gamma\Big(\mathbf{W}_l^i - (\mathbf{P}_l^i)^\mathrm{T}\big((\mathbf{P}_l^i)(\mathbf{P}_l^i)^\mathrm{T} + \mathbf{C}_{\mathrm{d}l}\big)^{-1}\mathbf{H}_l^i\Big), \tag{3.6}$$

where in $\mathbf{P}_l$ and $\mathbf{H}_l$, one selects only the rows of $\mathbf{P}$ and $\mathbf{H}$ corresponding to the data related to the local group. Hence,

$$\mathbf{X}_l^{i+1} = \mathbf{X}_l^i + \mathbf{A}_l^i\mathbf{W}_l^i \tag{3.7}$$

Nevertheless, it is not possible to taper the Kalman gain as in the IES-RML. One alternative is to taper the ensemble prediction anomalies [20]. If one considers $\mathbf{c}_l$ as the tapering vector for one local group, it is possible to conduct localization as in

$$\hat{\mathbf{P}}_l^i = \big(\mathbf{c}_l^{1/2}\mathbf{1}_{N_\mathrm{e}}^\mathrm{T}\big) \circ \mathbf{P}_l^i, \tag{3.8}$$

and

$$\hat{\mathbf{H}}_l^i = \hat{\mathbf{P}}_l^i\mathbf{W}_l^i + \big(\mathbf{c}_l^{1/2}\mathbf{1}_{N_\mathrm{e}}^\mathrm{T}\big) \circ \mathbf{D}_l^i, \tag{3.9}$$

Note that there are some differences regarding the tapering in these two methods. In IES-RML, the Kalman Gain has a dimension of $N_\mathrm{x} \times N_\mathrm{d}$ and shows a relationship between each model uncertain parameter and observed data explicitly. Therefore, one can calculate the tapering coefficient related to each observed data and each uncertain model parameter. However, in SEnRML, since one tapers the predicted ensemble anomalies, the tapering coefficient is related to each observed data and the local group as a whole. Hence, the vector $\mathbf{c}_l$ should contain the relationship between the whole group and each observed data.

Chen and Oliver [20] and Fahimuddin et al. [43] used local analysis in reservoir history matching, and both works reported exciting results. Chen and Oliver [20], for instance, pointed out that because of a lower loss of information during the truncation of the singular value decomposition, the local analysis achieved a faster convergence than the case that considered all uncertain model parameters and observed data (global analysis). Moreover, Sakov and Bertino [99] pointed out that the choice of using local analysis or global analysis should be based on computational efficiency. Thus, for applications with big models and vast data sets, local analysis schemes might be preferable.

There is also a possibility to apply tapering through the measurement-error covariance matrix $\mathbf{C}_\mathrm{d}$ [99]. In such an approach, one should increase the error to reduce the effect of measurements that should not be used. If $\mathbf{C}_\mathrm{d}$ is diagonal, this approach should be equivalent to tapering the anomalies [99]. However, for the case of a non-diagonal $\mathbf{C}_\mathrm{d}$, the problem gets more complicated.

The following two sections present two of the most popular methods to compute the localization tapering matrix $\mathbf{C}$: the distance- and correlation-based tapering.

3.4 Distance-based tapering

The distance-based tapering approach appears to be the most popular one. Several works have applied it [30, 104, 19, 93]. In this approach, one should take the location of a measured data point and select a range in which this data should influence. Thus, this specific data point will influence all the uncertain model parameters within this range.

One can also use the function proposed by Gaspari and Cohn [47] to calculate the values in $\mathbf{C}$ and obtain a smooth update

$$c\left(\frac{h}{L}\right) = \begin{cases} -\frac{1}{4}\left(\frac{h}{L}\right)^5 + \frac{1}{2}\left(\frac{h}{L}\right)^3 + \frac{5}{8}\left(\frac{h}{L}\right)^3 - \frac{5}{3}\left(\frac{h}{L}\right)^2 + 1, & \text{if } 0 \leq \frac{h}{L} \leq 1 \\ \frac{1}{12}\left(\frac{h}{L}\right)^5 - \frac{1}{2}\left(\frac{h}{L}\right)^4 + \frac{5}{8}\left(\frac{h}{L}\right)^3 + \frac{5}{3}\left(\frac{h}{L}\right)^2 - 5\left(\frac{h}{L}\right) + 4 - \frac{2}{3}\left(\frac{h}{L}\right)^{-1}, & \text{if } 1 < \frac{h}{L} \leq 2 \\ 0, & \text{if } \frac{h}{L} > 2, \end{cases} \tag{3.10}$$

where L is the length scale defined by the user, h is the distance between the observed data and the uncertain model parameter, and c is the entry value of $\mathbf{C}$.

There are different ways to select L. If one uses well data, one could evaluate the drainage area of each well through streamlines [30, 24, 104] and then define L based on the wells' region of influence. Fig. 3.2 shows an example of the distance-based tapering. In this case, Soares et al. [104] defined the drainage area through streamlines to define the critical length L and to compute the tapering values according to the Eq. (3.10). Note that the area in red is very close to the well; hence, the data will have a strong impact on the update of the uncertain model parameters (porosity, permeability) in this area. As one moves away from the well, this value decreases until it reaches zero (dark blue region), meaning that the data from this well will not influence the update of the parameters in this area. In the end, one can avoid spurious correlation, especially the long-distance spurious correlation [2], where one observed data shows a correlation with an uncertain model parameter far from it.

In the case of local analysis, one would select only the data with values of c larger than zero.

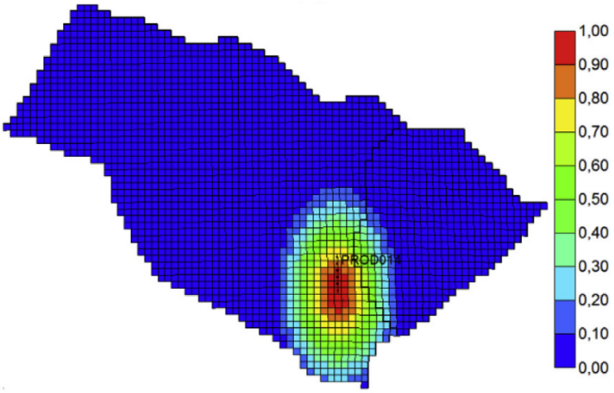


Figure 3.2: Example of distance-based tapering [104].

3.5 Adaptive correlation-based tapering

Luo and Bhakta [70] highlighted some drawbacks of using the distance-based tapering:

- Difficulty to select L for each data;
- Difficulty to incorporate time-lapse effect;
- Difficulty to handle non-local observations;
- The observed data needs to have an associated physical location;
- Often case-dependent implementation.

To tackle these issues, Evensen [37] proposed an adaptive localization tapering by using a correlation function. Later on, Luo and Bhakta [70] proposed a new method based on the same concept.

The main idea behind the adaptive correlation-based localization is to compute the correlation (ρ) between the uncertain model parameters and the simulated data by considering the ensemble (N_{e} realizations). Based on the correlation values, one selects which data should be used to update each uncertain model parameter. Fig. 3.3 shows a flowchart with the necessary steps to perform the adaptive correlation-based tapering [70]. The method can be divided into two parts. Firstly, in steps 1 and 2 in Fig. 3.3, one selects the data with a strong relationship with the uncertain model parameters based on a threshold. In this part, the authors proposed to compute the sampling error of the correlation between the simulated data and the uncertain model parameters and calculate the threshold based on that. In the second part (step 3 in Fig. 3.3), one computes the tapering matrix $\mathbf{C}$. However, instead of computing it based on the physical distance, as in Eq (3.10), the authors proposed a pseudo-distance taking into consideration the correlation values.

To select which data should be used to update each uncertain model parameter, Luo and Bhakta [70] proposed an automatic procedure to calculate a threshold θ that works as a cut-off in a role similar to L in the distance-based approach. In the first step shown in Fig. 3.3, the authors proposed two different ways to compute the sampling error ϵ of the

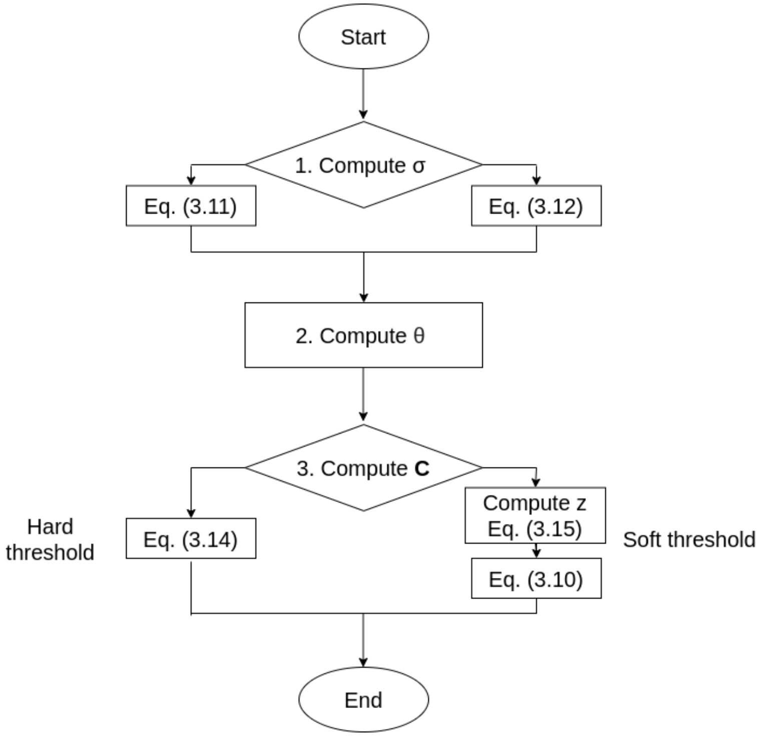


Figure 3.3: Flowchart of the adaptive correlation-based tapering [70].

correlation between the simulated data and the uncertain model parameters. The authors proposed to compute the correlation between the ensemble of the uncertain model parameters and a random ensemble of the simulated data. Theoretically speaking, the sample correlations between the model uncertain parameters of one ensemble member and another different ensemble member's simulated data would be zero, given an infinite ensemble size. However, due to the limited ensemble size, this correlation would not be zero. Therefore, the authors proposed to obtain the sampling errors (ϵ) through this sample correlation. Subsequently, Luo and Bhakta [70] calculated the standard deviation of ϵ according to the median absolute deviation estimator [26]

$$\sigma = \frac{\text{median}(\text{abs}(\varepsilon))}{0.6745}. \tag{3.11}$$

Donoho and Johnstone [26] mentioned that using the median to estimate the noise's standard deviation is a robust choice as it would not be influenced by a strong signal in case it is present. Note that by dividing the median by 0.6745, they considered a normal distribution of ϵ that covers 50% of the distribution, i.e., from the percentile $1/4$ to the percentile $3/4$ [96].

In this approach, one needs to compute the sample correlations between the ensemble of uncertain model parameters and the corresponding ensemble of simulated data. One also has to calculate the sample correlations between the ensemble of uncertain model parameters and a random ensemble of simulated data. To perform this latter computation, Luo and Bhakta [70] proposed to use a randomly shuffled ensemble of the uncertain model parameters and simulated data. Note that for that case, the ensemble members must be independent and identically distributed. Nevertheless, both computations can consume a lot of computer memory, especially for vast data sets.

Alternatively, one can consider that the sample correlations tend to follow a Gaussian distribution $\mathcal{N}\big(\rho, (1-\rho^2)^2/N_\mathrm{e}\big)$ asymptotically as $N_\mathrm{e} \to \infty$ [70, 6]. Thus, for correlations ρ equal to 0 (sampling errors), the standard deviation would be

$$\sigma = \frac{1}{\sqrt{N_\mathrm{e}}}. \tag{3.12}$$

By using Eq. (3.12) instead of Eq. (3.11), one can avoid the computation of the correlation between the ensemble of the uncertain model parameters and the random ensemble of simulated data and then reduce the memory consumption.

Finally, σ can be used to determine the threshold θ (step 2 in Fig. 3.3) using the universal threshold approach [25] as in

$$\theta = \sigma\sqrt{2\ln(\#\rho)}, \tag{3.13}$$

where $\#\rho$ is the number of model parameters used to calculate ρ, i.e., the size of the reservoir model. Note that for a larger model, one obtains a larger value of the threshold θ because of the higher probability to have larger deviations from the expected mean value. Also, multiplying σ by $\sqrt{2\ln(\#\rho)}$, makes one obtains a threshold larger than the standard deviation of the sampling error ϵ. Hence, with Eq. (3.13) one selects only the data that has a strong relationship with the model parameters. This approach is much used in wavelet denoising methods [52].

With the definition of the threshold θ, one can say that if a correlation between one simulated data and model uncertain parameter is larger than θ, this observed data should influence the update of this specific model uncertain parameter. By comparing the distance-based to the adaptive correlation-based tapering, it is possible to say that θ works as L. One can imagine that, for ρ values larger than the threshold, the value is within the localization region, as shown in Fig. 3.2.

Additionally, the authors proposed two different ways to compute the entry values c of the localization matrix $\mathbf{C}$ (step 3 in Fig. 3.3). First, if the correlation ρ between each uncertain model parameter and observed data is higher than θ, c related to them will have a value of 1, and 0 otherwise. This procedure is known as hard threshold and can be described as in

$$c = \begin{cases} 1, & \text{if } |\rho| \geq \theta \\ 0, & \text{if } |\rho| < \theta, \end{cases} \tag{3.14}$$

The other procedure, known as the soft threshold, uses the Gaspari and Cohn function (Eq. 3.10), but instead of using $\frac{h}{L}$ as the input, Luo and Bhakta [70] proposed a new variable z that works as a pseudo-distance

$$z = \frac{1 - \text{abs}(\rho)}{1 - \theta}. \tag{3.15}$$

It is essential to mention that one needs to compute this procedure for each uncertain group. For instance, one will have to perform the computations highlighted in Fig. 3.3 for porosity, permeability in x-, y-, and z-direction separately. Note, however, that if one uses Eq. (3.12) instead of Eq. (3.11), the threshold θ for each group will have the exact same value.

The hard threshold approach might lead to strong updates in specific regions [70], and it might generate models with abrupt changes in the porosity or permeability, for instance. Moreover, in both cases (soft and hard threshold), the correlation matrix calculation can require a large amount of memory in cases with big models and data sets, which might be a limitation for the method.

In local analysis, one should use either the distance- or correlation-based approach to select the data composing the subset during the update for each local group. Considering the IES-RML algorithm, Fig. 3.4 illustrates a simple example, where one has six uncertain model parameters (represented by the rows in $\mathbf{C}$ in the first plot) and 12 observed data (represented by the columns in $\mathbf{C}$ in the first plot). The frame with a solid line represents the matrix, and each small square with the dashed-line represents one element of the matrix. The small squares painted in blue show that the element is not zero and those in white indicate a zero value. If one divides the uncertain model parameters into a local groups considering the first three rows, one would first achieve a local tapering matrix considering all the measurements, as shown in the middle plot in Fig. 3.4. Subsequently, one would select only the measurements that influence the model parameters of that specific local group, i.e., one should select only the columns with values different from zero (right plot in Fig. 3.4). Therefore, the final $\mathbf{C}_{\mathrm{l}}$ matrix would use only the columns number 1, 2, 4, 6, 8, 10, and 12. Finally, one would also need to select the rows in the matrices $\tilde{\mathbf{S}}$ and $\tilde{\mathbf{D}}$ that are related to the measurements one selected based on $\mathbf{C}$. Hence, one needs to chose the rows number 1, 2, 4, 6, 8, 10 and 12 of $\tilde{\mathbf{S}}$ and $\tilde{\mathbf{D}}$ to form $\tilde{\mathbf{S}}_{\mathrm{l}}$ and $\tilde{\mathbf{D}}_{\mathrm{l}}$. Besides, one would also need to select the same rows and columns of $\mathbf{C}_{\mathrm{d}}$ and the rows in $\mathbf{A}$ related to the local model uncertain parameters to form $\mathbf{C}_{\mathrm{d}l}$ and $\mathbf{A}_l$.

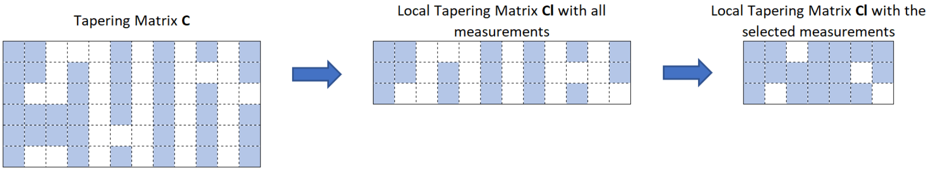

Tapering Matrix **C**
Local Tapering Matrix **Cl** with all measurements
Local Tapering Matrix **Cl** with the selected measurements

Figure 3.4: Localization matrix $\mathbf{C}$ *and* $\mathbf{C}_\mathrm{l}$.

The main point of using the distance- or correlation-based approach in local analysis is to select the observed data that one will use to update each local group. The tapering function, i.e., the computation of the values in $\mathbf{C}_\mathrm{l}$ is not necessarily mandatory. If one does not use the tapering function, one will achieve the same configuration as the hard threshold presented in [70]. However, if one uses the tapering function, one would get $\mathbf{C}_\mathrm{l}$ between 0 and 1 as the soft threshold approach.

Chapter 4

Sparse Representation

Many research fields use sparse representation methods, including signal and image processing, and many different application areas, such as oil and gas and health industries. The objectives within these fields and applications can vary, such as image denoising [4, 116], data compression [62], image restoration [77, 94], and classification of images and signals [114, 15]. In petroleum applications, sparse representation can be a valuable tool for dealing with vast and noisy data sets. As mentioned previously, the observed data sets' size has been increasing with the development of new sensors and technologies. The inclusion of 4D seismic data set, for instance, increased the number of observed data points from 1 400 to about 7 million in [73]. If one considers a Permanent Reservoir Monitoring, this number can increase even further. Besides, regardless of the type of the observed data, they usually contain redundant information. Hence, sparse representation methods can help to select only the important and non-redundant information. Note that one can also reduce the size of the uncertain model parameters by using sparse representation methods.

In this chapter, we first present the general sparse representation problem, in which one uses an over-complete dictionary to represent the original signal. Subsequently, we introduce the Orthogonal Matching Pursuit (OMP) algorithm to solve this problem, and present the process of learning the dictionary from the original signal by using the K-SVD method. In addition, we highlight different applications that used the K-SVD method. Note that the symbols and variables in this chapter have a different notation.

4.1 The sparse representation problem

Sparsity means that some of the values in a vector or matrix are equal to zero. One way of achieving sparsity regarding a signal or image is defining basis functions and exploring linear combinations of them to form the original signal or image. Suppose that $\mathbf{x} \in \mathbb{R}^{n^2 \times 1}$ is a vector containing the original signal and $\mathbf{D} \in \mathbb{R}^{n^2 \times k}$ is a set of pre-defined basis functions, also called the dictionary. Each column of the dictionary is also known as an atom, and $\mathbf{D}$ is an over-complete matrix, i.e., $k > n^2$. Therefore, if one writes

$$\mathbf{x} = \mathbf{D}\gamma, \tag{4.1}$$

it is possible to represent the original signal $\mathbf{x}$ by a sparse vector $\gamma \in \mathbb{R}^{k \times 1}$. In other words, the signal $\mathbf{x}$ is represented by a linear combination of the atoms (or columns) in $\mathbf{D}$. The solution of the Eq. (4.1) can be complicated as it is an under-determined linear

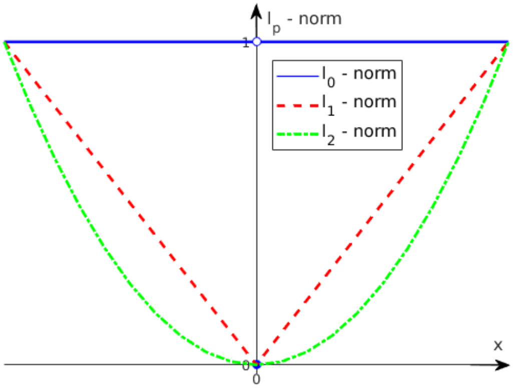


Figure 4.1: ℓ_p-norm interpretation - 1D.

system for γ with the number of variables in γ larger than the number of equations. Hence, it follows a similar logic as the history-matching problems, i.e., it is an ill-posed problem that does not have a single and unique solution [115].

A common practice to solve this equation is to apply a regularization criterion that can narrow the solution's space [28] as stated in the following problem

$$\text{argmin}_{\{\mathbf{x}\}} J(\mathbf{x}) \quad \text{subject to} \quad \mathbf{x} = \mathbf{D}\gamma, \tag{4.2}$$

where $J(\cdot)$ is a cost function. A reasonable choice for J that is the ℓ_2-norm. Since the ℓ_2-norm is a strictly convex and differentiable function, this type of function is attractive for the problems in Eq. (4.2) [28]. However, one may also consider more general functions such as ℓ_p-norms, in which $p > 0$. One can define ℓ_p-norm as

$$||\mathbf{x}||_p = \Big(\sum_{i=1}^{n^2} |\mathbf{x}_i|^p\Big)^{1/p}. \tag{4.3}$$

One can also define the pseudo-ℓ_0-norm as

$$||\mathbf{x}||_0 = \lim_{p \to 0} \Big(\sum_{i=1}^{n^2} |\mathbf{x}_i|^p\Big)^{1/p}. \tag{4.4}$$

One can interpret the pseudo-ℓ_0-norm as the number of non-zero coefficients of $\mathbf{x}$. Note that the pseudo-ℓ_0-norm does not fulfill all the requirements of a norm [28]. However, it is common to refer to it as the ℓ_0-norm, and we will use this notation from this point on.

Fig. 4.1 illustrates the 1D interpretation of the ℓ_0-, ℓ_1-, and ℓ_2-norms. Note that ℓ_0- and ℓ_1-norms are not locally differentiable at 0. Consequently, minimizing the ℓ_0- and ℓ_1-norms can be an issue in the sparse representation problem. Nevertheless, these norms can promote a sparsity solution that can be interesting and useful for some problems.

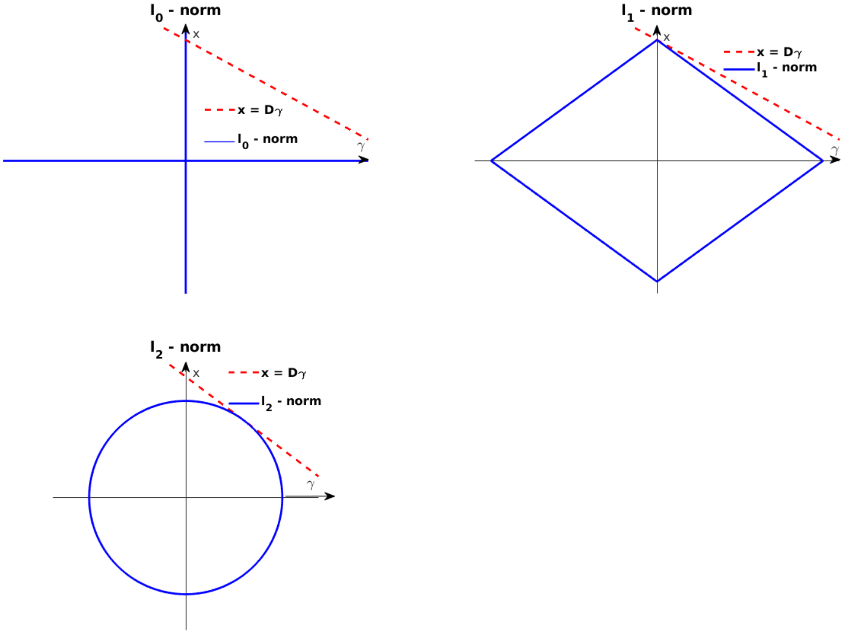


Figure 4.2: Geometric interpretation of the problem in Eq. (4.2): ℓ_0-norm (upper left); ℓ_1-norm (upper right); and ℓ_2-norm (lower left).

To have a better understanding of the property of promoting sparsity, Fig. 4.2 shows a 2D geometrical interpretation of the problem stated by Eq. (4.2) when J assumes the form of ℓ_0-, ℓ_1-, and ℓ_2-norms. All the solutions must intersect the line $\mathbf{x} = \mathbf{D}\gamma$ (red-dashed line). Hence, the problem's solution is represented by the intersection of the norms (blue-solid lines) and the red-dashed line. One can see that for the ℓ_0- and ℓ_1-norms, the solution is sparse as one gets a zero value. Elad [28] highlighted that is not very common to reach non-sparse solutions for the ℓ_1-norm case. However, the solution is not sparse for the ℓ_2-norm [28, 115].

The concept of the ℓ_0-norm is very convenient for the sparse representation problems. Hence, it is the preferable choice for J in most applications. Nevertheless, it is a challenge to exactly represent the initial signal $\mathbf{x}$ with few non-zero coefficients in real applications of image and signal processing [109]. Therefore, a common practice is to add an error ϵ that can be tolerated, and one can write Eq. (4.2) as

$$\text{argmin}_{\{\mathbf{x}\}} \|\mathbf{x}\|_0 \quad \text{subject to} \quad \|\mathbf{x} - \mathbf{D}\gamma\|_2^2 \leq \epsilon, \tag{4.5}$$

or

$$\text{argmin}_{\{\mathbf{x}\}} \|\mathbf{x} - \mathbf{D}\gamma\|_2^2 \quad \text{subject to} \quad \|\mathbf{x}\|_0 \leq s, \tag{4.6}$$

where s is the maximum sparsity level desired. In the next section, we provide more information about the algorithms used to solve such problems. Note that one can also use the ℓ_1-norm, which is also known as the Lasso problem [106], but the focus in the thesis is on the ℓ_0-norm problem.

4.1.1 Sparse representation algorithms

There are two approaches to solve the problem in Eq. (4.5) and (4.6) [28, 115]. One can either try to relax $\|\mathbf{x}\|_0$ or one can keep the ℓ_0-norm and focus on finding a solution that focus on the term $\|\mathbf{x} - \mathbf{D}\gamma\|_2^2$.

The first type of solution is also known as Convex Relaxation Techniques, where one replaces $\|\mathbf{x}\|_0$ by a smooth or continuous function. Among these methods, it is possible to point out the FOCal Undetermined System Solver (FOCUSS) [48], in which the authors proposed to substitute the ℓ_0-norm by the ℓ_p-norm (where $p \in (0, 1]$). Another method is to substitute the ℓ_0-norm by the ℓ_1-norm. This method is known as Basis Pursuit (BP) [14]. However, in this work, we focus on the second solution where one deals with the term $\|\mathbf{x} - \mathbf{D}\gamma\|_2^2$ instead since it tends to be faster [28].

In the second type, one can solve the problem by performing a combinatorial search [28]. Supposing the sparsest solution γ has N_{nz} non-zero coefficients, one would need to check all possible combinations of N_{nz} columns in $\mathbf{D}$, in which $\mathbf{x} = \mathbf{D}_J\gamma_J$. In this case, J represents the columns' indices in the dictionary and the rows in the sparse vector. If one assumes that the dictionary has 750 columns and the sparsest solution has ten non-zero coefficients, one would need to search within $1.46E + 22$ possible combinations, which is impractical.

Many works in the literature used greedy algorithms to tackle this issue, achieving a good approximate solution [108, 107, 80]. One of the most common greedy algorithms used in sparse representation problems is the Orthogonal Matching Pursuit (OMP) [89].

Orthogonal Matching Pursuit (OMP)

Several works used the OMP to sparsely represent data [4, 98, 7, 62]. The basic idea behind the OMP algorithm is to add one non-zero coefficient to the sparse vector at a time until the stopping criterion is reached.

The first step of the calculation of the sparse vector is to define an initial dictionary. Rubinstein et al. [97] pointed out that there are two different sources of dictionaries: a mathematical model or a set of realizations of the original data. Among the mathematical models, it is possible to highlight the Discrete Cosine Transform (DCT), Curvelets, Contourlets, and Discrete Wavelet Transforms (DWT). One advantage of such functions is their easy implementation. However, using realizations of the data is more flexible.

Algorithm 1 shows all the necessary steps in the OMP to perform the sparse representation. It starts with the definition of which atom (column) in the dictionary $\mathbf{D}$ has a strong relationship with the signal $\mathbf{x}$. To do so, one computes the inner product between each atom $\mathbf{d}_j$ and the residual $\mathbf{r}$ and selects the one with the largest value (line 5). Note that for the first iteration the residual is equal to the signal $\mathbf{x}$. Subsequently, one should add the index related to this atom (j) into the variable J (line 6), so one can project the signal $\mathbf{x}$ onto the space spanned by the J atoms in $\mathbf{D}$ (line 7), where $(\mathbf{D}_J)^+$ represents the pseudo-inverse of $\mathbf{D}_J$. Finally, one calculates the residual $\mathbf{r}$ (line 8) and repeats the process until the stopping criterion (error or sparsity level) is reached. It is worth mentioning that one fills the sparse vector with one non-zero coefficient at a time.

On the other hand, Aharon et al. [4] highlighted that this algorithm might not work well if one faces a big original signal $\mathbf{x}$. Therefore, they proposed to divide $\mathbf{x}$ into smaller patches. For instance, if one has a 2D initial signal, one should extract smaller regular patches (with size $n \times n$) and transform them into column vectors to compute the

Algorithm 1: OMP

```
Input: signal $\mathbf{x}$, Dictionary $\mathbf{D}$, error $\epsilon$ or sparsity level $s$
Output: sparse vector $\gamma$
Initialization: $\mathbf{r} := \mathbf{x}, \gamma := 0, J := ()$
while stopping criterion not met do
    $\hat{j} := \text{argmax}_{\{j\}} |\mathbf{r}^{\mathrm{T}}\mathbf{d}_j|$
    $J := (J, \hat{j})$
    $\gamma_J := (\mathbf{D}_J)^{+}\mathbf{x}$
    $\mathbf{r} := \mathbf{x} - \mathbf{D}_J\gamma_J$
end
```

sparse representation of each patch separately. These patches need to cover the whole initial signal, and they can overlap with each other. The following equation considers the original signal as a collection of patches

$$\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_{N_{\text{ts}}}], \tag{4.7}$$

where N_{ts} is the total number of patches (or the size of the training data set). Consequently, it is possible to rewrite Eq. (4.1) as

$$\mathbf{X} = \mathbf{D\Gamma}. \tag{4.8}$$

Here, $\mathbf{\Gamma}$ is the sparse matrix composed of the sparse vectors γ

$$\mathbf{\Gamma} = [\gamma_1, \gamma_2, \dots, \gamma_{N_{\text{ts}}}]. \tag{4.9}$$

To calculate $\mathbf{\Gamma}$, one needs to compute the OMP algorithm N_{ts} times, i.e., one time for each patch.

Rubinstein et al. [98] and Davis et al. [22] proposed a more efficient formulation of the OMP algorithm, in which it is possible to reduce the computational cost. Consequently, it is more suitable for problems involving vast data sets (see Algorithm 2).

The main idea of the new formulation is to replace the computation of the residual $\mathbf{r}$ with $\mathbf{D}^{\mathrm{T}}\mathbf{r}$. Thus, if one considers $\alpha = \mathbf{D}^{\mathrm{T}}\mathbf{r}$, $\alpha^0 = \mathbf{D}^{\mathrm{T}}\mathbf{x}$, the sparse vector $\gamma_J = \mathbf{D}_J^{+}\mathbf{x}$, and $\mathbf{G} = \mathbf{D}^{\mathrm{T}}\mathbf{D}$, it is possible to write α as in

$$\begin{aligned}
\alpha &= \mathbf{D}^{\mathrm{T}}\mathbf{r} \\
&= \mathbf{D}^{\mathrm{T}}(\mathbf{x} - \mathbf{D}_J\gamma_J) \\
&= \mathbf{D}^{\mathrm{T}}(\mathbf{x} - \mathbf{D}_J(\mathbf{D}_J)^{+}\mathbf{x}) \\
&= \alpha^0 - \mathbf{G}_J(\mathbf{D}_J)^{+}\mathbf{x} \\
&= \alpha^0 - \mathbf{G}_J(\mathbf{D}_J^{\mathrm{T}}\mathbf{D}_J)^{-1}\mathbf{D}_J^{\mathrm{T}}\mathbf{x} \\
&= \alpha^0 - \mathbf{G}_J(\mathbf{G}_{J,J})^{-1}\alpha_J^0.
\end{aligned} \tag{4.10}$$

Note that $\mathbf{G}_J = \mathbf{D}^{\mathrm{T}}\mathbf{D}_J$ and $\mathbf{G}_{J,J} = \mathbf{D}_J^{\mathrm{T}}\mathbf{D}_J$.

Therefore, one can run the OMP algorithm without explicitly computing the residual $\mathbf{r}$. On the other hand, if one uses the problem stated in Eq. (4.5), one needs the residual to verify the stopping criterion. To overcome this issue, Rubinstein et al. [98] proposed the following

$$\begin{aligned}\|\mathbf{r}^m\|_2^2 &= (\mathbf{r}^m)^{\mathrm{T}}\mathbf{r}^m \\ &= (\mathbf{x} - \mathbf{D}\gamma^m)^{\mathrm{T}}(\mathbf{x} - \mathbf{D}\gamma^m) \\ &= \mathbf{x}^{\mathrm{T}}\mathbf{x} - 2\mathbf{x}^{\mathrm{T}}\mathbf{D}\gamma^m + (\gamma^m)^{\mathrm{T}}\mathbf{G}\gamma^m \\ &= \mathbf{x}^{\mathrm{T}}\mathbf{x} - 2(\mathbf{r}^m + \mathbf{D}\gamma^m)^{\mathrm{T}}\mathbf{D}\gamma^m + (\gamma^m)^{\mathrm{T}}\mathbf{G}\gamma^m \\ &= \mathbf{x}^{\mathrm{T}}\mathbf{x} - 2(\gamma^m)^{\mathrm{T}}\mathbf{G}\gamma^m + (\gamma^m)^{\mathrm{T}}\mathbf{G}\gamma^m \\ &= \|\mathbf{x}\|_2^2 - (\gamma^m)^{\mathrm{T}}\mathbf{G}\gamma^m,\end{aligned} \tag{4.11}$$

where m is the iteration number within the calculation of one sparse vector and the signal approximation is orthogonal to the residual such that

$$(\mathbf{r}^m)^{\mathrm{T}}\mathbf{D}\gamma^m = 0. \tag{4.12}$$

Besides, it is possible to write the square of the ℓ_2-norm of the original signal vector $\|\mathbf{x}\|_2^2$ as

$$\begin{aligned}\|\mathbf{x}\|_2^2 &= (\mathbf{r}^{m-1} + \mathbf{D}\gamma^{m-1})^{\mathrm{T}}(\mathbf{r}^{m-1} + \mathbf{D}\gamma^{m-1}) \\ &= \left\|\mathbf{r}^{m-1}\right\|_2^2 + 2(\mathbf{r}^{m-1})^{\mathrm{T}}\mathbf{D}\gamma^{m-1} + (\gamma^{m-1})^{\mathrm{T}}\mathbf{D}^{\mathrm{T}}\mathbf{D}\gamma^{m-1}.\end{aligned} \tag{4.13}$$

Finally, if one substitutes Eq. (4.13) in Eq. (4.11), one can achieve the following formulation for the residual

$$\begin{aligned}\|\mathbf{r}^m\|_2^2 &= \left\|\mathbf{r}^{m-1}\right\|_2^2 + (\gamma^{m-1})^{\mathrm{T}}\mathbf{G}\gamma^{m-1} - (\gamma^m)^{\mathrm{T}}\mathbf{G}\gamma^m \\ &= \epsilon^{m-1} + \delta^{m-1} - \delta^m,\end{aligned} \tag{4.14}$$

where ϵ^m can be defined as the computed error at the iteration m and δ as a weighted norm.

Algorithm 2 shows all the steps proposed by [98]. Firstly, one needs to define α^0, the initial computed error ϵ^0, $\mathbf{G}$, and the specified error ϵ (line 1). Then, one needs to determine the atom in the dictionary $\mathbf{D}$ that has the largest inner product with the residual $\mathbf{r}$ and append it to J (lines 5 and 6). In line 7, one must calculate the sparse vector γ_J, and one can use the Cholesky decomposition to invert the matrix $\mathbf{G}_{J,J}$. Subsequently, to calculate α (Eq. (4.10)), one should follow the lines 8 and 9. Finally, lines 10 and 11 are responsible for calculating the residual ($\|\mathbf{r}^m\|_2^2$ or ϵ^m) (Eq. (4.14)) in order to check if the stopping criterion is satisfied.

The definition of the specified error ϵ can be complex, but the signal may be contaminated by noise in some applications. As a result, one can define this noise and its standard deviation σ to determine the error. To avoid overfitting, Elad and Aharon [29] proposed the following formula to set the error

$$\epsilon = \left(C\sigma\sqrt{n^2}\right)^2. \tag{4.15}$$

Here C is a constant that was empirically defined in [29] as 1.15. Note that this algorithm is only valid for the problem in Eq. (4.5). For the problem in Eq (4.6), one does not need to compute the residual, and the algorithm is more straightforward.

Algorithm 2: OMP [98]

```
Input: $\alpha^0 = \mathbf{D}^{\mathrm{T}}\mathbf{x}$, $\epsilon^0 = \mathbf{x}^{\mathrm{T}}\mathbf{x}$, $\mathbf{G} = \mathbf{D}^{\mathrm{T}}\mathbf{D}$, specified error $\epsilon$
Output: sparse vector $\gamma$
Initialization: $m := 1, \delta^0 := 0, J := (), \alpha := \alpha^0$
while $\epsilon^{m-1} > \epsilon$ do
    $\hat{j} := \mathrm{argmax}_{\{j\}}\{|\alpha_j|\}$
    $J := (J, \hat{j})$
    $\gamma_J := (\mathbf{G}_{J,J})^{-1}\alpha^0_J$
    $\beta := \mathbf{G}_J\gamma_J$
    $\alpha := \alpha^0 - \beta$
    $\delta^m := (\gamma_J)^{\mathrm{T}}\beta_J$
    $\epsilon^m := \epsilon^{m-1} - \delta^m + \delta^{m-1}$
    $m := m + 1$
end
```

4.2 Learning the Dictionary

The idea to adapt the initial dictionary according to the signal that is being sparsely represented has gained much attention as one can achieve a better representation of the signals with fewer coefficients [58, 4, 87, 61, 109].

Among the algorithms used for this task, it is possible to point out the maximum likelihood methods [87], the method of optimal directions (MOD) [34], the maximum a posteriori probability method [58], the Union of orthonormal bases [60], and the K-SVD [4].

Firstly, we will give a brief introduction to each of these methods. Then, we will provide more information about the K-SVD (which is used in this work).

The maximum likelihood method uses the following likelihood $p(\mathbf{X}|\mathbf{D})$ and tries to find the dictionary $\mathbf{D}$ that maximizes it [87]. The maximum a posteriori method also uses the concept of likelihood, but it also considers the initial distribution of $\mathbf{D}$ by using Bayes' theorem. Hence, Kreutz-Delgado et al. [58] proposed to solve the following problem $p(\mathbf{D}|\mathbf{X}) \propto p(\mathbf{X}|\mathbf{D})p(\mathbf{D})$.

In the MOD algorithm, Engan et al. [34] minimizes the residual $\mathbf{E}$

$$||\mathbf{E}||_F^2 = ||\mathbf{X} - \mathbf{D}\mathbf{\Gamma}||_F^2, \tag{4.16}$$

where the subscript F denotes the Frobenius norm, while keeping the original signal $\mathbf{X}$ fixed. Note that one can define the Frobenius norm as

$$||\mathbf{E}||_F^2 = \sqrt{\mathrm{trace}(\mathbf{E}^{\mathrm{T}}\mathbf{E})}. \tag{4.17}$$

Lesage et al. [60] proposed to use a union of orthonormal bases to compose the Dictionary. By doing that, the sparse representation problem gets simpler as one can define the sparse vector γ as

$$\gamma = \mathbf{D}^{\mathrm{T}}\mathbf{x}. \tag{4.18}$$

Nevertheless, the restriction imposed by the orthonormality of the dictionary may have downgraded the process, and the results showed an inferior performance [4].

Aharon et al. [4] mentioned that all these algorithms tend to be slow, and the dictionary's update does not take into account the coefficients defined in the sparse representation. Therefore, they proposed a new method, named K-SVD, which updates the dictionary and the nonzero coefficients of the sparse matrix at the same time.

4.2.1 K-SVD

The main contribution of the K-SVD method is to optimize the dictionary $\mathbf{D}$ together with only the nonzero coefficients of the sparse matrix $\mathbf{\Gamma}$ while keeping all zero values constant. To tackle this problem, the authors optimized each atom of the dictionary at a time. The first step is to define the residual $\mathbf{E}_j$, where one should not consider the contribution of the j-th atom that is being updated, as in

$$\mathbf{E}_j = \mathbf{X}_{I_j} - \sum_{a \neq j} \mathbf{d}_a \mathbf{\Gamma}_{a,I_j}. \tag{4.19}$$

Here I_j represents the column indices in $\mathbf{X}$ and $\mathbf{\Gamma}$ that uses the atom j.

Another way of understanding the sparse matrix $\mathbf{\Gamma}$ is to consider it as a collection of row vectors $\mathbf{g}^{\mathrm{T}}$. Thus, $\mathbf{g}^{\mathrm{T}}_{j,I_j}$ is the j-th row of $\mathbf{\Gamma}$ over the columns defined in I_j. For instance, if $\mathbf{g}^{\mathrm{T}}_j = (0, \gamma_{j,2}, 0, \ldots, 0, \gamma_{j,10}, 0, \ldots, 0, \gamma_{j,22}, 0, \ldots, 0)$. Then, $\mathbf{g}^{\mathrm{T}}_{j,I_j} = (\gamma_{j,2}, \gamma_{j,10}, \gamma_{j,22})$. Finally, one can write the optimization problem as

$$(\mathbf{d}_j, \mathbf{g}^{\mathrm{T}}_{j,I_j}) = \mathrm{argmin}_{\mathbf{d}_j, \mathbf{g}^{\mathrm{T}}_{j,I_j}} \left\| \mathbf{E}_j - \mathbf{d}_j \mathbf{g}^{\mathrm{T}}_{j,I_j} \right\|_F^2 \text{ subject to } \quad \left\| \mathbf{d}_j \right\|_2 = 1. \tag{4.20}$$

To solve this problem, one can compute the Singular Value Decomposition (SVD) of the residual $\mathbf{E}_j$ as in

$$\mathbf{E}_j = \mathbf{U} \mathbf{\Sigma} \mathbf{V}^{\mathrm{T}}, \tag{4.21}$$

and assign the atom $\mathbf{d}_j$ to the first column of $\mathbf{U}$ and $\mathbf{g}^{\mathrm{T}}_{j,I_j}$ to the first column of $\mathbf{V}$ times the first singular value $\mathbf{\Sigma}(1,1)$ [115].

Algorithm 3 describes the K-SVD method, an iterative algorithm, where one alternates between calculating the sparse representation matrix and the dictionary update. As such, one needs to define the total number of iterations N_{it} beforehand.

The first step is defining the initial dictionary $\mathbf{D}$, the specified error ϵ or the sparsity level s, and calculating the sparse representation matrix (line 4). Note that one should compute the Algorithm 1 and or 2 N_{ts} times to form the sparse matrix $\mathbf{\Gamma}$. Subsequently, one should update each column of the dictionary and the nonzero coefficients of each row of the sparse matrix $\mathbf{\Gamma}$ at a time. We first select the indices in the original signal $\mathbf{X}$ that uses the atom j (line 6), followed by the residual calculation without considering the j atom $\mathbf{E}_j$(line 7). The next step (line 8) calculates the SVD of the residual $\mathbf{E}_j$. Then, the updated atom takes the first column of $\mathbf{U}$, and the nonzero values of the j-th row of $\mathbf{\Gamma}$ are equal to the largest singular value times the first column of $\mathbf{V}$ (lines 9 and 10, respectively).

Nevertheless, Rubinstein et al. [98] mentioned that if one faces a large initial signal $\mathbf{X}$, the algorithm can be time-consuming, mainly because of the explicit computation of the residual $\mathbf{E}_j$. Moreover, they claimed that the goal of the K-SVD is to improve the dictionary to achieve better sparse representation. Therefore, the updated dictionary does not have to be optimal. Hence, they developed an approximate method faster than the original formulation that provides close results to its full computation.

To solve the problem stated in Eq. (4.20), one can think about making the two parts of the subtraction equal, as in

$$\mathbf{E}_j = \mathbf{d}_j \mathbf{g}^{\mathrm{T}}_{j,I_j}. \tag{4.22}$$

Algorithm 3: K-SVD [4]

```
Input: signal $\mathbf{X}$, initial Dictionary $\mathbf{D}$, number of iterations $N_{\text{it}}$, specified error $\epsilon$ or sparsity level $s$
Output: sparse matrix $\mathbf{\Gamma}$ and updated dictionary $\mathbf{D}$
for $i = 1$ to $N_{it}$ do
    Sparse representation calculation (Algorithm 1 or 2) for each patch to form $\mathbf{\Gamma}$
    for $j = 1$ to $N_k$ do
        $I_j$ := indices of the columns in $\mathbf{X}$ that uses $\mathbf{d}_j$
        $\mathbf{E}_j := \mathbf{X}_{I_j} - \sum_{a \neq j} \mathbf{d}_a \mathbf{\Gamma}_{a,I_j}$
        $\mathbf{E}_j := \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\mathrm{T}}$
        $\mathbf{d}_j := \mathbf{U}_1$
        $\mathbf{g}_{j,I_j}^{\mathrm{T}} := \Sigma(1,1) * \mathbf{V}_1$
    end
end
```

By right multiplying this equation by $\mathbf{g}_{j,I_j}$ and dividing by $\mathbf{g}_{j,I_j}^{\mathrm{T}}\mathbf{g}_{j,I_j}$, one can achieve

$$\mathbf{d}_j = \frac{\mathbf{E}_j \mathbf{g}_{j,I_j}}{\mathbf{g}_{j,I_j}^{\mathrm{T}} \mathbf{g}_{j,I_j}}. \tag{4.23}$$

Considering that one can normalize the atom $\mathbf{d}_j$ afterwards, it is possible to disregard the denominator. Besides, substituting Eq. (4.19) into Eq. (4.23), one has

$$\mathbf{d}_j = \mathbf{X}_{I_j}\mathbf{g}_{j,I_j} - \Big(\sum_{a \neq j} \mathbf{d}_a \mathbf{\Gamma}_{a,I_j}\Big)\mathbf{g}_{j,I_j}. \tag{4.24}$$

Finally, one should normalize the atom $\mathbf{d}_j$ as in

$$\mathbf{d}_j = \frac{\mathbf{d}_j}{\left\|\mathbf{d}_j\right\|_2}. \tag{4.25}$$

The next step is to update the nonzero coefficients $\mathbf{g}_{j,I_j}^{\mathrm{T}}$. Thus, if one multiplies from the left by $\mathbf{d}_j^{\mathrm{T}}$ both sides of Eq. (4.22), one obtains

$$\mathbf{g}_{j,I_j}^{\mathrm{T}} = \mathbf{d}_j^{\mathrm{T}} \mathbf{E}_j, \tag{4.26}$$

and by substituting $\mathbf{E}_j$ (Eq. (4.19)), it is possible to get

$$\mathbf{g}_{j,I_j}^{\mathrm{T}} = \mathbf{d}_j^{\mathrm{T}} \mathbf{X}_{I_j} - \mathbf{d}_j^{\mathrm{T}} \Big(\sum_{a \neq j} \mathbf{d}_a \mathbf{\Gamma}_{a,I_j}\Big). \tag{4.27}$$

The step-by-step implementation of the new development in [98] is shown in Algorithm 4. Note that to learn the dictionary, the number of patches (or the training dataset size) has to be larger than the number of atoms in the dictionary (N_K). Besides, Aharon et al. [4] mentioned that if an atom in the dictionary is not used during the sparse representation, i.e., if $\mathbf{g}_{j,I_j}^{\mathrm{T}}$ is empty, one can replace the atom by calculating the residual $\mathbf{R}_i = \mathbf{X} - \mathbf{D}\mathbf{\Gamma}$ at a specific iteration, and choosing the column in $\mathbf{R}_i$ with the largest column sum. By doing that, one can avoid overfitting and local minima.

As mentioned previously, it is not easy to exactly represent the signal $\mathbf{X}$ with a sparse representation. Therefore, one can reconstruct the original signal with the dictionary $\mathbf{D}$

Algorithm 4: K-SVD - New formulation [98]

```
Input: signal $\mathbf{X}$, initial Dictionary $\mathbf{D}$, number of iterations $N_{\mathrm{it}}$, specified error $\epsilon$ or sparsity level $s$
Output: sparse matrix $\boldsymbol{\Gamma}$ and updated dictionary $\mathbf{D}$
for $i = 1$ to $N_{it}$ do
    Sparse representation calculation (Algorithm 1 or 2) for each patch to form $\boldsymbol{\Gamma}$
    for $j = 1$ to $N_k$ do
        $I_j$ := indices of the columns in $\mathbf{X}$ that uses $\mathbf{d}_j$
        $\mathbf{d}_j := \mathbf{X}_{I_j}\mathbf{g}_{j,I_j} - \left(\sum_{a\neq j}\mathbf{d}_a\boldsymbol{\Gamma}_{a,I_j}\right)\mathbf{g}_{j,I_j}$
        $\mathbf{d}_j := \mathbf{d}_j/\|\mathbf{d}_j\|_2$
        $\mathbf{g}^{\mathrm{T}}_{j,I_j} := \mathbf{d}_j^{\mathrm{T}}\mathbf{X}_{I_j} - \mathbf{d}_j^{\mathrm{T}}\left(\sum_{a\neq j}\mathbf{d}_a\boldsymbol{\Gamma}_{a,I_j}\right)$
    end
end
```

and the sparse matrix $\boldsymbol{\Gamma}$, but it will not be the same as the original one. For that reason, we will call the reconstructed image as $\mathbf{X}^{\mathrm{rec}}$ instead where

$$\mathbf{X}^{\mathrm{rec}} = \mathbf{D}\boldsymbol{\Gamma}. \tag{4.28}$$

It is worth mentioning that if the patches overlap, one should calculate the average between them after computing $\mathbf{X}^{\mathrm{rec}}$.

4.3 Applications

Aharon et al. [4] used K-SVD in two different applications. Firstly, the authors used face images to train a dictionary. Afterward, they selected one face and deleted some pixels in the image to restore it by filling in the missing pixels by using the dictionary. They compared how the K-SVD performed compared to an over-complete DCT and wavelet dictionary. In addition, they also compared a compressing application using the same methodology. In both applications, the authors showed the K-SVD algorithm outperformed the other methods for a better-reconstructed image with the learned dictionary.

In a different work, Elad and Aharon [29] used the K-SVD method as a denoising tool and showed that it is possible to achieve better results by training the dictionary with the proposed algorithm. Other works in the literature also used the K-SVD algorithm to denoise images. For instance, different authors attenuated noise in seismic data sets [110, 117]. Each of these works proposed minor alterations in the original K-SVD algorithm to tackle the denoising problem. Turquais et al. [110] developed an automatic way to calculate the noise ϵ in the signal $\mathbf{X}$. Zu et al. [117] considered a patch selection procedure to learn the dictionary, and they pointed out that using the most complex patches can generate a better reconstruction of the original signal.

Also, Ravishankar and Bresler [94] and Chen et al. [21] restored magnetic resonance images. Liu and Jafarpour [62] and Khaninezhad et al. [56] parameterized oil reservoir subsurface characteristics (facies, channels, and permeability) through the K-SVD algorithm.

Other more distinct applications of the K-SVD involve face recognition problems [114, 15]. The authors included a discriminative characteristic to the dictionary and added a classification performance into the process.

In summary, the K-SVD algorithm has a broad range of applications, and one can use it in different research areas. However, in this work, the focus is on the algorithm's compression and denoising properties applied to a seismic data set.

Chapter 5

The use of seismic data in reservoir models

Seismic data is a valuable source of information to construct and update reservoir models. 3D Seismic can help identify faults, barriers, reservoir structure and is essential when evaluating a prospect or building a reservoir model. 4D seismic refers to the difference between two seismic surveys conducted at different times and can bring additional information about the reservoir dynamics through production-induced saturation and pressure changes. Besides, 4D seismic is an example of vasts data sets used in history matching. Hence, this chapter shows a specific type of vast data set with valuable usage in history-matching processes.

This chapter introduces 3D seismic data for use in reservoir models and discusses seismic data sets' main characteristics. Then, we explain how one can use 4D seismic data in a history-matching algorithm to update the reservoir models. The main objective is to give a general overview of the practical use of 4D seismic in reservoir management.

5.1 3D Seismic

3D seismic (3DS) is a valuable tool for oil and gas reservoirs as it provides information about the subsurface structure. Artificially elastic waves generated on the surface travels into the subsurface, and at each interface between two different media, part of the waves reflect. Then, one can measure the intensity of the reflected wave on the surface and the travel time. The upper plot in Fig. 5.1 illustrates the 3D seismic data. Here (1) represents the source of the waves, and (2), (3), and (4) are the receivers that capture the reflected waves. Note that the different media can be two distinct lithologies (shale and sandstone, for instance) or the same type of rock but filled with contrasting fluids (e.g., the contact surface between oil and water in sandstone).

As such, 3D seismic helps to determine faults and reservoir boundaries. Furthermore, it is possible to estimate wave velocities and acoustic impedances (subsurface density × wave velocity). These parameters are also known as the rock elastic properties. One can also determine pore volume, lithology, and the fluid present in the reservoir based on 3DS. As a result, many studies used 3DS to construct geologic models for reservoirs [27, 50, 45].

It is essential to highlight the several challenging steps from the seismic-data acquisition to their interpretation, known as seismic processing. Despite being very important for the use of 3DS, seismic data processing is outside the thesis's scope.

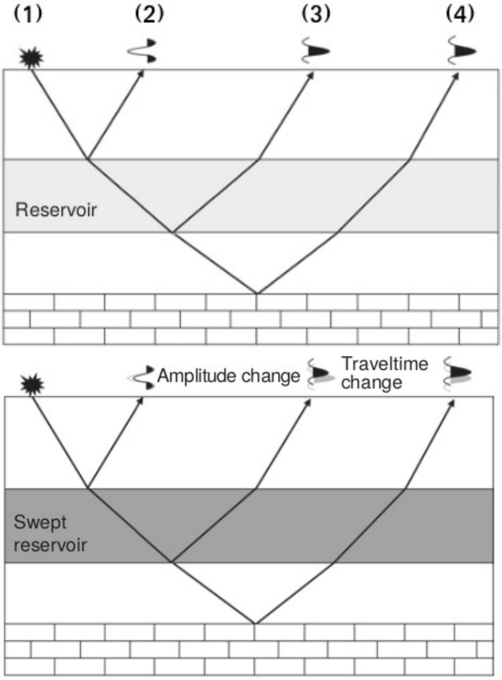


Figure 5.1: 3D Seismic scheme (adapted from [55]): First survey (upper plot); Second survey (lower plot).

5.2 4D Seismic history matching

As the reservoir starts to produce, its equilibrium and dynamics change. For instance, pressure drops or increases, gas comes out of the solution, and water floods parts of the reservoir. Consequently, if one acquires another seismic survey, the result should be different since the reflected waves' amplitude and the travel time change. The lower plot in Fig. 5.1 illustrates a situation where the injected water swept the reservoir. As a result, there is a difference in the seismic amplitude and travel time shown in the receivers (2), (3), and (4). The process of repeatedly taking seismic surveys in time is also known as time-lapse seismic or 4D Seismic (4DS).

The scheme shown in Fig. 5.1 is a simple example that illustrates the potential of using 4DS to understand the reservoir better. Nevertheless, in more complex applications, one must consider some features before using 4DS. Johnston [55] and Lumley and Behrens [67] pointed out three fundamental factors for a successful application of 4DS: repeatability, detectability, and interpretability.

Repeatability means that a seismic survey should be able to be reproduced later in a very similar condition. In other words, two or more surveys at different times should be as similar as possible, so one can verify only the change of the specific parameters that are related to the reservoir production, such as amplitude and travel time. Johnston [55] stated that signal-to-noise ratio, processing similarity, and acquisition geometry are among the factors that can affect repeatability.

The pressure and saturation changes in the reservoir need to reach a certain level

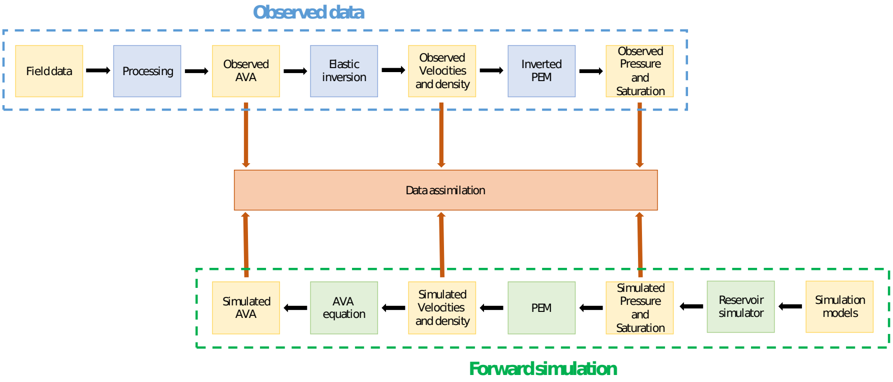


Figure 5.2: Different 4DS domains (based on [49])

to be detected by the time-lapse seismic, referred to as detectability. Therefore, if a reservoir has a high detectability and repeatability, there is a good chance of success in using 4DS.

Finally, interpretability is a consequence of repeatability and detectability, and it is essential for a good interpretation of the 4DS.

There is a need to link 4DS with the oil reservoir simulator for using time-lapse seismic in history matching. This link is challenging as they provide answers in different domains. Usually, commercial reservoir simulators provide simulated pressure and saturation maps of the reservoir, while the observed seismic data is recorded as wave amplitude and travel time. Therefore, one needs to transform these two types of data (simulated and observed) to have a common comparison point.

Fig. 5.2 illustrates the different domains of reservoir simulation models and the observed seismic. The green dashed frame represents the forward simulation of seismic, where one transforms the reservoir simulation response (pressure and saturation maps) into a domain that can be compared with the observed seismic. The blue dashed frame represents the transformation of the acquired seismic data into a domain comparable with the forward simulation. The yellow boxes represent the observed and simulated data, while the blue, green, and orange boxes are the necessary processing steps. We provide more information about each box later in this chapter.

The main point of Fig. 5.2 is to show that one can perform 4DS data assimilation using AVA [72, 73, 69], velocities and density (acoustic impedance) [31, 64, 5, 63, 49, 82], and pressure and saturation or even fluid fronts [57, 59].

However, these transformations can be very challenging. Hence, the first applications of 4DS data assimilation were performed qualitatively. Johnston Johnston [55] highlighted that one could use qualitative 4DS history matching to identify sealing faults, bypassed oil, and baffles. Helgerud et al. [53] applied 4DS history matching to a real field case. They identified a flow baffle in the reservoir through the 4DS and updated the reservoir simulation models' transmissibilities to match this scenario. Other qualitative applications of 4DS include [83, 8].

In addition to the challenge of bringing seismic and simulated data into the same domain, the observed seismic used in history matching usually consists of big data sets. Hence, the computational resources needed to deal with these data sets are high.

The continuous increase in computational power has been fundamental for applying the 4D seismic history matching quantitatively. Besides, the development of methods capable of compressing the 4D seismic signal without losing valuable information about the data also helped increase the use of quantitative 4DS in history matching reservoir models [63, 65, 73].

To better understand the steps to carry out a quantitatively 4DS history matching, the following section contains information about the green boxes, which are necessary to transform simulation model response into simulated seismic data.

5.2.1 Reservoir simulator

One uses several types of information (seismic data, well logging, and cores) to build the simulation models. Additionally, when using ensemble history-matching methods, one needs to define the uncertain parameters to form the initial ensemble of many different model realizations. It is, therefore, necessary to run the reservoir simulator to obtain the simulated pressure and saturation maps and production rates for each realization in the ensemble.

There are many parameters involved in a reservoir simulator, such as the type of simulator (black-oil, compositional), the solution technique (IMPES or Implicit), different grid systems (block-centered point, Voronoi), and many others. As this is not the scope of this work, we refer the readers to the book from Peacemean [90] for more information.

5.2.2 Petro Elastic Models (PEMs)

Petro Elastic Models (PEM), sometimes also referred to as rock physics models, are responsible for transforming pressure and saturation into the rock's elastic properties, such as wave velocity and density.

The wave that penetrates the subsurface has two components: the compressional (P-wave) and the shear (S-wave). Johnston [55] highlighted that P-wave is dependent on the system's resistance to compression and deformation, and density. P-wave is very sensitive to the reservoir rock matrix's compressibility and the fluids in the reservoir. The S-wave depends on the system's resistance to deformation, and it is more sensitive to the density. Additionally, a change in pressure and stress might also alter the porosity or the strength of the grain contacts. Thus, PEMs model complex phenomena, and many factors, including lithology, porosity, pore shape, clay content, fluid type, saturation, pressure, and rock cementation, influence their behaviors. As such, they are sources of errors [31]. Note, however, that these errors can be incorporated as model errors in the history-matching processes [68].

In 4DS history matching, one is more interested in the difference between the two surveys. Therefore, it depends on the reservoir depletion mechanisms and how production is carried out. Johnston [55] provided a summary of what happens to the wave velocities and density depending on each situation, as Table 5.1 shows.

However, the interpretation is not straightforward because the effects can compete with each other. For instance, a gas saturation increase leads to a decrease in the P-wave

Table 5.1: Summary of 4D Seismic changes.

	P-wave velocity	S-wave velocity	Density
Waterflooding	increase	constant	increase
Gas replacing water/oil	decrease	constant	decrease
Pressure depletion/Compaction	increase	increase	-
Pressure increase	decrease	decrease	-

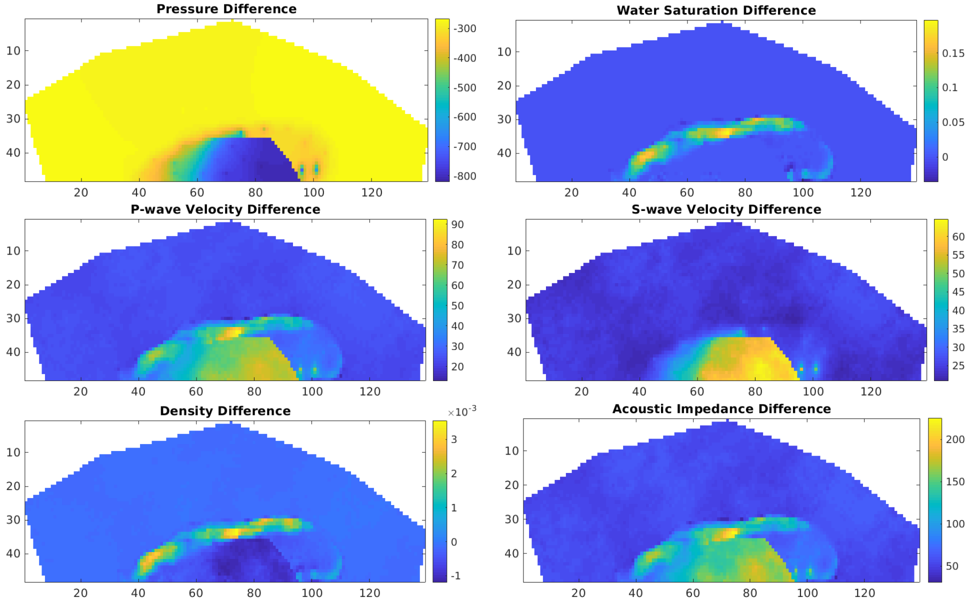


Figure 5.3: Difference between two times: Pressure (top left); Water saturation (top right); P-wave velocity (middle left); S-wave velocity (middle right); Density (lower left); Acoustic Impedance (lower right).

velocity, but this situation might be linked to a pressure depletion, which would increase the velocity. Usually, the saturation effects dominate the changes.

Fig. 5.3 illustrates how the wave velocities and density change with reservoir production. This example is from the Brugge benchmark [73] case in which there is a difference of 2 999 days between the two seismic surveys. The plots show the second layer of the reservoir. As the top-left plot shows, one can see that the pressure decreased in the whole reservoir. Besides, the water saturation increased in some parts of the model, specifically where the injector wells are located (top-right plot). Thus, P- and S-wave velocity increased, as the middle-left and middle-right plots show. For the density map (lower-left plot), the value increased where the water was injected. However, as the oil density decreased around the producer wells because of the pressure depletion, the density also decreased in this region. Note, that this decrease was very small.

A more common variable is the acoustic impedance (density times P- or S-wave velocity). The bottom-right plot in Fig. 5.3 shows the difference between the two times in acoustic impedance using the P-wave velocity. The acoustic impedance value increased in the whole reservoir because of the pressure depletion and waterflooding. The difference at the model's border was small, while the difference was more considerable close to injector wells, where the water saturation increased. It is worth mentioning that the acoustic impedance data usually has the same dimension as the reservoir model. However, one can also combine information from different layers to reduce the size of the data set.

5.2.3 AVA

The waves emitted by the source reflect at each interface, and this reflection is a function of the acoustic impedance of the two media, as Fig. 5.1 shows. Also, they can reach the interfaces at different angles, depending on the position of the source. The higher the angle, the longer is the time required for the wave to travel. However, there is also a difference in the magnitude. Hence, one can use this property to obtain additional information from the reservoir.

The different angles used in the AVA data sets are usually divided into far, mid, and near. There is no specific definition of how to classify different angles [73]. Generally, angles until 15° can be considered as near and angles larger than 20° as far, but it varies from work to work. Traces from near angles are usually more sensitive to P-wave velocities, while traces from far angles are more sensitive to S-wave velocities [71]. Consequently, near angles are more suitable to identify saturation changes, while far angles are more related to pressure changes.

The interpretation of AVA data sets is not straightforward as it is case-dependent. Therefore, to interpret AVA data sets, it is common to perform a sensitivity analysis to check the AVA values' differences for different scenarios. A critical characteristic of the AVA data set is that it is recorded in the time-domain and comprises under- and overburden parts of the reservoir. Consequently, its size is much bigger than the acoustic impedance, and some parts of the data contain information that is not useful for the history matching process (under- and overburden parts). If one takes Fig. 5.4 as an example [73] , one will use the whole cube as the measured data. Nevertheless, only the region with stronger signals is related to the reservoir.

Moreover, Luo et al. [73] stated that one does not need to perform the seismic data inversion to generate acoustic impedance when using AVA. Hence, one can avoid introducing additional errors into the workflow, as this inversion process involves uncertainties.

The use of sparse representation methods can help 4D seismic history matching if one uses AVA data set as measured data because they can help reduce the number of data points. These methods can also assist in selecting only the information contents related to the reservoir while discarding those related to under- and overburdens.

Concerning the type of localization method used, either distance-dependent or correlation-based localization can be adopted if one uses acoustic impedance and does not perform any sparse representation method. On the other hand, if one uses AVA or a sparse representation method, it is only possible to use correlation-based localization since the observed data and the uncertain model parameters are in a different domain and scale.

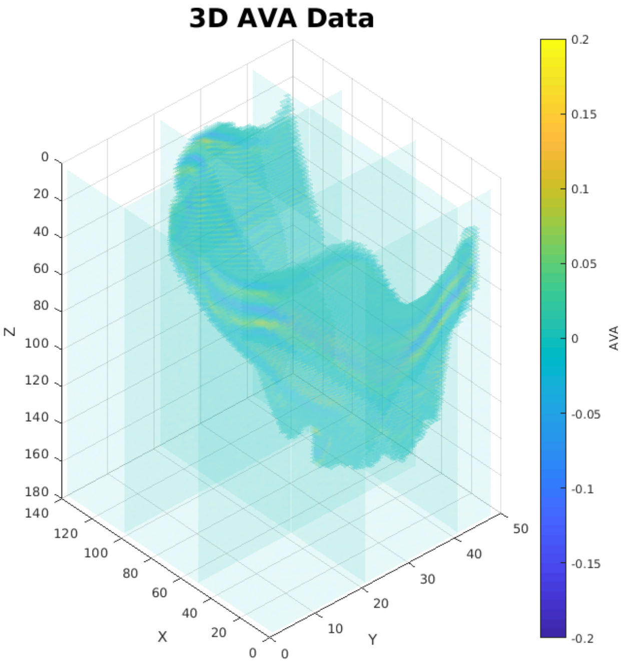


Figure 5.4: 3D AVA data (observed data): far trace in the base survey (day 1). The color bar in the right indicates the values of AVA data points.

5.3 4D Seismic history matching applications and practical issues

Emerick [31] performed a 4D seismic history matching using real data from an offshore field in Brazil. He used the Ensemble Smoother with Multiple Data Assimilation (ESMDA) to history match acoustic impedance and highlighted some issues regarding the 4D seismic. Firstly, he said that the definition of the critical length (L) when using the distance-dependent localization is difficult, and he used a length of 1 000 meters for all data points to simplify the application. Secondly, Emerick [31] had three different time surveys. He used the first survey separately, and the differences between the second and the first surveys and the differences between the third and the second surveys. He stated that when the base survey was assimilated, the porosity field had a more substantial update. In contrast, when he assimilated the differences, the permeability field had a stronger update instead. He justified that because porosity is one of the inputs of the PEMs, consequently, the base survey affected more porosity updates. Furthermore, since the differences consider changes in the impedance and, consequently, saturation and pressure changes, this assimilation affected the permeability maps more. In the end, the author concluded that the inclusion of seismic data in the history-matching process improved data match but underestimated the variability.

Luo et al. [73] performed a 4DS history matching using AVA as the observed data. They determined the noise involved in the seismic data set through wavelet coefficients. Note that the noise definition is essential for ensemble-based methods, as it is used in the matrix $\mathbf{C}_\mathrm{d}$. In this application, the noise was uncorrelated (white noise), but it is also possible to use wavelet coefficients to determine correlated errors (colored noise) as in [64]. Moreover, both authors also used wavelet coefficients to select only the leading coefficients to reduce the seismic data set size. Luo et al. [73] history matched both well production and seismic data in a benchmark case, and they achieved a final ensemble closer to the true solution. Nevertheless, they did not use any localization scheme, and they reported a very strong variability underestimation.

Later, Luo and Bhakta [69] used the same application as before. By introducing the correlation-based Kalman gain localization, they fixed the excessive variability reduction without impairing the final ensemble representation.

Leeuwenburgh and Arts [59] used waterfronts detected from 4D seismic data to condition the reservoir models. The authors reduced the number of observed data and achieved a good match regarding the fluid fronts.

Liu and Grana [63] were also concerned with the size of the seismic data set. Hence, they proposed a deep learning method to reduce the seismic data set size. The authors reduced the original data set to about 0.20% of its original size using a deep convolutional autoencoder. Nevertheless, the computational time concerning deep learning is usually very high. For more applications of 4D seismic history matching, the readers are referred to [5, 1, 44, 57, 95].

Chapter 6

Summary and Outlook

In the previous chapters, we introduced history-matching problems and how one can deal with vast data sets in such problems. In Chapter 2, we discussed ensemble-based history matching processes and the main challenges that arise when one faces vast data sets. Chapter 3 presented localization methods, which alleviates the problem of spurious correlation and underestimating the ensemble variance when using ensemble methods. We also showed one localization method, known as local analysis, which is suitable for dealing with big models and data sets. In Chapter 4, we introduced sparse representation methods with a focus on Dictionary Learning and K-SVD as a method to reduce the size of the data set one is dealing with. Finally, we finished with the description of the 4D seismic history matching problem, which represents a valuable application for the oil and gas reservoir community, and it consists of problems with vast data sets.

The concepts and ideas introduced before are relevant to the methodology we developed in this work. The main objective of our methodology is to perform history matching considering vast data sets efficiently. The next part of this work consists of four papers that present our methodology and main results. In Paper A, we developed a K-SVD application to compress a 4D seismic data set size. In Paper B, we used the K-SVD application developed in Paper A together with a new formulation of the Iterative Ensemble Smoother Regularized Levenberg–Marquardt in a 4D seismic history matching framework. To improve the methodology, we develop a novel local analysis scheme in Paper C that better handles the observed data set's big size. Finally, in Paper D, we also applied local analysis in a 4D seismic history matching problem, but now we use the Subspace Ensemble Randomized Maximum Likelihood. In the following, we present more information about the four articles.

6.1 Summary of the papers

Paper A

Title:	Sparse Representation of 4D Seismic Signal based on Dictionary Learning
Authors:	Ricardo V. Soares, Xiaodong Luo, Geir Evensen
Conference:	SPE Norway One Day Seminar
DOI:	10.2118/195599-MS

In the first paper, we used the Dictionary Learning Method through the K-SVD algorithm to sparsely represent a synthetic seismic data set. The paper's main objective is to compress the original data set as much as possible without losing the seismic signal's main information. Moreover, we were also concerned about the computational time spent during the application of the algorithm. First, we evaluated the impact of the two original approaches proposed by Elad and Aharon [4] to solve the problem: error-constrained and sparsity-constrained. Subsequently, we analyzed how each parameter involved in the algorithm affects the data set reduction, retained information, and computational time. These parameters include dictionary size, patch size, training data set size, and the number of iterations. The study case used in this work was a benchmark named Brugge [91, 73]. We started the study using a 2D version of the Amplitude versus Angle (AVA) seismic, and then we expanded to the 3D and 4D data sets. Most of the applications of dictionary learning methods are related to image restoration and denoising. Therefore, Paper A's main contribution is the application of K-SVD in a dimensionality-reduction problem using a seismic data set.

Paper B

Title:	4D Seismic History Matching: Assessing the use of a Dictionary Learning Based Sparse Representation Method
Authors:	Ricardo V. Soares, Xiaodong Luo, Geir Evensen, Tuhin Bhakta
Journal:	Journal of Petroleum Science and Engineering
DOI:	10.1016/j.petrol.2020.107763

In Paper B, we developed a new version of the IES-RML [75], in which the matrix to be inverted is smaller, and one does not need to apply the truncated singular value decomposition. To deal with the big size of the AVA seismic data set, we compared three different approaches. In the first one, we applied K-SVD once to sparsely represent the observed seismic data (named SSD), focusing on compressing the data set without losing the original signal's main characteristics. In the second approach (named DSSD), we used the K-SVD twice, one to denoise the original signal without concerning data compression and the other to reduce the data set's size. However, both approaches incurred an increase in computational time. Hence, in our last approach, we projected the observed data into a smaller subspace spanned by the ensemble to deal with the big size of it (named PSD). As a result, the measured data set had the same size as the ensemble, and there was no additional computational time added to the assimilation process.

Results showed that we achieved a weaker update of the reservoir uncertain parameters because we consider the complete seismic data set in PSD (including the non-reservoir parts). SSD and DSSD, on the other hand, showed stronger updates, which generated better final reservoir models. The denoising process in DSSD helped achieve a better representation of the original seismic signal, generating better final reservoir models. Nevertheless, the difference between SSD and DSSD was small.

The reduction achieved with K-SVD was significant for the SSD case, where we used only 0.27% of the original coefficients in the AVA data set. For the DSSD case, this number was about 0.55%. Note that even though there was a significant compression, the absolute number can still be large. For the Brugge benchmark case, the DSSD case used 38 876 coefficients, which can be challenging to handle.

Paper C

Title:	Handling Big Models and Big Data Sets in History Matching Problems Through an Adaptive Local Analysis Scheme
Authors:	Ricardo V. Soares, Xiaodong Luo, Geir Evensen, Tuhin Bhakta
Journal:	SPE Journal
DOI:	10.2118/204221-PA

In Paper B, we reduced the original seismic data set through K-SVD to either 0.27% or 0.55% of the original size. Nevertheless, if one also considers the number of uncertain model parameters, one would need to store and manipulate a Kalman gain matrix in a high dimension. Thus, in this third paper, we developed a local analysis scheme that deals with spurious correlation, underestimation of the ensemble variability, and the big sizes of both the measurement data and the uncertain model parameters.

Previous works about local analysis used the distance-dependent approach to select which data points were used to update each local group. Nevertheless, because we have sparsely represented data, we could not use this approach. We developed a new correlation-based procedure to select the data used to update each local group. In this novel method, we proposed a different tapering function based on the Gaspari and Cohn equation [47].

As in Paper B, we used the IES-RML method, the Brugge benchmark case, and AVA seismic data as the observed measurement. We performed local analysis with the correlation-based approach using both hard and soft threshold. We also analyzed how local groups with different sizes affect the history-matching process. Local analysis assisted in better management of the storage and memory consumption due to the big matrix sizes. Besides, with the new method developed, we achieved faster convergence and a final ensemble closer to the reference (true) case.

Paper D

Title:	Subspace Ensemble Randomized Maximum Likelihood with Local Analysis for Time-Lapse-Seismic-Data Assimilation
Authors:	Gilson M.S. Neto, Ricardo V. Soares, Geir Evensen, Alessandra Davolio, Denis J. Schiozer
Journal:	SPE Journal
DOI:	10.2118/205029-PA

In the Subspace Ensemble Randomized Maximum Likelihood (SEnRML) formulation [42], one does not compute the Kalman Gain. Consequently, the local analysis scheme is not the same as in IES-RML. In this paper, we applied local analysis using SEnRML, and because we used acoustic impedance as our seismic data set, we compared the distance-dependent and the correlation-based approach. Differently from the previous paper and its original formulation [69], the correlation-based approach developed in this work does not depend on the number of realizations in the ensemble. Moreover, to reduce the seismic data set's size, we projected it into the ensemble subspace. Because we used acoustic impedance as observed data, the whole data set concerns reservoir parts. Thus, we did not observe a significant loss of information during the projection.

Results showed that the proposed scheme presented comparable results with other methods better known in the literature, such as the Ensemble Smoother with Multiple Data Assimilation (ESMDA) [33] with Kalman gain localization. However, SEnRML dealt better with big data set as the memory requirement and CPU time scales linearly with the observed data's size.

6.2 Outlook

We developed different methodologies to deal with big data sets in history matching in this work, focusing on using 4D seismic data as the observed data. The study cases in this work were synthetic cases that mimic the characteristics and behavior of real reservoirs. Even though the cases are complex, they do not share all the complexity of an actual reservoir application. These complexities may include even larger data sets and models, other types of model parameter uncertainties (water relative permeability, well index), and incorporation of production well data. Therefore, applying the proposed methodologies to more complicated problems can bring additional challenges to the history matching process, and it can be the focus of future research activities.

When applying the time-lapse seismic in history matching, we have not evaluated how to use this data better to achieve the best reservoir characterization. In papers B and C, we applied near and far traces of the AVA data sets in three time surveys, while in paper D, we used the difference between the acoustic impedance from two different time surveys. Since we were concerned about using vast data sets in history matching, we did not focus on this matter. However, it can have an impact on the updated simulation models.

Finally, in papers C and D, we developed a local analysis scheme with a correlation-based approach for two different ensemble methods. The developed methods in both works were specific for each tested case. Thus, it would be beneficial to extend them to a more general case in a future work.

Bibliography

[1] S. Aanonsen, I. Aavatsmark, T. Barkve, R. Gonardm, O. Gosselin, M. Kolasinki, and H. Reme, *Effect of scale dependent data correlations in an integrated history matching loop combining production data and 4d seismic data*, in SPE Reservoir Simulation Symposium, 2003. SPE-79665-MS.

[2] S. I. Aanonsen, G. Naevdal, D. S. Oliver, A. Reynolds, and B. Valles, *Ensemble Kalman Filter in Reservoir Engineering – A Review*, SPE Journal, SPE-117274-PA, 14(3) (2009), pp. 393–412.

[3] B. Agarwal, H. Hermansen, J. Sylte, and L. Thomas, *Reservoir characterization of ekofisk field: A giant, fractured chalk reservoir in the norwegian north sea–history match*, SPE Reservoir Evaluation and Engineering, 3 (2000), pp. 534 – 543.

[4] M. Aharon, M. Elad, and A. Bruckstein, *K-svd: An algorithm for designing overcomplete dictionaries for sparse representation*, IEEE Transactions on Signal Processing, 54 (2006), pp. 4311–4322.

[5] M. Alfonzo and D. S. Oliver, *Seismic data assimilation with an imperfect model*, Comput Geosci, (2019), pp. 889 – 905.

[6] T. Anderson, *An Introduction to Multivariate Statistical Analysis*, John Wiley & Sons, Inc., 3 ed., 2003.

[7] S. Beckouche and J. Ma, *Simultaneous dictionary learning and denoising for seismic data*, GEOPHYSICS, 79 (2014), pp. A27–A31.

[8] R. Behrens, R. Condon, W. Haworth, M. Bergeron, and C. Ecker, *4d seismic monitoring of water influx at bay marchand: The practical use of 4d in an imperfect world*, SPE Reservoir Evaluation and Engineering, 5 (2002), pp. 410 – 420. SPE-79961-PA.

[9] K. Brusdal, J. Brankart, G. Halberstadt, G. Evensen, P. Brasseur, P. J. van Leeuwen, E. Dombrowsky, and J. Verron, *An evaluation of ensemble based assimilation methods with a layered OGCM*, J. Marine. Sys., 40-41 (2003), pp. 253–289.

[10] D. Cai, C. Zhang, and X. He, *Unsupervised feature selection for multi-cluster data*, in Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '10, New York, NY, USA, 2010, Association for Computing Machinery, pp. 333–342.

[11] S. W. Canchumuni, A. A. Emerick, and M. A. C. Pacheco, *History matching geological facies models based on ensemble smoother and deep generative models*, Journal of Petroleum Science and Engineering, 177 (2019), pp. 941 – 958.

[12] Y. Chang, G. Lorentzen, R. Nævdal, and T. Feng, *Olympus optimization under geological uncertainties*, Comput Geosci, (2021), pp. 475–501.

[13] C. Chen, W. Wang, G. Li, and A. Reynolds, *Closed-loop reservoir management on the brugge test case*, Comput Geosci, 14 (2010), pp. 691–703.

[14] S. S. Chen, D. L. Donoho, and M. A. Saunders, *Atomic decomposition by basis pursuit*, SIAM Review, 43 (2001), pp. 129–159.

[15] Y. Chen, N. M. Nasrabadi, and T. D. Tran, *Hyperspectral image classification using dictionary-based sparse representation*, IEEE Transactions on Geoscience and Remote Sensing, 49 (2011), pp. 3973–3985.

[16] Y. Chen and D. S. Oliver, *Cross–covariances and localization for enkf in multiphase flow data assimilation*, Comput Geosci, 14 (2010), pp. 579–601.

[17] Y. Chen and D. S. Oliver, *Ensemble randomized maximum likelihood method as an iterative ensemble smoother*, Math. Geosci., 44 (2012), pp. 1–26.

[18] ——, *Levenberg-Marquardt forms of the iterative ensemble smoother for efficient history matching and uncertainty quantification*, Computat Geosci, 17 (2013), pp. 689–703.

[19] Y. Chen and D. S. Oliver, *History Matching of the Norne Full-Field Model With an Iterative Ensemble Smoother*, SPE Reservoir Evaluation & Engineering, 17 (2014), pp. 244–256. SPE-164902-PA.

[20] ——, *Localization and regularization for iterative ensemble smoothers*, Comput Geosci, 21 (2017), pp. 13–30.

[21] Y. Chen, X. Ye, and F. Huang, *A novel method and fast algorithm for mr image reconstruction with significantly under-sampled data*, Inverse Problems & Imaging, 4 (2010), p. 223.

[22] G. Davis, S. Mallat, and M. Avellaneda, *Adaptive greedy approximations*, Constructive Approximation, 13 (1997), pp. 57–98.

[23] A. Davolio and D. J. Schiozer, *Probabilistic seismic history matching using binary images*, Journal of Geophysics and Engineering, 15 (2018), pp. 261–274.

[24] D. Devegowda, E. Arroyo-Negrete, A. Datta-Gupta, and S. G. Douma, *Efficient and robust reservoir model updating using ensemble Kalman filter with sensitivity-based covariance localization*, in SPE Reservoir Simulation Symposium, 2007. SPE 106144.

[25] D. L. Donoho and I. M. Johnstone, *Ideal spatial adaptation by wavelet shrinkage*, Biometrika, 81 (1994), pp. 425–455.

[26] ——, *Adapting to unknown smoothness via wavelet shrinkage*, Journal of the American Statistical Association, 90 (1995), pp. 1200–1224.

[27] O. Dubrule, M. Thibaut, P. Lamy, and A. Haas, *Geostatistical reservoir characterization constrained by 3d seismic data*, Petroleum Geoscience, 4 (1998), pp. 121–128.

[28] M. Elad, *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing*, Springer, first ed., 2010.

[29] M. Elad and M. Aharon, *Image denoising via sparse and redundant representations over learned dictionaries*, IEEE Transactions on Image Processing, 15 (2006), pp. 3736–3745.

[30] A. Emerick and A. Reynolds, *History Matching a Field Case Using the Ensemble Kalman Filter With Covariance Localization*, SPE Reservoir Evaluation & Engineering, 14 (2011), pp. 443–452. SPE-141216-PA.

[31] A. A. Emerick, *Analysis of performance of ensemble-based assimilation of production and seismic data*, Journal of Petroleum science and engineering, 139 (2016), pp. 219–239.

[32] A. A. Emerick, *Analysis of the performance of ensemble-based assimilation of production and seismic data*, Journal of Petroleum Science and Engineering, 139 (2016), pp. 219–239.

[33] A. A. Emerick and A. C. Reynolds, *Ensemble smoother with multiple data assimilation*, Computers and Geosciences, 55 (2013), pp. 3–15.

[34] K. Engan, S. O. Aase, and J. H. Husøy, *Multi-frame compression: theory and design*, Signal Processing, 80 (2000), pp. 2121 – 2140.

[35] G. Evensen, *Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics*, Journal of Geophysical Research, 99 (1994), pp. 10143–10162.

[36] G. Evensen, *The ensemble Kalman filter: Theoretical formulation and practical implementation*, Ocean Dynamics, 53 (2003), pp. 343–367.

[37] G. Evensen, *Data Assimilation: The Ensemble Kalman Filter*, Springer, 2nd ed., 2009.

[38] G. Evensen, *The Ensemble Kalman Filter for Combined State and Parameter Estimation: Monte Carlo techniques for data assimilation in large systems*, IEEE Control Systems, (2009), pp. 83 – 104.

[39] G. Evensen, *Analysis of iterative ensemble smoothers for solving inverse problems*, Computat Geosci, 22 (2018), pp. 885–908.

[40] G. Evensen, *Accounting for model errors in iterative ensemble smoothers*, Comput Geosci, 23 (2019), pp. 761–775.

[41] G. Evensen, *Formulating the history matching problem with consistent error statistics*, Computat Geosci, (2021), p. 26.

[42] G. Evensen, P. Raanes, A. Stordal, and J. Hove, *Efficient implementation of an iterative ensemble smoother for data assimilation and reservoir history matching*, Frontiers in Applied Mathematics and Statistics, 5 (2019), p. 47.

[43] A. Fahimuddin, S. I. Aanonsen, and J.-A. Skjervheim, *4d seismic history matching of a real field case with enkf: Use of local analysis for model updating*, in SPE Annual Technical Conference and Exhibition, 2010. SPE 134894.

[44] ——, *4d seismic history matching of a real field case with enkf: Use of local analysis for model updating*. SPE 134894, 2010.

[45] J. L. Fatti, G. C. Smith, P. J. Vail, P. J. Strauss, and P. R. Levitt, *Detection of gas in sandstone reservoirs using avo analysis: A 3-d seismic case history using the geostack technique*, GEOPHYSICS, 59 (1994), pp. 1362–1376.

[46] R. R.-M. Fonseca, B. Chen, J. D. Jansen, and A. Reynolds, *A stochastic simplex approximate gradient (stosag) for optimization under uncertainty*, International Journal for Numerical Methods in Engineering, 109 (2017), pp. 1756–1776.

[47] G. Gaspari and S. E. Cohn, *Construction of correlation functions in two and three dimensions*, Quarterly Journal of the Royal Metereological Society, 125 (1999), pp. 723–757.

[48] I. F. Gorodnitsky and B. D. Rao, *Sparse signal reconstruction from limited data using focuss: a re-weighted minimum norm algorithm*, IEEE Transactions on Signal Processing, 45 (1997), pp. 600–616.

[49] O. Gosselin, S. Aanonsen, I. Aavatsmark, A. Cominelli, R. Gonard, M. Kolasinski, F. Ferdinandi, L. Kovacic, and K. Neylon, *History matching using time-lapse seismic (HUTS)*, in SPE Annual Technical Conference and Exhibition, 2003. SPE-84464-MS.

[50] J. Grötsch and C. Mercadier, *Integrated 3-D Reservoir Modeling Based on 3-D Seismic: The Tertiary Malampaya and Camago Buildups, Offshore Palawan, Philippines1*, AAPG Bulletin, 83 (1999), pp. 1703–1728.

[51] T. M. Hamill, J. S. Whitaker, and C. Snyder, *Distance-dependent filtering of background error covariance estimates in an ensemble kalman filter*, Monthly Weather Review, 129 (2001), pp. 2776–2790.

[52] C. He, J. Xing, J. Li, Q. Yang, and R. Wang, *A new wavelet threshold determination method considering interscale correlation in signal denoising*, Mathematical Problems in Engineering, (2015), p. 9.

[53] M. B. Helgerud, A. C. Miller, D. H. Johnston, M. S. Udoh, B. G. Jardine, C. Harris, and N. Aubuchon, *4d in the deepwater gulf of mexico: Hoover, madison and marshall fields*, The Leading Edge, 30 (2011), pp. 1008–1018.

[54] J.-D. Jansen, R. Brouwer, and S. G. Douma, *Closed Loop Reservoir Management*, SPE-119098-MS, (2009).

[55] D. Johnston, *Practical applications of Time-lapse Seismic Data*, Society of Exploration Geophysicists (SEG), 2013.

[56] M. M. Khaninezhad, B. Jafarpour, and L. Li, *Sparse geologic dictionaries for subsurface flow model calibration: Part i. inversion formulation*, Advances in Water Resources, 39 (2012), pp. 106 – 121.

[57] V. Kretz, B. Valles, and L. Sonneland, *Fluid front history matching using 4d seismic and streamline simulation*, in SPE Annual Technical Conference and Exhibition, Society of Petroleum Engineers, 2004. SPE-90136-MS.

[58] K. Kreutz-Delgado, J. F. Murray, B. D. Rao, K. Engan, T.-W. Lee, and T. J. Sejnowski, *Dictionary learning algorithms for sparse representation*, Neural Computation, 15 (2003), pp. 349–396.

[59] O. Leeuwenburgh and R. Arts, *Distance parameterization for efficient seismic history matching with the ensemble kalman filter*, Comput Geosci, 18 (2014), pp. 535 – 548.

[60] S. Lesage, R. Gribonval, F. Bimbot, and L. Benaroya, *Learning unions of orthonormal bases with thresholded singular value decomposition*, in Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing, 2005., vol. 5, March 2005, pp. v/293–v/296 Vol. 5.

[61] M. S. Lewicki and T. J. Sejnowski, *Learning overcomplete representations*, Neural Computation, 12 (2000), pp. 337–365.

[62] E. Liu and B. Jafarpour, *Learning sparse geologic dictionaries from low-rank representations of facies connectivity for flow model calibration*, Water Resources Research, 49 (2013), pp. 7088–7101.

[63] M. Liu and D. Grana, *Time-lapse seismic history matching with iterative ensemble smoother and deep convolutional autoencoder*, GEOPHYSICS, 85 (2020), pp. M15–M31.

[64] R. Lorentzen, T. Bhakta, D. Grana, X. Luo, R. Valestrand, and G. Naevdal, *Simultaneous assimilation of production and seismic data: application to the norne field*, Comput Geosci, 24 (2020), pp. 907–920.

[65] R. Lorentzen, X. Luo, T. Bhakta, and R. Valestrand, *History matching the full Norne field model using seismic and production data*, SPE Journal, 24 (2019), pp. 1452 – 1467. SPE-194205-PA.

[66] R. J. Lorentzen, G. Nævdal, and A. C. V. M. Lage, *Tuning of parameters in a two-phase flow model using an ensemble Kalman filter*, Int. Jour. of Multiphase Flow, 29 (2003), pp. 1283–1309.

[67] D. Lumley and R. Behrens, *Practical issues of 4d seismic reservoir monitoring: what an engineer needs to know*, SPE Reservoir Evaluation and Engineering, 1 (1998), pp. 528 – 538. SPE-53004-PA.

[68] X. Luo, *Ensemble-based kernel learning for a class of data assimilation problems with imperfect forward simulators*, PLOS ONE, 14 (2019), pp. 1–40.

[69] X. Luo and T. Bhakta, *Automatic and adaptive localization for ensemble-based history matching*, Journal of Petroleum Science and Engineering, (2019), p. 106559.

[70] X. Luo and T. Bhakta, *Automatic and adaptive localization for ensemble-based history matching*, Journal of Petroleum Science and Engineering, 184 (2020), p. 106559.

[71] X. Luo, T. Bhakta, M. Jakobsen, and G. Nævdal, *An Ensemble 4D Seismic History Matching Framework with Sparse Representation Based on Wavelet Multiresolution Analysis*, SPE Journal, (2016), pp. 985–1010. SPE-180025-PA.

[72] X. Luo, T. Bhakta, M. Jakobsen, and G. Naevdal, *An ensemble 4D-seismic history-matching framework with sparse representation based on wavelet multiresolution analysis*, SPE Journal, 22 (2017), pp. 985–1010.

[73] X. Luo, T. Bhakta, M. Jakobsen, and G. Nævdal, *Efficient big data assimilation through sparse representation: A 3D benchmark case study in petroleum engineering*, PLOS ONE, 13 (2018), pp. 1–32.

[74] X. Luo, R. Lorentzen, R. Valestrand, and G. Evensen, *Correlation-Based Adaptive Localization for Ensemble-Based History Matching: Applied To the Norne Field Case Study*, SPE Reservoir Evaluation & Engineering, 22 (2019), pp. 1084–1109. SPE-191305-PA.

[75] X. Luo, A. S. Stordal, R. J. Lorentzen, and G. Nævdal, *Iterative Ensemble Smoother as an Approximate Solution to a Regularized Minimum-Average-Cost Problem: Theory and Applications*, SPE Journal, SPE-176023-PA, 20 (2015), pp. 962–982.

[76] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, *Online dictionary learning for sparse coding*, in Proceedings of the 26th Annual International Conference on Machine Learning, ICML '09, New York, NY, USA, 2009, ACM, pp. 689–696.

[77] J. Mairal, M. Elad, and G. Sapiro, *Sparse representation for color image restoration*, IEEE Transactions on Image Processing, 17 (2008), pp. 53–69.

[78] C. Maschio and D. J. Schiozer, *Probabilistic history matching using discrete Latin Hypercube sampling and nonparametric density estimation*, Journal of Petroleum Science and Engineering, 147 (2016), pp. 98–115.

[79] G. Nævdal, T. Mannseth, and E. Vefring, *Near well reservoir monitoring through ensemble Kalman filter*, in SPE/DOE Improved Oil recovery Symposium, 2002. SPE-75235-MS.

[80] D. Needell and R. Vershynin, *Uniform uncertainty principle and signal recovery via regularized orthogonal matching pursuit*, Found Comput Math, 9 (2009), pp. 317–334.

[81] L. Nerger, T. Janji, J. Schröter, and W. Hiller, *A regulated localization scheme for ensemble-based kalman filters*, Quarterly Journal of the Royal Meteorological Society, 138 (2012), pp. 802–812.

[82] D. Obidegwu, R. Chassagne, and C. MacBeth, *Seismic assisted history matching using binary maps*, Journal of Natural Gas Science and Engineering, 42 (2017), pp. 69–84.

[83] A. O'Donovan, S. Smith, and P. Kristiansen, *Foinaven 4d seismic - dynamic reservoir parameters and reservoir management*, in SPE Annual Technical Conference and Exhibition, Society of Petroleum Engineers, 2000. SPE-63294-MS.

[84] P. R. Oke, P. Sakov, M. L. Cahill, J. R. Dunn, R. Fiedler, D. A. Griffin, J. V. Mansbridge, K. R. Ridgway, and A. Schiller, *Towards a dynamically balanced eddy-resolving ocean reanalysis: Bran3*, Ocean Modelling, 67 (2013), pp. 52 – 70.

[85] G. S. Oliveira, A. O. Soares, D. J. Schiozer, and C. Maschio, *Reducing uncertainty in reservoir parameters combining history matching and conditioned geostatistical realizations*, Journal of Petroleum Science and Engineering, 156 (2017), pp. 75–90.

[86] D. S. Oliver and Y. Chen, *Recent progress on reservoir history matching: a review*, Computat Geosci, 15 (2011), pp. 185–221.

[87] B. A. Olshausen and D. J. Field, *Sparse coding with an overcomplete basis set: A strategy employed by v1?*, Vision Research, 37 (1997), pp. 3311 – 3325.

[88] E. Ott, B. R. Hunt, I. Szunyogh, A. V. Zimin, E. J. Kostelich, E. J. Corazza, E. Kalnay, D. J. Patil, and J. A. Yorke, *A local ensemble Kalman filter for atmospheric data assimilation*, Tellus, 56A (2004), pp. 415–428.

[89] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad, *Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition*, in Proceedings of 27th Asilomar Conference on Signals, Systems and Computers, 1993, pp. 40–44 vol.1.

[90] D. W. Peacemean, *Fundamentals of Numerical Reservoir Simulation*, Elsevier, first ed., 1977.

[91] L. Peters, R. Arts, G. Brouwer, C. Geel, S. Cullick, R. J. Lorentzen, Y. Chen, N. Dunlop, F. C. Vossepoel, R. Xu, P. Sarma, A. H. Alhuthali, and A. Reynolds, *Results of the Brugge Benchmark Study for Flooding Optimization and History Matching*, SPE Reservoir Evaluation & Engineering, (2010). SPE-119094-PA.

[92] P. N. Raanes, A. S. Stordal, and G. Evensen, *Revising the stochastic iterative ensemble smoother*, Nonlin. Processes Geophys, 26 (2019), pp. 325–338.

[93] P. H. Ranazzi and M. A. Sampaio, *Influence of the kalman gain localization in adaptive ensemble smoother history matching*, Journal of Petroleum Science and Engineering, 179 (2019), pp. 244 – 256.

[94] S. Ravishankar and Y. Bresler, *Mr image reconstruction from highly undersampled k-space data by dictionary learning*, IEEE Transactions on Medical Imaging, 30 (2011), pp. 1028–1041.

[95] F. Roggero, O. Lerat, D. Ding, P. Berthet, C. Bordenave, F. Lefeuvre, and P. Perfetti, *History matching of production and 4d seismic data: Application to the girassol field, offshore angola*, Oil Gas Sci. Technol. - Rev. IFP Energies nouvelles, 67 (2012), pp. 237–262.

[96] P. J. Rousseeuw and C. Croux, *Alternatives to the median absolute deviation*, Journal of the American Statistical Association, 88 (1993), pp. 1273–1283.

[97] R. Rubinstein, A. M. Bruckstein, and M. Elad, *Dictionaries for sparse representation modeling*, Proceedings of the IEEE, 98 (2010), pp. 1045–1057.

[98] R. Rubinstein, M. Zibulevsky, and M. Elad, *Efficient Implementation of the K-SVD Algorithm using Batch Orthogonal Matching Pursuit*, tech. rep., Computer Science Department, Technion - Israel Institute of Technology, 2008.

[99] P. Sakov and L. Bertino, *Relation between two common localization methods for the EnKF*, Computational Geosciences, 15 (2011), pp. 225–237.

[100] P. Sarma, L. Durlofsky, K. Aziz, and W. Chen, *A new approach to automatic history matching using kernel pca*, in SPE Reservoir Simulation Symposium, 2007. SPE-106176-MS.

[101] Schiozer, Denis José, de Souza dos Santos, Antonio Alberto, de Graça Santos, Susana Margarida, and von Hohendorff Filho, João Carlos, *Model-based decision analysis applied to petroleum field development and management*, Oil Gas Sci. Technol. - Rev. IFP Energies nouvelles, 74 (2019), p. 46.

[102] J.-A. Skjervheim, G. Evensen, S. I. Aanonsen, B. O. Ruud, and T. A. Johansen, *Incorporating 4D seismic data in reservoir simulation models using ensemble Kalman filter*, SPE Journal, SPE 95789-PA, 12(3) (2007), pp. 282–292.

[103] J.-A. Skjervheim, G. Evensen, J. Hove, and J. Vabø, *An ensemble smoother for assisted history matching*, in SPE Reservoir Simulation Symposium, 2011. SPE-141929-MS.

[104] R. V. Soares, C. Maschio, and D. J. Schiozer, *Applying a localization technique to kalman gain and assessing the influence on the variability of models in history matching*, Journal of Petroleum Science and Engineering, 169 (2018), pp. 110 – 125.

[105] M. Thompson, M. Andersen, R. Elde, S. Roy, and S. Skogland, *Delivering permanent reservoir monitoring for snorre and grane*, in Third EAGE Workshop on Permanent Reservoir Monitoring 2015, vol. 2015, European Association of Geoscientists & Engineers, 2015, pp. 1–5.

[106] R. Tibshirani, *Regression shrinkage and selection via the lasso*, Journal of the Royal Statistical Society. Series B Methodological, 58 (1996), pp. 267–288.

[107] J. A. Tropp, *Greed is good: algorithmic results for sparse approximation*, IEEE Transactions on Information Theory, 50 (2004), pp. 2231–2242.

[108] J. A. Tropp, A. C. Gilbert, and M. J. Strauss, *Algorithms for simultaneous sparse approximation. part i: Greedy pursuit*, Signal Processing, 86 (2006), pp. 572–588. Sparse Approximations in Signal and Image Processing.

[109] P. Turquais, *Dictionary Learning and Sparse Representations for Denoising and Reconstruction of Marine Seismic Data*, PhD thesis, University of Oslo, Oslo, 2018.

[110] P. Turquais, E. G. Asgedom, and W. Söllner, *A method of combining coherence-constrained sparse coding and dictionary learning for denoising*, GEOPHYSICS, 82 (2017), pp. V137–V148.

[111] P. J. van Leeuwen and G. Evensen, *Data assimilation and inverse methods in terms of a probabilistic formulation*, Monthly Weather Review, 124 (1996), pp. 2898–2913.

[112] L. Wang and D. Oliver, *Fast robust optimization using bias correction applied to the mean model*, Comput Geosci, (2021), pp. 475–501.

[113] M. Williams, J. Keating, and M. Barghouty, *The stratigraphic method: A structured approach to history matching complex simulation models*, SPE Reservoir Evaluation and Engineering, 1 (1998), pp. 169 – 176. SPE-38014-PA.

[114] Q. Zhang and B. Li, *Discriminative k-svd for dictionary learning in face recognition*, in 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, June 2010, pp. 2691–2698.

[115] Z. Zhang, Y. Xu, J. Yang, X. Li, and D. Zhang, *A Survey of Sparse Representation: Algorithms and Applications*, IEEE Access, 3 (2015), pp. 490–530.

[116] Y. Zhao and J. Yang, *Hyperspectral image denoising via sparse representation and low-rank constraint*, IEEE Transactions on Geoscience and Remote Sensing, 53 (2015), pp. 296–308.

[117] S. Zu, H. Zhou, R. Wu, M. Jiang, and Y. Chen, *Dictionary learning based on dip patch selection training for random noise attenuation*, GEOPHYSICS, 84 (2019), pp. V169–V183.

Part II

Scientific results

Paper B

4D Seismic History Matching: Assessing the use of a Dictionary Learning Based Sparse Representation Method

Ricardo V. Soares, Xiaodong Luo, Geir Evensen, and Tuhin Bhakta

Journal of Petroleum Science and Engineering, 195, 2020

DOI: 10.1016/j.petrol.2020.107763

4D Seismic History Matching: Assessing the use of a Dictionary Learning Based Sparse Representation Method

Ricardo Vasconcellos Soares[a,b], Xiaodong Luo[a], Geir Evensen[a,c], and Tuhin Bhakta[a]

[a] - NORCE Norwegian Research Centre AS, Norway
[b] - University of Bergen, Norway
[c] - Nansen Environmental and Remote Sensing Center (NERSC), Norway

Received 25 February 2020; Received in revised form 5 August 2020; Accepted 7 August 2020

Available online 14 August 2020

0920-4105/ ©2020. The authors. Published by Elsevier B. V. This is an open access article under the CC BY license (https://creativecommons.org/licenses/by/4.0/)

Abstract

It is possible to improve oil-reservoir simulation models by conditioning them on 4D seismic data. Computational issues may arise related to both storage and CPU time due to the size of the 4D seismic dataset. An approach to reducing the computational requirements is to use a sparse representation method, e.g., Dictionary Learning, to select only the main features of the 4D seismic data. However, the introduction of a sparse representation method incurs an additional computational cost. Alternatively, if one uses ensemble-based methods, it is possible to reduce storage and CPU time by projecting the full seismic dataset on a smaller subspace. This paper evaluates the potential of sparsely representing the seismic data. We compare two experiments, one where we condition on the full dataset projected on a smaller subspace, and one where we use Dictionary Learning to represent the data sparsely. We use Dictionary Learning both on the complete 4D seismic dataset and also on a denoised version of the data. We perform the data assimilation in a slightly different formulation of the Iterative Ensemble Smoother Regularized Levenburg-Marquardt together with correlation-based adaptive localization. We apply these methods to the Brugge benchmark case. Experiment results show that sparse representation methods lead to a final ensemble that is closer to the reference solution, and denoising the seismic data before applying the sparse representation allows us to capture the 4D effect better. Thus, using a sparse representation method in 4D-seismic history matching leads to improved results compared to what we obtain when conditioning the models on the projected 4D seismic dataset.

Keywords: Ensemble data assimilation; Iterative ensemble smoother; 4D Seismic history matching; Big data assimilation; Dictionary learning

Introduction

Data assimilation, also known as history matching in reservoir engineering, denotes the process of conditioning reservoir models on dynamical observations from a reservoir, e.g., well rates and 4D seismic data, to update uncertain reservoir model parameters, e.g., porosity and permeability. Historically, most of the data assimilation processes have been mainly using production data from the wells because of its availability and easiness of gather. There is a vast list in the literature that successfully performed history

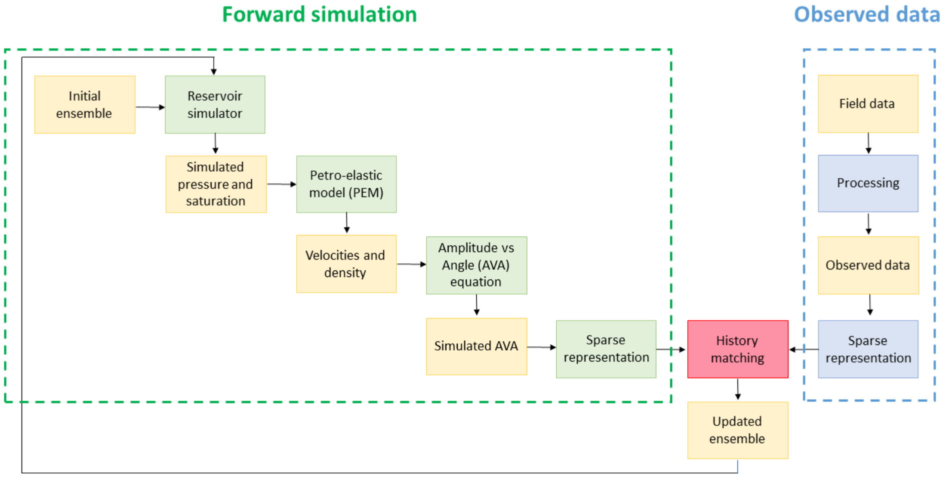


Figure 1: Flowchart of the use of 4D seismic data in history matching. The area within the green dashed rectangle represents the forward simulation of seismic data, while the area within the blue dashed rectangle represents the observed data from the field. The yellow boxes represent data, and the blue, green, and red boxes represent the required processing steps.

matching in various benchmark and real field cases. See, e.g., [19, 57, 12, 31, 52, 67], for some history-matching experiments.

Even though it might be possible to achieve a good representation of the reservoir using only well data, the inclusion of different data, such as seismic, can help to understand the flow dynamics in the oil reservoir better. Johnston [38] mentioned several important points for practical applications of 4D seismic, including rock physics models, seismic processing, data integration among others. Also, the author pointed out that seismic data are much denser in space than in time. Therefore, 4D seismic data will complement the information we have from rate data in the history-matching process. Several authors have proved the benefits of using 4D seismic data in history matching, both in benchmark and real cases [20, 61, 28, 56, 41, 36, 4].

There are several challenges when considering 4D seismic data in data assimilation. As commercial simulators usually provide only well rate and pressure and saturation maps, one needs to bring simulator response and seismic into the same domain. In other words, one has to transform acquired seismic data into pressure and saturation maps or bring them into another domain, such as acoustic impedance or amplitude-versus-angle (AVA). Besides, seismic data comprise huge datasets and, therefore, issues regarding computational storage and CPU time may arise. To deal with the big-data issue, authors such as Luo et al. [48], Liu and Grana [43] and Soares et al. [66] suggested different methodologies to represent the seismic data sparsely.

Fig. 1 depicts a flowchart concerning the use of 4D seismic data in history matching [49], where the authors consider AVA as the seismic data. The area covered by the green dashed border lines (referred to as the green part hereafter) represents the simulated data (forward simulation of seismic), while the area enclosed by the blue dashed border lines (named blue part hereafter) represents the observed data from the field. The yellow boxes represent data, and the blue, green, and red boxes represent the necessary steps to deal with this kind of data.

The green part describes the forward seismic simulation workflow that generates simulated seismic attributes (AVA in this case) from an ensemble of reservoir models. The boxes in the blue part convert seismic from raw waveforms to AVA data. This conversion procedure involves seismic processing, which is outside the scope of the current work. Thus, we assume that the observed AVA attributes in the blue part are readily available. Note that one can also assimilate other types of seismic attributes, such as Acoustic Impedance (AI) following a similar workflow [28, 61, 3], or other parameters that can be inferred from seismic, such as fluid fronts [41, 39].

As mentioned previously, the 4D seismic dataset is very big. Therefore, the flowchart considers a method capable of sparsely representing the seismic signal while retaining the main characteristics of the data. For this purpose, Luo et al. [49] and Lorentzen et al. [45] used the Discrete Wavelet Transform (DWT) to represent the 4D seismic dataset sparsely. Through DWT, they separated the data into low and high-frequency sub-bands. Then, they estimated the noise of the wavelet coefficients and applied a threshold value to retain only the leading wavelet coefficients. Although they reduced the number of retained coefficients substantially, it was necessary to define an empirical parameter to achieve a trade-off between the number of coefficients and the retained characteristics, which we will illustrate below (cf. Fig. 10).

Liu and Grana [43] proposed to learn a sparse representation of 4D seismic data through a deep convolutional autoencoder. The authors performed experiments in a 2D and 3D model and concluded that the method improved the data-assimilation performance. Nevertheless, one might need a more powerful tool, such as GPU, to obtain the results in a reasonable computational time. For the 3D model, for instance, they reduced the original dataset to 0.20% of its original size, and it took about 30 minutes to train the model with dimension $64 \times 64 \times 64$ in two different time surveys. Besides, Canchumuni et al. [9] reported that an extension of the deep learning method from 2D to 3D case studies might incur substantially more computational costs.

Soares et al. [66] investigated the use of a Dictionary Learning (DL) method for sparse representation of seismic data in a 3D case study with dimension $139 \times 48 \times 176$. The method retained the main characteristics of the 4D seismic by using a set of non-zero coefficients whose number was only about 0.25% of the original data size, and the related computational time was about 6.7 seconds.

The end products of the sparse representation procedure (applied to observed seismic data) will be the obtained non-zero coefficients. These non-zero coefficients will then be taken as the observations in 4D seismic history matching problems, as indicated in the blue part of Fig. 1. By doing so, a practical benefit is that the effective data size can be substantially smaller than the size of the original 4D seismic dataset, which will then help to mitigate the issues of computational memory and time during history matching.

Iterative ensemble-based methods are among state of the art for history matching, as they are relatively easy to implement, and have the capacity of dealing with large scale problems [23]. In ensemble-based methods, one can choose to project big 4D seismic data onto a subspace which is related to the ensemble of reservoir models, and whose dimension is no more than the ensemble size. These projected data then serve as the effective observations [45, 50, 13]. This approach is an alternative way to handle the issue of big data in history matching problems, yet without using any sparse representation method.

Therefore, there are two possible ways to deal with the big data issue: through sparse representations methods and through the projection of the observed data into a subspace related to the ensemble models. Note that in both approaches, one faces information loss in the dataset. However, the use of a sparse data representation introduces additional computational time. Hence, compared to the projection-based approach, is it still beneficial to use a sparse representation method for data-size reduction? To answer this question, we will examine the data assimilation performance in two sets of experiments: one employing the projection-based approach and the other adopting a Dictionary Learning based sparse presentation method. Furthermore, we also investigate an alternative way to apply the Dictionary Learning based sparse representation to obtain further improved assimilation performance. To perform history matching, we modify the Iterative Ensemble Smoother Regularized Levenburg-Marquadrdt (IES-RML) developed by Luo et al. [51] in a way that there is no need to perform the truncated singular value decomposition during data assimilation.

In the following, we start by explaining each component of the flowchart shown in Fig. 1. We divide this part into three sections. First, we explain the individual steps in the forward seismic simulation workflow. Then, we discuss the history matching part, where we focus on ensemble-based methods and we provide more details regarding the need of sparse representation methods. Afterwards, we introduce one of the Dictionary Learning methods, called the K-SVD algorithm, which is responsible for sparsely representing the data. Subsequently, we provide information about the case study (the Brugge benchmark case), methodology, and application. Finally, we show and discuss the results and draw conclusions.

As the different sections present unrelated topics, we prefer to keep the traditional notation for each topic and we introduce a list of symbols and variables at the beginning of each section.

Forward Simulation of 4D Seismic Data

Nomenclature	
μ	Shear modulus
ν	Poisson's ratio
ϕ	Porosity
ρ	Density
C_p	Average number of contacts per sphere
K	Bulk modulus
n	Degree of root
P	Pressure
S	Saturation
V	Wave velocity
Subcripts	
c	critical
d	rock grain porosity values lower than ϕ_c
eff	effective
f	saturation effect on the presence of oil and water

HM	Hertz-Mindlin model - dry rock at critical porosity
m	mineral
o	oil
P	Compressional wave
s	rock grain
S	Shear wave
sat	saturation effect
w	water

In this section, we provide information about the green boxes in Fig. 1, except the Sparse Representation, which will be given in the section "Observed Data - Sparse Representation: Dictionary Learning Through the K-SVD Algorithm".

Once one generates the initial ensemble containing the uncertain model variables, such as porosity, permeability, and fault multipliers, the next step is to run the simulation models to obtain simulated pressure and saturation profiles, which are among the important quantities to calculate the simulated seismic data.

Petro-elastic model (PEM)

Petro-elastic models convert pressure and saturation data into elastic attributes, such as acoustic impedance (or equivalently, wave velocities and densities), for seismic interpretation or inversion. PEM is dependent on the type of reservoir formation and fluids.

One of the most commonly used PEMs is the soft-sand model [53], in which the first step is to calculate the dry bulk modulus (K_{HM}) and the shear modulus (μ_{HM}) through the Hertz-Mindlin model [54] as in

$$K_{\mathrm{HM}} = \sqrt[n]{\frac{C_p^2(1-\phi_c)^2\mu_s^2}{18\pi^2(1-\nu_s)^2}P_{\mathrm{eff}}}, \tag{1}$$

and

$$\mu_{\mathrm{HM}} = \frac{5-4\nu_s}{5(2-\nu_s)}\sqrt[n]{\frac{3C_p^2(1-\phi_c)^2\mu_s^2}{2\pi^2(1-\nu_s)^2}P_{\mathrm{eff}}}. \tag{2}$$

In Eqs. (1) and (2), n is the degree of root, C_p is the average number of contacts per sphere, ϕ_c is the critical porosity, μ_s is the grain shear modulus, ν_s is the Poisson's ratio, and P_{eff} is the effective stress, i.e., the lithostatic pressure minus pore pressure. In this work, n is set to 3, C_p is set to 9 and ϕ_c to 36% (the maximum porosity value). Physically, one can interpret K_{HM} and μ_{HM} as the dry rock resistance to normal and shear stress, respectively. In addition, one can calculate Poisson's ratio as in

$$\nu_s = \frac{3K_s - \mu_s}{6K_s - \mu_s}. \tag{3}$$

K_s and μ_s are bulk modulus and the shear modulus of the rock grain, respectively.

The modified Lower Hashin-Shtrikman (MLHS) [35, 53] calculates the effective dry bulk modulus (K_d) and the effective shear modulus (μ_d) for porosity values (ϕ) lower than ϕ_c

$$K_d = \left(\frac{\frac{\phi}{\phi_c}}{K_{\mathrm{HM}} + \frac{4}{3}\mu_{\mathrm{HM}}} + \frac{\frac{1-\phi}{\phi_c}}{K_s + \frac{4}{3}\mu_{\mathrm{HM}}} \right)^{-1} - \frac{4}{3}\mu_{\mathrm{HM}}, \tag{4}$$

and

$$\mu_d = \left(\frac{\frac{\phi}{\phi_c}}{\mu_{\mathrm{HM}} + \frac{\mu_{\mathrm{HM}}}{6}Z} + \frac{\frac{1-\phi}{\phi_c}}{\mu_s + \frac{\mu_{\mathrm{HM}}}{6}Z} \right)^{-1} - \frac{\mu_{\mathrm{HM}}}{6}Z, \tag{5}$$

with

$$Z = \frac{9K_{\mathrm{HM}} + 8\mu_{\mathrm{HM}}}{K_{\mathrm{HM}} + 2\mu_{\mathrm{HM}}}. \tag{6}$$

The next step is known as the Gassmann model [30], which generates the saturated bulk modulus and shear modulus (K_{sat} and μ_{sat}, respectively) by including the saturation effect (Eqs. (7) and (8)).

$$K_{sat} = K_d + \frac{\left(1 - \frac{K_d}{K_s}\right)^2}{\frac{\phi}{K_f} + \frac{1-\phi}{K_s} - \frac{K_d}{K_s^2}}, \tag{7}$$

$$\mu_{sat} = \mu_d, \tag{8}$$

and K_f takes into consideration the presence of both water and oil

$$K_f = \left(\frac{S_w}{K_w} + \frac{S_o}{K_o} \right)^{-1}. \tag{9}$$

Here K_o and K_w are the bulk modulus of the oil and water, respectively, while S_o and S_w are the oil and water saturation. Subsequently, one calculates the saturated density (ρ_{sat}), P-wave and S-wave velocities (V_P and V_S), as in Mavko et al. [53]

$$\rho_{sat} = (1 - \phi)\rho_m + \phi S_w \rho_w + \phi S_o \rho_o, \tag{10}$$

$$V_p = \sqrt{\frac{K_{sat} + \frac{4}{3}\mu_{sat}}{\rho_{sat}}}, \tag{11}$$

and

$$V_s = \sqrt{\frac{\mu_{sat}}{\rho_{sat}}}, \tag{12}$$

where ρ_m, ρ_o, ρ_w and, ρ_{sat} are mineral density, oil density, water density, and saturated rock density, respectively. Note that one can obtain wave impedance by multiplying ρ_{sat} by V_p or V_s.

AVA equation

To calculate the AVA, one first needs to define the reflection coefficients at each interface between two adjacent layers by using the Zoeppritz equation [5]. Then, one should compute the travel time using V_p and the thickness of the gridblocks. Finally, the AVA data can be obtained by convolving the reflectivity series with a Ricker wavelet.

Here, a Ricker wavelet of 45 Hz is considered. Note that different incident angles can be considered while generating the AVA data. The readers are referred to Luo et al. [48] for more information about how to generate the AVA data.

In seismic history-matching problems, acoustic impedance can also be adopted, see, for example, Lorentzen et al. [45], Emerick [20], and Gosselin et al. [32]. However, AVA has the advantage of avoiding the inclusion of biases or errors during the seismic inversion process [49] and AVA can help to identify different situations in the reservoir.

History matching: Ensemble-based Methods

Nomenclature	
α	Regularization/Weight term
β	Scalar parameter responsible to increase/decrease α
δ	Measurement error
ε	Noise in the correlation calculated in the localization
ρ	Correlation between the model variables in the ensemble and the simulated data
σ	Standard deviation
$\mathbf{\Sigma}$	Singular values matrix
$\hat{\mathbf{\Sigma}}$	Truncated singular values matrix
θ	Threshold value in the automatic and adaptive correlation-based localization
ζ	Average data mismatch
$\mathbf{A}$	Ensemble anomalies
c	Entry of the localization matrix
$\mathbf{C}$	Localization matrix
$\mathbf{C}_{\mathrm{d}}$	Measurement error covariance matrix
$\mathbf{C}_{\mathrm{x}}$	Uncertain model parameter error covariance matrix
$\mathbf{d}$	Observed data
$\mathbf{d}^{\mathrm{sim}}$	Simulated data
$\mathbf{D}$	Ensemble matrix of observation residual
$\widetilde{\mathbf{D}}$	Rotated $\mathbf{D}$
$\mathbf{g}$	Reservoir simulator
$\mathbf{K}$	Kalman gain
N_{d}	Total number of observed data
N_{e}	Total number of realizations in the ensemble
N_{i}	Total number of iterations
N_{sv}	Total number of retained singular values
N_{x}	Total number of uncertain variables
$p(\cdot)$	Prior distribution

$p(\cdot\,\vert\cdot\,)$	Likelihood or posterior probability density function given a parameter
$\mathbf{S}$	Simulated data anomalies
$\widetilde{\mathbf{S}}$	Rotated $\mathbf{S}$
$\mathbf{U}$	Left singular vectors matrix
$\hat{\mathbf{U}}$	Truncated left singular vectors matrix
$\mathbf{V}$	Right singular vectors matrix
$\hat{\mathbf{V}}$	Truncated right singular vectors matrix
$\mathbf{x}$	Uncertain model parameter
$\mathbf{X}$	Ensemble matrix of uncertain model variables
z	Dummy variable defined in the correlation-based adaptive localization
Subscripts	
c	Common point
eff	Effective
j	Realization index
Superscripts	
0	Prior
i	Iteration index
true	True
T	Transpose

We will now introduce the ensemble method used for conditioning the model on the observations (red box in Fig. 1). Model conditioning or history matching is an essential component in the development and management of petroleum reservoirs. By using the information about the dynamical properties of the reservoir contained in observed data, such as well rates (oil, water, and gas) and seismic, one can constrain reservoir models and update the uncertain model parameters, such as permeability and porosity fields. The Ensemble Kalman Filter (EnKF) by Evensen [22], initially developed for sequential data assimilation, provides an efficient means for model conditioning. Nævdal et al. [55] introduced the EnKF for use in the petroleum industry, and since then, ensemble-based methods have attracted a lot of attention within the industry. Notably, the introduction of the Ensemble Smoother [69] (which is an EnKF that uses all data simultaneously to update all parameters in one step) for petroleum applications by Skjervheim et al. [65] triggered further development of effective iterative forms of the Ensemble Smoother.

History-matching problem

To formulate the history-matching problem, we denote the uncertain model parameters as $\mathbf{x}$, the (potentially imperfect) forward operator which transforms the model variables into the simulated data $\mathbf{d}^{\mathrm{sim}}$ as $\mathbf{g}$, such that

$$\mathbf{d}^{\mathrm{sim}} = \mathbf{g}(\mathbf{x}). \tag{13}$$

Besides, we have one set of observed data $\mathbf{d}$, which are generated through the following observation system

$$\mathbf{d} = \mathbf{g}^{\text{true}}(\mathbf{x}^{\text{true}}) + \boldsymbol{\delta}. \tag{14}$$

Here, $\mathbf{x}^{\text{true}}$ stands for the ground-truth model, and $\boldsymbol{\delta}$ represents measurement errors that are normally distributed random errors with zero mean and variance represented by the measurement error-covariance matrix $\mathbf{C}_{\text{d}}$, e.g., $\boldsymbol{\delta} \sim \mathcal{N}(\mathbf{0}, \mathbf{C}_{\text{d}})$. The true (but unknown) forward simulator $\mathbf{g}^{\text{true}}$ may be different from the actual forward operator $\mathbf{g}$ in use, hence model errors can arise. In this work, however, we do not consider model errors as in Luo and Bhakta [47], Chen and Oliver [13], and Emerick [20], for instance. More information on how to deal with model errors can be found in Evensen [26] and Luo [46].

Since $\mathbf{x}$ is uncertain, history matching is an inverse problem where one solves for a $\mathbf{x}$ that results in a model prediction $\mathbf{g}(\mathbf{x})$ that is close to the observed data $\mathbf{d}$. Furthermore, the number of parameters is normally much larger than the independent degrees of freedom in the observations. In this case, estimating model variables $\mathbf{x}$ based on the observations $\mathbf{d}$ and the forward simulator $\mathbf{g}$ defines a high-dimensional and under-determined inverse problem.

Through Bayes' theorem, it is possible to define the posterior probability density function of $\mathbf{x}$ given the observed data

$$p(\mathbf{x}|\mathbf{d}) \propto p(\mathbf{d}|\mathbf{x})p(\mathbf{x}), \tag{15}$$

where $p(\mathbf{x})$ is the prior distribution of the model parameters and $p(\mathbf{d}|\mathbf{x})$ is the likelihood. By assuming that both the prior and likelihood are Gaussian distributed and considering an ensemble of multiple realizations [23], it is possible to sample the posterior distribution $p(\mathbf{x}|\mathbf{d})$ by the minimization of an ensemble of cost functions associated for each realization j

$$\text{argmin}_{\{\mathbf{x}_j\}}\left\{\left(\mathbf{g}(\mathbf{x}_j) - \mathbf{d}_j\right)^{\text{T}}\mathbf{C}_{\text{d}}^{-1}\left(\mathbf{g}(\mathbf{x}_j) - \mathbf{d}_j\right) + \left(\mathbf{x}_j - \mathbf{x}_j^0\right)^{\text{T}}\mathbf{C}_{\text{x}}^{-1}\left(\mathbf{x}_j - \mathbf{x}_j^0\right)\right\}, \tag{16}$$

Here, $\mathbf{x}^0$ is the prior estimate of the model parameters and $\mathbf{C}_{\text{x}}$ is the error covariance matrix of the uncertain model parameters (or the prior covariance matrix for the model parameters). Note that one needs to sample a prior ensemble of models $\mathbf{x}_j^0$ from $\mathcal{N}(\mathbf{x}^0, \mathbf{C}_{\text{x}})$ and an ensemble of measurements $\mathbf{d}_j$ from $\mathcal{N}(\mathbf{d}, \mathbf{C}_{\text{d}})$.

For the case where $\mathbf{g}$ is linear, the minimization of the Eq. (16) will sample the correct posterior distribution for an infinite ensemble size, N_{e}, [23]. However, if $\mathbf{g}$ is non-linear, as in the case of reservoir applications, the minimization of the Eq. (16) is an approximation.

By minimizing Eq. (16), one can obtain the following formula to update the uncertain model variables

$$\mathbf{x}_j = \mathbf{x}_j^0 + \mathbf{A}\mathbf{S}^{\text{T}}\left(\mathbf{S}\mathbf{S}^{\text{T}} + \mathbf{C}_{\text{d}}\right)^{-1}\left(\mathbf{d}_j - \mathbf{g}(\mathbf{x}_j^0)\right), \tag{17}$$

where

$$\mathbf{A} = \frac{1}{\sqrt{N_{\text{e}} - 1}}\left(\mathbf{x}_1^0 - \mathbf{x}_c^0, \ldots, \mathbf{x}_{N_{\text{e}}}^0 - \mathbf{x}_c^0\right);\ \mathbf{x}_c^0 = \frac{1}{N_{\text{e}}}\sum_{j=1}^{N_{\text{e}}}\mathbf{x}_j^0, \tag{18}$$

$$\mathbf{S} = \frac{1}{\sqrt{N_{\text{e}} - 1}}\left(\mathbf{g}(\mathbf{x}_1^0) - \bar{\mathbf{d}}, \ldots, \mathbf{g}(\mathbf{x}_{N_{\text{e}}}^0) - \bar{\mathbf{d}}\right);\ \bar{\mathbf{d}} = \frac{1}{N_{\text{e}}}\sum_{j=1}^{N_{\text{e}}}\mathbf{g}(\mathbf{x}_j^0), \tag{19}$$

and N_{e} is the total number of realizations in the ensemble. Note that $\mathbf{C}_{\mathrm{x}} = \mathbf{A}(\mathbf{A})^{\mathrm{T}}$. Eq. (17) is known as the analysis equation for the Ensemble Smoother [69]. For more information about the minimization process and ensemble-based methods in general, the readers are referred to Evensen [23] and Aanonsen et al. [1].

To reduce the impact of the approximation, one can use iterative forms of the ensemble smoother as they generate better approximate solutions than the non-iterative form [25]. Among recent iterative ensemble-smoothers, the most popular are the EnRML method by Chen and Oliver [10, 11] and the Ensemble Smoother with Multiple Data Assimilation (ESMDA) by Emerick and Reynolds [21]. A recent reformulation of the EnRML by Evensen et al. [27] and Raanes et al. [59] is a more effiecient alternative than the original EnRML method. In this work, we employ the Iterative Ensemble Smoother Regularized Levenburg-Marquardt (IES-RML) by Luo et al. [51]. Even though the different iterative smoothers are similar to each other, they solve slightly different problems, and it is difficult to choose the best one since the results may be case dependent [25]. In the next section, we provide more information about the IES-RLM, which we use in this work.

IES-RLM

Luo et al. [51] were more interested in the average change in data mismatch between iterations. Hence, they proposed the following sequence of minimization problems for each realization

$$\begin{aligned}\operatorname{argmin}_{\{\mathbf{x}_j^{i+1}\}} \frac{1}{N_{\mathrm{e}}} \sum_{j=1}^{N_{\mathrm{e}}} \Bigg\{ & \Big(\mathbf{d}_j - \mathbf{g}(\mathbf{x}_j^{i+1})\Big)^{\mathrm{T}} \mathbf{C}_{\mathrm{d}}^{-1} \Big(\mathbf{d}_j - \mathbf{g}(\mathbf{x}_j^{i+1})\Big) + \\ & \alpha^i \Big(\mathbf{x}_j^{i+1} - \mathbf{x}_j^i\Big)^{\mathrm{T}} \mathbf{C}_{\mathrm{x}}^{-1} \Big(\mathbf{x}_j^{i+1} - \mathbf{x}_j^i\Big) \Bigg\},\end{aligned} \tag{20}$$

where i is the iteration number ($i = 0, 1, ..., N_{\mathrm{i}}$). The first term accounts for the difference between the simulated and observed data as in Eq. (16), while the second term penalizes the update increments in each iteration regarding the prior ensemble. In this formulation, α^i is a positive scalar responsible for assigning a weight to the second term.

The minimization of the Eq. (20) gives us the following formula to update the uncertain model variables for the IES-RML method

$$\mathbf{x}_j^{i+1} = \mathbf{x}_j^i + \mathbf{A}^i (\mathbf{S}^i)^{\mathrm{T}} \Big(\mathbf{S}^i (\mathbf{S}^i)^{\mathrm{T}} + \alpha^i \mathbf{C}_{\mathrm{d}}\Big)^{-1} \Big(\mathbf{d}_j - \mathbf{g}(\mathbf{x}_j^i)\Big). \tag{21}$$

Here, $\mathbf{A}^i$ is similar as in Eq. (18), with the exception that now one calculates them at each iteration, α^i is the regularization coefficient which changes over the iteration steps, to be explained later, and $\mathbf{S}^i$ is defined as

$$\mathbf{S}^i = \frac{1}{\sqrt{N_{\mathrm{e}} - 1}} \Big(\mathbf{g}(\mathbf{x}_1^i) - \mathbf{g}(\mathbf{x}_c^i), ..., \mathbf{g}(\mathbf{x}_{N_{\mathrm{e}}}^i) - \mathbf{g}(\mathbf{x}_c^i)\Big). \tag{22}$$

In this algorithm, one needs to choose certain stopping criteria for the number of iterations. According to Luo et al. [51], such criteria should be based on the following:

1. If the data mismatch becomes smaller than the number of observed data points (N_{d}) times a factor (1 in this study).

2. A maximum number of iterations (20 in this study).
3. The relative change between data mismatch in two consecutive iterations is small (less than 0.01% in this study).

The average data mismatch used in criterion 1 and 3 is defined as

$$\zeta = \frac{1}{N_\mathrm{e}} \sum_{j=1}^{N_\mathrm{e}} \Big(\mathbf{d}_j - \mathbf{g}(\mathbf{x}_j)\Big)^\mathrm{T} \mathbf{C}_\mathrm{d}^{-1} \Big(\mathbf{d}_j - \mathbf{g}(\mathbf{x}_j)\Big). \tag{23}$$

We now define the ensemble matrix

$$\mathbf{X}^i = \left(\mathbf{x}_1^i, \mathbf{x}_2^i, \ldots, \mathbf{x}_{N_\mathrm{e}}^i\right), \tag{24}$$

and additionally one can define the ensemble matrix of observation residuals as the observed data minus the simulated data as in

$$\mathbf{D}^i = \Big(\mathbf{d}_1 - \mathbf{g}(\mathbf{x}_1^i), \ldots, \mathbf{d}_{N_\mathrm{e}} - \mathbf{g}(\mathbf{x}_{N_\mathrm{e}}^i)\Big). \tag{25}$$

Besides, assuming that we have access to a symmetric square root $\mathbf{C}_\mathrm{d}^{\frac{1}{2}}$ of $\mathbf{C}_\mathrm{d}$ such that $\mathbf{C}_\mathrm{d} = \mathbf{C}_\mathrm{d}^{\frac{1}{2}}(\mathbf{C}_\mathrm{d}^{\frac{1}{2}})^\mathrm{T}$, and considering the following rotated operators

$$\widetilde{\mathbf{S}}^i = \mathbf{C}_\mathrm{d}^{-\frac{1}{2}} \mathbf{S}^i, \tag{26}$$

$$\widetilde{\mathbf{D}}^i = \mathbf{C}_\mathrm{d}^{-\frac{1}{2}} \mathbf{D}^i, \tag{27}$$

one can write the analysis equation as

$$\mathbf{X}^{i+1} = \mathbf{X}^i + \mathbf{A}^i (\widetilde{\mathbf{S}}^i)^\mathrm{T} \Big(\widetilde{\mathbf{S}}^i (\widetilde{\mathbf{S}}^i)^\mathrm{T} + \alpha^i \mathbf{I}\Big)^{-1} \widetilde{\mathbf{D}}^i, \tag{28}$$

Note that in many practical cases, $\mathbf{C}_\mathrm{d}$ is assumed to be diagonal (uncorrelated measurement errors). Hence, Eq. (27) and (26) can be viewed as a re-scaling.

Nevertheless, the matrix product to be inverted, $\Big(\widetilde{\mathbf{S}}^i (\widetilde{\mathbf{S}}^i)^\mathrm{T}\Big)$, has dimension $N_\mathrm{d} \times N_\mathrm{d}$, which can be very big (especially when using 4D seismic data). Therefore, a common procedure in the literature to obtain a more numerically stable algorithm is to apply the Truncated Singular Value Decomposition (TSVD) to the matrix $\widetilde{\mathbf{S}}^i$. In this regard, Evensen [23] suggested to keep the leading singular values that add up to between 90% and 99.9% of the total sum of squared singular values. Suppose that, through the TSVD, one obtains

$$\widetilde{\mathbf{S}}^i \approx \hat{\mathbf{U}}^i \hat{\mathbf{\Sigma}}^i (\hat{\mathbf{V}}^i)^\mathrm{T}, \tag{29}$$

where the matrices with respect to singular vectors or values, $\hat{\mathbf{U}}^i$, $\hat{\mathbf{\Sigma}}^i$, and $(\hat{\mathbf{V}}^i)^T$, have the dimensions of $N_\mathrm{d} \times N_\mathrm{sv}$, $N_\mathrm{sv} \times N_\mathrm{sv}$, and $N_\mathrm{sv} \times N_\mathrm{e}$, respectively. N_sv is the number of kept leading singular values.

Inserting Eq. (29) into Eq. (28) and applying some algebra, one has

$$\mathbf{X}^{i+1} = \mathbf{X}^i + \mathbf{A}^i \hat{\mathbf{V}}^i (\hat{\mathbf{\Sigma}}^i)^\mathrm{T} \Big(\hat{\mathbf{\Sigma}}^i (\hat{\mathbf{\Sigma}}^i)^\mathrm{T} + \alpha^i \mathbf{I}\Big)^{-1} (\hat{\mathbf{U}}^i)^\mathrm{T} \widetilde{\mathbf{D}}^i. \tag{30}$$

Several authors, such as Luo et al. [51], Luo and Bhakta [47], Lorentzen et al. [45] and Chen and Oliver [13], used this formulation. Nevertheless, if we consider the Woodbury identity (see, e.g., [27])

$$\mathbf{B}^{\mathrm{T}}(\mathbf{B}\mathbf{B}^{\mathrm{T}}+\mathbf{R})^{-1}=(\mathbf{I}+\mathbf{B}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{B})^{-1}\mathbf{B}^{\mathrm{T}}\mathbf{R}^{-1}, \tag{31}$$

it is possible to rearrange Eq. (28) into the following form

$$\mathbf{X}^{i+1}=\mathbf{X}^{i}+\mathbf{A}^{i}\left(\left(\widetilde{\mathbf{S}}^{i}\right)^{\mathrm{T}}\widetilde{\mathbf{S}}^{i}+\alpha^{i}\mathbf{I}\right)^{-1}\left(\widetilde{\mathbf{S}}^{i}\right)^{\mathrm{T}}\widetilde{\mathbf{D}}^{i}. \tag{32}$$

The nice property of the Eq. (32) is that the inversion is of a matrix of dimension the ensemble size $N_{\mathrm{e}} \times N_{\mathrm{e}}$ rather than the measurement dimension $(N_{\mathrm{d}} \times N_{\mathrm{d}})$ in Eq. (28). Therefore, one can avoid the application of the TSVD.

The final matrix product in Eq. (32) $\left(\left(\widetilde{\mathbf{S}}^{i}\right)^{\mathrm{T}}\widetilde{\mathbf{D}}^{i}\right)$ is effectively a projection of the observation residuals onto the ensemble space spanned by $\widetilde{\mathbf{S}}^{i}$. Thus, by solving the update equation with Eq. (32), one can only exploit information from the measurements that can be represented by the ensemble. In addition, the matrix multiplication $\left(\left(\widetilde{\mathbf{S}}^{i}\right)^{\mathrm{T}}\widetilde{\mathbf{D}}^{i}\right)$ helps to handle the big data assimilation problem, as it produces a lower-dimensional representation of the observation residuals.

Given Eq. (32), the regularization parameter α^{i} is determined following the rule below [51, 50]

$$\alpha^{i}=\beta^{i}\times\operatorname{trace}\left(\left(\widetilde{\mathbf{S}}^{i}\right)^{\mathrm{T}}\widetilde{\mathbf{S}}^{i}\right)/N_{\mathrm{e}}, \tag{33}$$

where β^{i} is a scalar that should decrease by a certain rate if the average data mismatch is reduced while increasing by a certain rate instead if the average data mismatch increases, and trace is an operator that calculates the trace of a matrix. Note that depending on the formulation, the calculation of α^{i} will change, as it depends on the matrix that is being summed $\left(\left(\widetilde{\mathbf{S}}^{i}\right)^{\mathrm{T}}\widetilde{\mathbf{S}}^{i}\right)$, in this case.

Nevertheless, the limited number of realizations in the ensemble and a possible big dataset in the observed data can make the problem over-determined, which would cause a strong reduction of the ensemble variability. An approach much used to deal with this issue is the application of a localization technique.

Localization

The traditional view is that localization is required to mitigate the effect of spurious correlation from remote observations, which leads to variance reduction in the ensemble [1, 19, 13, 63, 16, 47, 67]. Localization is described by the matrix $\mathbf{C}$, which represents the relationship between each observed data and the uncertain model variables. Hence, one can use the localization matrix $\mathbf{C}$ to specify which data should be used to update which set of uncertain parameters.

To conduct localization, it is better to rearrange Eq. (32) as in

$$\mathbf{X}^{i+1}=\mathbf{X}^{i}+\mathbf{K}^{i}\widetilde{\mathbf{D}}^{i}, \tag{34}$$

where we have introduced the so-called Kalman gain matrix $\mathbf{K}^{i} \in \Re^{N_{\mathrm{x}}\times N_{\mathrm{d}}}$ that can take different forms using either one of the Eqs. (28) or (32), e.g.,

$$\mathbf{K}^{i}=\mathbf{A}^{i}\left(\left(\widetilde{\mathbf{S}}^{i}\right)^{\mathrm{T}}\widetilde{\mathbf{S}}^{i}+\alpha^{i}\mathbf{I}\right)^{-1}\left(\widetilde{\mathbf{S}}^{i}\right)^{\mathrm{T}}. \tag{35}$$

Computationally, it is better not to compute the Kalman gain matrix, since it has a big dimension $N_\mathrm{x} \times N_\mathrm{d}$. However, for using localization based on tapering of the Kalman gain, the Kalman gain matrix needs to be computed.

Localization is implemented as a tapering of the values contained in the Kalman Gain matrix. For instance, one can define a tapering function that equals one at the measurement location and zero far from the measurement location where there should be no significant impact of the measurement. A common way of implementing localization is to compute the update from the following equation

$$\mathbf{X}^{i+1} = \mathbf{X}^{i} + \mathbf{C} \circ \mathbf{K}^{i}\widetilde{\mathbf{D}}^{i}. \tag{36}$$

Here the Schur or Hademard product $\mathbf{C} \circ \mathbf{K}^{i}$ defines an element-wise multiplication and results in a new matrix of the same dimension as the original Kalman Gain matrix $\mathbf{K}^{i} \in \Re^{N_\mathrm{x} \times N_\mathrm{d}}$. Therefore, it is possible to use different linear combinations to update the uncertain model variables [13].

One alternative to deal with the big dimension of the Kalman gain is to use Eq. (36) and sparsely represent the observed data, so it is possible to achieve a much lower number of observed data (N_d) and, then, one can deal with the Kalman gain matrix.

Another alternative is to define the effective Kalman gain matrix ($\mathbf{K}^{i}_\mathrm{eff}$) [45, 47, 13]. For that purpose, if one considers Eq. (32) and compute the exact Singular Value Decomposition of $\widetilde{\mathbf{S}}^{i}$ as in

$$\widetilde{\mathbf{S}}^{i} = \mathbf{U}^{i}\mathbf{\Sigma}^{i}(\mathbf{V}^{i})^\mathrm{T}, \tag{37}$$

one will obtain the following equation

$$\mathbf{X}^{i+1} = \mathbf{X}^{i} + \mathbf{A}^{i}\mathbf{V}^{i}\Big((\mathbf{\Sigma}^{i})^\mathrm{T}\mathbf{\Sigma}^{i} + \alpha^{i}\mathbf{I}\Big)^{-1}(\mathbf{\Sigma}^{i})^\mathrm{T}(\mathbf{U}^{i})^\mathrm{T}\widetilde{\mathbf{D}}^{i}. \tag{38}$$

Therefore, one can define $\mathbf{K}^{i}_\mathrm{eff}$ as

$$\mathbf{K}^{i}_\mathrm{eff} = \mathbf{A}^{i}\mathbf{V}^{i}\Big((\mathbf{\Sigma}^{i})^\mathrm{T}\mathbf{\Sigma} + \alpha^{i}\mathbf{I}\Big)^{-1} \tag{39}$$

and apply localization

$$\mathbf{X}^{i+1} = \mathbf{X}^{i} + \mathbf{C}^{i}_\mathrm{eff} \circ \mathbf{K}^{i}_\mathrm{eff}(\mathbf{U}^{i}\mathbf{\Sigma}^{i})^\mathrm{T}\widetilde{\mathbf{D}}^{i}, \tag{40}$$

where $\mathbf{C}_\mathrm{eff}$ has dimension of $N_\mathrm{x} \times N_\mathrm{e}$ (same as the $\mathbf{K}_\mathrm{eff}$). The advantage of using this formulation is that by first multiplying the last three matrices $\Big((\mathbf{U}^{i}\mathbf{\Sigma}^{i})^\mathrm{T}\widetilde{\mathbf{D}}^{i}\Big)$, the effective Kalman gain matrix $\Big(\mathbf{K}^{i}_\mathrm{eff}\Big)$ will have dimension of $N_\mathrm{x} \times N_\mathrm{e}$, which is numerically more efficient to store and manipulate as $N_\mathrm{e} \ll N_\mathrm{d}$. Furthermore, as mentioned previously, this multiplication can be viewed as a projection of the observation residuals into a smaller subspace. However, there is some information loss, as we will show later (cf. Fig. 9).

There are different ways of computing $\mathbf{C}$. One of the most common approaches appears to be distance-dependent localization [33, 20, 63], in which all data points and model variables are assumed to be associated with certain physical locations. Besides, one needs to define a critical length to specify the regions in which each data should influence the model variables.

Nevertheless, Luo and Bhakta [47] pointed out that the definition of the critical length might be difficult and case-dependent. Moreover, some types of observations used in data assimilation, like seismic data represented in a different transform domain, may not have any associated physical locations, in contrast to the more conventional production data from wells. To overcome these (and a few other) noticed issues, Luo and Bhakta [47] proposed an automatic and adaptive correlation-based localization scheme.

Automatic and adaptive correlation-based localization

Luo and Bhakta [47] calculated the correlation (ρ) between uncertain model variables in the initial ensemble ($\mathbf{X}^0$) and the initial simulated data $\left(\mathbf{g}(\mathbf{X}^0)\right)$. Then, they defined threshold values to determine which observed data should be used to update which uncertain model variables. Since in practical implementations of ensemble-based methods, the ensemble size is limited (usually around 100), the estimated correlation from a limited sample between two uncorrelated variables might be different from zero. Therefore, the main idea in Luo and Bhakta [47] is to obtain the spurious correlations (or noise) of these uncorrelated data, and use them to calculate the threshold values which are needed for the computation of $\mathbf{C}$.

To estimate the noise level, one can calculate the correlation fields between an ensemble of reservoir models (independent and identically distributed) and an ensemble of simulated data that are produced independently of the previous ensemble. Due to the independence between them, one can take the resulting correlation fields as the desired noise fields ε, which can be approximated by some Gaussian distributions with zero mean and unknown standard deviations denoted by σ hereafter. To calculate σ, Luo and Bhakta [47] proposed to use the median absolute deviation estimator [17], as in

$$\sigma = \frac{\text{median}(\text{abs}(\varepsilon))}{0.6745}. \tag{41}$$

After that, one can compute the threshold (θ) by

$$\theta = \sigma\sqrt{2\ln(\#\rho)}\,, \tag{42}$$

where $\#\rho$ is the number of elements in ρ.

Note that one should perform this procedure for each group type of uncertain variables. In other words, one must calculate the noise, standard deviation, and threshold for each group, such as porosity, horizontal permeability, and vertical permeability, for instance.

Finally, to generate a smooth tapering function, Luo and Bhakta [47] defined a dummy variable

$$z = \frac{1 - \text{abs}(\rho)}{1 - \theta}, \tag{43}$$

where ρ is the sample correlation between a model variable and a data point, and θ is the corresponding threshold value obtained from Eq. (42). The variable z is then used in the Gaspari and Cohn formula [29]

$$c(z) = \begin{cases} -\frac{1}{4}z^5 + \frac{1}{2}z^3 + \frac{5}{8}z^3 - \frac{5}{3}z^2 + 1, & \text{if } 0 \leq z \leq 1 \\ -\frac{1}{12}z^5 - \frac{1}{2}z^4 + \frac{5}{8}z^3 + \frac{5}{3}z^2 - 5z + 4 - \frac{2}{3}z^{-1}, & \text{if } 1 < z \leq 2 \\ 0, & \text{if } z > 2. \end{cases} \tag{44}$$

Here, c are the entry values of the localization matrix $\mathbf{C}$.

Note that if one uses Eq. (36), one should calculate the correlation between the initial uncertain model variables $(\mathbf{X}^0)$ and the simulated data $\big(\mathbf{g}(\mathbf{X}^0)\big)$. However if one uses Eq. (40), the correlation should be computed between the inital uncertain model variables and the projected data $\Big((\mathbf{U}^i\mathbf{\Sigma}^i)^{\mathrm{T}}\big(\mathbf{g}(\mathbf{X}^0)\big)\Big)$. For that reason, it is necessary to compute $\mathbf{C}^i_{\mathrm{eff}}$ at each iteration, according to $(\mathbf{U}^i\mathbf{\Sigma}^i)^{\mathrm{T}}$.

It is important to highlight that when we use Eq. (36) and the sparsely observed data, we might still achieve a large N_{d} and, consequently, $\mathbf{K}^i$ might still be difficult to store and manipulate. Hence, one can compute each row or group of rows separately, or apply local analysis to deal better with the size of $\mathbf{K}^i$ [63, 13, 27, 8]. Nevertheless, we do not conduct local analysis in this work, but we will consider it in the future.

Observed Data - Sparse Representation: Dictionary Learning Through the K-SVD Algorithm

Nomenclature	
Δx	Step size in the x-direction
Δy	Step size in the y-direction
$\mathbf{\Gamma}$	Column vector of the Sparse matrix - sparse vector
$\mathbf{\Gamma}$	Sparse matrix
ϵ	Error tolerance
λ	Penalty for image reconstruction
σ	Noise standard deviation
c	Inner product coefficient between the atoms of the Dictionary and the residual
C	Constant in the calculation of the error tolerance
$\mathbf{d}$	Atom in the dictionary
$\mathbf{D}$	Dictionary
$\mathbf{E}$	Residual matrix
$\mathbf{g}$	Row vector of the Sparse matrix
$\mathbf{l}$	Data
L	Information loss
$\max$	Maximum value
n	Patch size
N_{it}	Total number of iterations
N_{k}	Total number of atoms in the Dictionary
N_{nz}	Total number of nonzero coefficients
N_{ts}	Total number of patches (training dataset size)
$\mathbf{r}$	Residual vector
$\mathbf{x}$	Patches in the original seismic dataset
$\mathbf{X}$	Seismic dataset
Subscripts	

F	Frobenius
i	Patch index
I	Indexes of the locations of nonzero coefficients in $\mathbf{g}^{\mathrm{T}}$
j	Dictionary index
J	Collection of indexes of the atoms in the dictionaries used to calculate the sparse vector
Superscripts	
$+$	Pseudo-inverse
m	Iteration index for the sparse vector calculation
rec	Reconstructed
true	True
T	Transpose

As mentioned in the previous section, one alternative to deal with the big data from 4D seismic in history matching is through the application of sparse representation methods. Therefore, this part focuses on the Dictionary Learning method through the K-SVD algorithm [2], which is responsible for sparsely represent the 4D seismic data to be used in history matching.

In the sparse representation problem, let us denote by $\mathbf{X}$ a matrix that is derived from the original seismic dataset (more information on how to form $\mathbf{X}$ is given in the next paragraphs). Besides, we assume that we have an initial guess of a matrix $\mathbf{D}$ (called dictionary) whose columns are filled with a set of predefined redundant basis elements (called atoms), and a separate matrix $\mathbf{\Gamma}$ associated with $\mathbf{D}$. Hence, our purpose here is to approximate $\mathbf{X}$ as

$$\mathbf{X} \approx \mathbf{D\Gamma}. \tag{45}$$

Given the dictionary $\mathbf{D}$, the matrix $\mathbf{X}$ is (approximately) represented by the non-zero elements of the matrix $\mathbf{\Gamma}$. Therefore, if $\mathbf{\Gamma}$ is a sparse matrix, then one can achieve the purpose of sparse representation.

For numerical efficiency in handling big datasets, Aharon et al. [2] suggested that a big dataset be first divided into a number of (possibly overlapping) subsets (called patches), so one can form $\mathbf{X}$. Note that in general, each patch can be a multidimensional (e.g., 2D or 3D) array extracted from the original seismic data (represented by the yellow box named "Observed data" on the right-hand side of Fig. 1). Here, we first consider 2D arrays each with dimension $n \times n$ where n denotes the patch size. We then reshape these 2D arrays into column vectors $\mathbf{x}_i \in \mathbb{R}^{n^2}$ for $i = 1, 2, \cdots, N_{\mathrm{ts}}$, such that

$$\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{N_{\mathrm{ts}}}] \in \mathbb{R}^{n^2 \times N_{\mathrm{ts}}}, \tag{46}$$

where N_{ts} is the number of patches (also called the training dataset size).

Accordingly, let $\mathbf{D} \in \mathbb{R}^{n^2 \times N_{\mathrm{k}}}$, where N_{k} denotes the number of atoms (columns) in the dictionary (called dictionary size), then one has the associated matrix $\mathbf{\Gamma}$ defined as

$$\mathbf{\Gamma} = [\gamma_1, \gamma_2, \ldots, \gamma_{N_{\mathrm{ts}}}] \in \mathbb{R}^{N_{\mathrm{k}} \times N_{\mathrm{ts}}}, \tag{47}$$

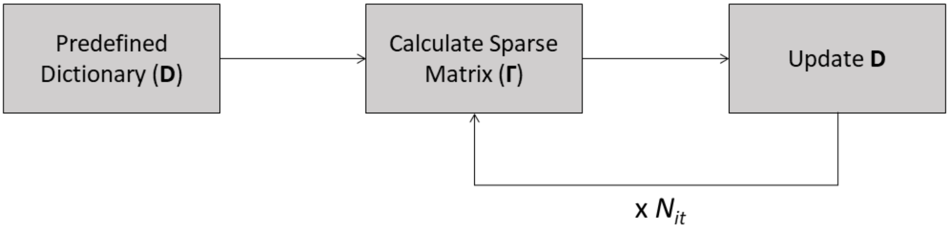

Predefined Dictionary (**D**)
Calculate Sparse Matrix (**Γ**)
Update **D**
x N_{it}

Figure 2: Dictionary Learning scheme.

where γ_i ($i = 1, 2, \cdots, N_{\text{ts}}$) are columns vectors that contain the representation coefficients associated with individual atoms. In other words, each $\mathbf{x}_i$ are approximated by a linear combination of the atoms in $\mathbf{D}$, in terms of

$$\mathbf{x}_i \approx \mathbf{D}\gamma_i. \tag{48}$$

In the current work, we adopt the K-SVD algorithm [2] to tackle the sparse representation problem. Aharon et al. [2] originally proposed K-SVD as a Dictionary Learning method for image denoising. Later on, this algorithm has also been applied to different areas, for instance, face recognition problems [71], magnetic resonance image reconstruction [60], and representation of geological facies in oil reservoirs [42].

K-SVD is an iterative algorithm and can be divided into two main parts: (1) calculation of the sparse representation matrix; and (2) dictionary update. As such, one needs to pre-choose a dictionary to initialize the algorithm. Elad and Aharon [18] and Soares et al. [66] used a Discrete Cosine Transform (DCT) function as the initial dictionary and reported it as a good choice. Note that the initial dictionary can also be in other forms, such as wavelet or curvelet basis functions, or some random parts of the original image. Through the learning process, the dictionary will be updated to better adapt to each specific problem on hand, and consequently, to achieve a better representation with fewer non-zero coefficients [68]. Fig. 2 depicts the scheme for the Dictionary Learning method, and as an iterative method, one should define the total number of iterations N_{it}.

Calculation of the sparse representation matrix

After the initial dictionary is chosen, one can move to calculate the sparse matrix $\mathbf{\Gamma}$. As suggested by Aharon et al. [2], it is necessary to divide the original dataset into smaller patches and transform them into column vectors ($\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{N_{\text{ts}}}$). Subsequently, one can calculate the sparse vector for each patch ($\gamma_1, \gamma_2, \ldots, \gamma_{N_{\text{ts}}}$).

One way of solving the sparse representation problem is to add an error constraint [62] as in

$$\gamma_i = \text{Argmin}_{\gamma_i} \|\gamma_i\|_0 \quad \text{subject to} \quad \|\mathbf{x}_i - \mathbf{D}\gamma_i\|_2^2 \leq \epsilon\,. \tag{49}$$

Here ϵ specifies the error tolerance, and is calculated as

$$\epsilon = \left(C\sigma\sqrt{n^2}\right)^2, \tag{50}$$

where C is a constant determined as 1.15 by Elad and Aharon[18], and σ is the noise standard deviation, which needs to be defined beforehand.

The approach proposed by Elad and Aharon [18] is to use the Orthogonal Matching Pursuit (OMP) to iteratively fill the sparse vector γ_i with one non-zero coefficient each time, until the constraint ($\|\mathbf{x}_i - \mathbf{D}\gamma_i\|_2^2 \leq \epsilon$) is satisfied.

In the OMP algorithm, the first step is to calculate the inner product coefficient between the columns (or atoms) $\mathbf{d}_j$ $(j = 1, 2, \cdots, N_\mathrm{k})$ of the dictionary $\mathbf{D}$ and the residual $\mathbf{r}_i^m$ $(i = 1, 2, \cdots, N_\mathrm{ts})$

$$c_i^m = (\mathbf{r}_i^m)^\mathrm{T}\mathbf{d}_j, \tag{51}$$

where m is the iteration index for the sparse vector calculation, and the residual $\mathbf{r}_i^m$ is defined as

$$\mathbf{r}_i^m = \mathbf{x}_i - \mathbf{D}\gamma_i^m. \tag{52}$$

Note that for the first iteration $\gamma_i^0 = 0$, then $\mathbf{r}_i^m = \mathbf{x}_i$.

Next, one selects the atom with the highest inner product coefficient with $\mathbf{r}_i^m$ and stores its index in J. In other words, one selects the atom with the largest correlation with the residual. Finally, it is necessary to orthogonalize the original data ($\mathbf{x}_i$) in the space spanned by the atoms in J (denote by $\mathbf{D}_J$), in terms of

$$\gamma_i^m = (\mathbf{D}_J)^+\mathbf{x}_i = (\mathbf{D}_J^\mathrm{T}\mathbf{D}_J)^{-1}\mathbf{D}_J^\mathrm{T}\mathbf{x}_i. \tag{53}$$

One should do these steps multiple times until reaching the stopping criterion. It is important to highlight that at every iteration step (m), one adds one index to J and one non-zero coefficient to γ_i, and these iterations are related only to the calculation of the sparse matrix.

Dictionary update

After calculating the sparse matrix $\mathbf{\Gamma}$, one needs to update the dictionary $\mathbf{D}$. In this regard, Elad and Aharon [18] proposed to do the update column by column. Hence, one should only consider the signals in $\mathbf{X}$ that uses the atom being updated. In addition, Elad and Aharon [18] also proposed to update only the non-zero coefficients in $\mathbf{\Gamma}$.

If one thinks of $\mathbf{\Gamma}$ as a matrix composed of the row vectors $\mathbf{g}_j^T$ and considers updating the j-th atom $\mathbf{d}_j$ of the Dictionary $\mathbf{D}$, one can calculate the residual

$$\mathbf{E}_j = \mathbf{X}_{I_j} - \sum_{a \neq j} \mathbf{d}_a \mathbf{g}_{a,I_j}^T, \tag{54}$$

when the $\mathbf{d}_j$ atom is excluded for sparse representation. The list I_j indexes the locations of non-zero coefficients in $\mathbf{g}_j^T$. For instance, if

$$\mathbf{g}_j^T = (0, \gamma_{j,2}, 0, \ldots, 0, \gamma_{j,10}, 0, \ldots, 0, \gamma_{j,22}, 0, \ldots, 0),$$

then $\mathbf{g}_{j,I_j}^T = (\gamma_{j,2}, \gamma_{j,10}, \gamma_{j,22})$. Therefore, one can expect that

$$\mathbf{E}_j = \mathbf{X}_{I_j} - \sum_{a \neq j} \mathbf{d}_a \mathbf{g}_{a,I_j}^T \approx \mathbf{d}_j \mathbf{g}_{j,I_j}^T. \tag{55}$$

Consequently, one can update $\mathbf{d}_j$ and $\mathbf{g}_{j,I_j}^T$ by solving the following minimization problem

$$\mathrm{Armin}_{\mathbf{d}_j, \mathbf{g}_{j,I_j}^T} \left\| \mathbf{E}_j - \mathbf{d}_j \mathbf{g}_{j,I_j}^T \right\|_F^2 \quad \text{subject to} \quad \left\| \mathbf{d}_j \right\|_2 = 1\,, \tag{56}$$

where $\|\cdot\|_F$ denotes the Frobenius norm.

This problem can be solved by Singular Value Decomposition (SVD) or by an approximate solution proposed by Rubinstein et al. [62], where they accelerate the optimization process by seeking a sub-optimal solution instead through the following equations

$$\mathbf{d}_j = \mathbf{X}_{I_j}\mathbf{g}_{j,I_j} - \Big(\sum_{a\neq j}\mathbf{d}_a\mathbf{g}_{a,I_j}^T\Big)\mathbf{g}_{j,I_j}\,. \tag{57}$$

$$\mathbf{d}_j = \mathbf{d}_j/\|\mathbf{d}_j\|_2\,. \tag{58}$$

$$\mathbf{g}_{j,I_j}^{\mathrm{T}} = \mathbf{d}_j^{\mathrm{T}}\mathbf{X}_{I_j} - \mathbf{d}_j^{\mathrm{T}}\Big(\sum_{a\neq j}\mathbf{d}_a\mathbf{g}_{a,I_j}^T\Big)\,. \tag{59}$$

More information about the method can be found Rubinstein et al. [62] and Soares et al. [66].

To check if the sparse representation matrix is able to capture the main characteristics of the data, one can calculate the reconstructed dataset by

$$\mathbf{X}^{\mathrm{rec}} = \mathbf{D}\boldsymbol{\Gamma}, \tag{60}$$

and compare if $\mathbf{X}^{\mathrm{rec}}$ and $\mathbf{X}$ are similar. Luo et al. [49] used the following measure of information loss (L)

$$L = \frac{\|\mathbf{l}^{\mathrm{true}} - \mathbf{l}^{\mathrm{rec}}\|_2}{\|\mathbf{l}^{\mathrm{true}}\|_2} \times 100, \tag{61}$$

where $\mathbf{l}^{\mathrm{true}}$ is the quantity containing the true or reference data (after excluding noise, if any) and $\mathbf{l}^{\mathrm{rec}}$ the quantity with respect to the reconstructed data.

If one uses the 4D AVA dataset in seismic history matching, most of the data may just stem from 4D noise. Hence, the non-zero coefficients of the sparse matrix $\boldsymbol{\Gamma}$ may be concentrated on some specific parts of the matrix. Consequently, the reconstructed image may contain certain discontinuities. In order to reconstruct a smoother image, instead of using Eq. (60), Elad and Aharon[18] proposed the following formula

$$\mathbf{X}^{\mathrm{rec}} = (\lambda\mathbf{I})^{-1}(\lambda\mathbf{X} + \mathbf{D}\boldsymbol{\Gamma}), \tag{62}$$

which represents a weighted average between the reconstructed dataset ($\mathbf{D}\boldsymbol{\Gamma}$) and the original one. In Eq. (62), λ is a penalty (or weight) parameter, which in this work is chosen as

$$\lambda = \frac{\max(\mathbf{X})}{10\sigma}, \tag{63}$$

following the suggestion of Rubinstein et al. [62], where $\max$ is the maximum value of a property.

Parameters in the K-SVD method

From the previous introduction, one can see that there are several parameters involved in the K-SVD method, such as dictionary size (N_{k}), training dataset size (N_{ts}), patch size (n), and number of iterations (N_{it}). Depending on the objective and the dataset in the case study, the optimal values of these parameters may differ.

Relevant to the current study, Soares et al. [66] evaluated how each of these parameters affects the number of non-zero coefficients retained in the sparse matrix (N_{nz}) and the quality of the reconstructed image using a 4D seismic dataset. According to the authors, the most influential parameter on the number of non-zero coefficients is the training dataset size. The lower N_{ts} is, the smaller N_{nz} becomes. Hence, to make N_{nz} as small as possible, it is better to collect patches in the dataset without any overlapping

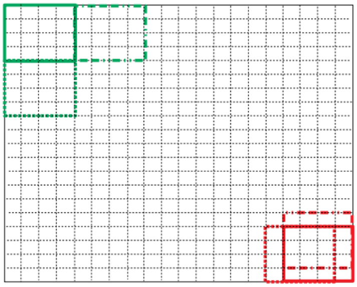


Figure 3: Example of training dataset and image patches.

between them. If one uses a higher N_{ts}, the quality of the reconstructed image tends to be better. However, one will end up with a larger number of non-zero coefficients.

To demonstrate how to select the training dataset according to the patch size, Fig. 3 shows a case with regular patches in the size of 4×4 (denoted by 4 hereafter). In the figure, the green color corresponds to the case in which one generates patches without any overlapping in-between, and thus achieves the minimum training dataset size N_{ts}. In this case, the step size $\Delta x \times \Delta y$, which is determined by the distances of a moving patch that travels along horizontal (x) and vertical (y) directions, is equal to the patch size (e.g., 4×4 in the current example). It is important to highlight that one should travel along each axis at a time to be able to encompass the whole grid. Meanwhile, in the same figure, we also use the red color to indicate the case with the maximum training dataset size N_{ts}, where the step size corresponds to 1×1 (denoted by 1 hereafter). Note that for image reconstruction, one should average the values from the overlapped patches.

Another important parameter is the dictionary size N_{k}, i.e., the number of atoms in the dictionary. Soares et al. [66] showed that the higher N_{k} is, the smaller N_{nz} one can achieve. However, the maximum number of N_{k} cannot be larger than the training dataset size N_{ts}.

In addition, there is a big trade-off concerning the number of iterations (N_{it}), since a larger N_{it} can indeed reduce the number N_{nz} of non-zero coefficients, but at the cost of an increased computational time. Finally, as discussed in Aharon et al. [2], larger image patches do not tend to work well with the K-SVD algorithm, and Soares et al. [66] reported that lower values of n can help to achieve lower N_{nz}.

The end products of the K-SVD algorithm are the values of the non-zero coefficients, their locations in $\mathbf{\Gamma}$ and the learned dictionary $\mathbf{D}$, which will be utilized in the subsequent 4D seismic history matching problem.

Sparse representation - Forward simulation

Sparse representation of the simulated data follows a similar procedure as that for the observed data. After obtaining sparse representation for the observed data, one should save the Dictionary $\mathbf{D}$ of the iteration $N_{\mathrm{it}} - 1$ and the location of the non-zero coefficients in the final $\mathbf{\Gamma}$ [1]. Hence, one can use the same procedure detailed in the subsection "Sparse Matrix Definition" and select the coefficients in the simulated $\mathbf{\Gamma}$ at the same position of the non-zero coefficients in the observed $\mathbf{\Gamma}$. Note that even if there

[1] The reason for us to use the dictionary $\mathbf{D}$ at the iteration step $N_{\mathrm{it}} - 1$, instead of N_{it}, is that in our code implementation, the sparse matrix $\mathbf{\Gamma}$ at the final iteration step N_{it} is obtained by solving the sparse representation problem Eq. (49), using $\mathbf{D}$ at the iteration step $N_{it} - 1$.

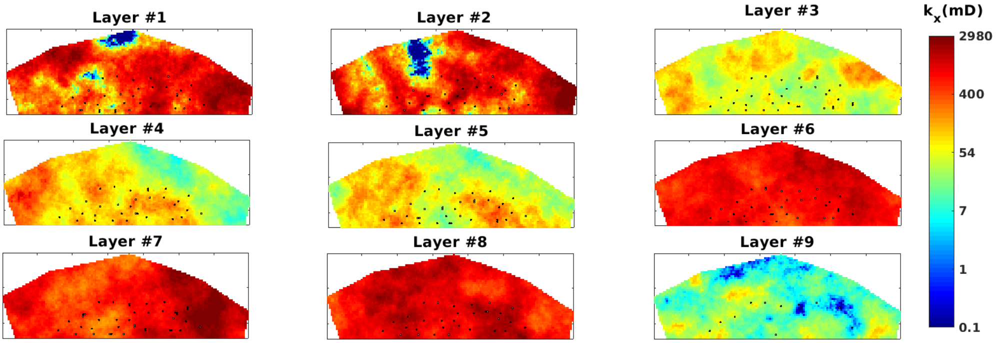


Figure 4: Horizontal permeability (k_x) from realization number 1.

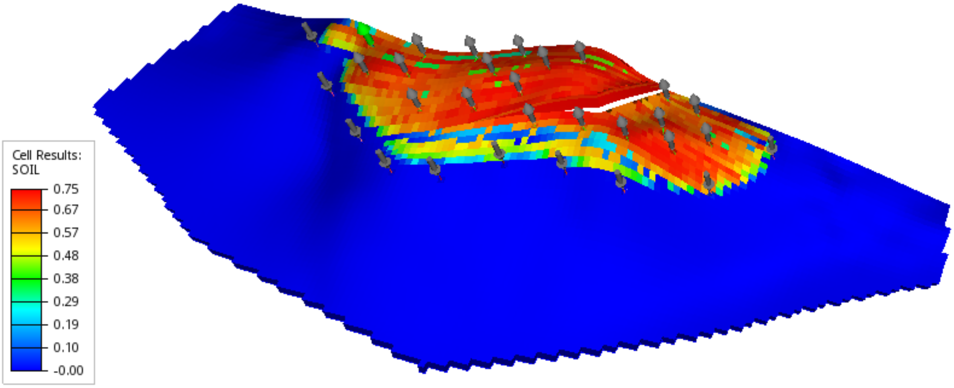


Figure 5: Distribution of the wells. The grid indicates the initial oil saturation.

are other non-zero coefficients in the simulated $\mathbf{\Gamma}$, they will not be considered.

Case study: Brugge Field

The case study used in this work is the Brugge benchmark [58], which consists of a reservoir with four production zones, two high and two low permeable zones. The two high permeable zones are in layers 1-2 and 6-8, with the averaged permeability values being 810 mD and 1 105 mD, respectively. The two low permeable zones are in layers 3-5 and 9, with the averaged permeability values being 90 and 36 mD, respectively. Fig. 4 shows the distributions of the permeability (along the x-direction) on each layer, with respect to one of the reservoir models from the initial ensemble. In addition, this benchmark is an isotropic case, i.e., the permeability in the y-direction is the same as in the x-direction. Note that we did not show the initial porosity maps for succinctness.

The numerical reservoir model of the Brugge Field has $139 \times 48 \times 9$ cells, summing up to 60 048 gridblocks in total, among which 44 550 are active. The model contains 20 producer and 10 injector wells, with water being the only injected fluid. Fig. 5 shows the distribution of the wells, where the injectors are located in the border and the producers are more centralized. The benchmark case has 10 years of production data, and we use a black oil simulator (ECLIPSE) for reservoir simulation.

Table 1: Information summary about the Brugge benchmark case study.

Model dimension	$139 \times 48 \times 9$ (60 048 cells)
Active cells	44 550
Average gridblock size	$93\,\mathrm{m} \times 91\,\mathrm{m} \times 5\,\mathrm{m}$
Wells	20 producers and 10 injectors
Production data	10 years (3 647.5 days)
4D seismic data	Near (10°) and far (20°) offsets amplitude-versus-angle (AVA) data
Seismic survey time	Base (day 1), Monitor #1 (day 991), Monitor #2 (day 2 999)
Seismic dimension	7 045 632 coefficients $(6 \times 139 \times 48 \times 176)$
Model variables	Porosity + Permeability along the x-,y- and z-directions. Total number: 178 200

There are 104 different realizations of reservoir models provided in the benchmark case. In this work, we randomly pick one of the models as the reference case to generate the "observed" seismic data, and let the rest compose the initial ensemble of 103 models. In each reservoir model from the initial ensemble, the uncertain parameters include the porosity and permeability in the x-, y-, and z-directions.

The seismic attributes in our experiments are the amplitude-versus-angle (AVA) data with two different offset angles, near (10°) and far (20°), from 3 different surveys: base (day 1), monitor #1 (day 991), and monitor #2 (day 2 999) [49]. The AVA dataset is on a seismic scale, so it has a different dimension compared to the reservoir model. In our experiments, the dimension of the AVA dataset at each offset angle and at each survey time is $139 \times 48 \times 176$, summing up to a total of 1 174 272 data points. Therefore, the total number of data points in our 4D seismic dataset is 7 045 632 (6 × 1 174 272).

To generate the AVA data, we require over- and under-burden layers and properties in addition to the reservoir model. Furthermore, the seismic trace/signal is recorded in some specific sampling rate in the time domain (z-axis in Fig. 6), which makes the dimension of the seismic traces (in time-domain) larger. In this work, we consider the whole cube of the AVA data as observed data (Fig. 6), however, most of the parts of the cubes are coming from over-burden and under-burden areas (non-reservoir parts). Therefore, the AVA dataset has a larger dimension if we compare it to the simulation model. Note that in applications with acoustic impedance datasets, the data is upscaled to the same dimension of the simulation model or even in a lower dimension, in which one can use fewer seismic horizons that are correlated with more than one layer in the simulation model, for instance.

Table 1 summarizes the main characteristics of the Brugge benchmark case, together with some of the experimental settings in the current work.

Methodology and Application

In the current work, we directly use the seismic attributes at three surveys as the observations in data assimilation. In principle, one may also try using the differences between different surveys as the observations. This would help to reduce the data size of 4D seismic, on the other hand, though, the signal-to-noise ratios in these difference datasets may become much lower, as is observed in our experiments (cf. upper-left plot in Fig. 22).

As indicated in Fig. 7, we divide our investigations into two main parts. In the first part, we evaluate the necessity of using sparse representation in a 4D seismic data assimilation problem by comparing the assimilation performance resulting from using

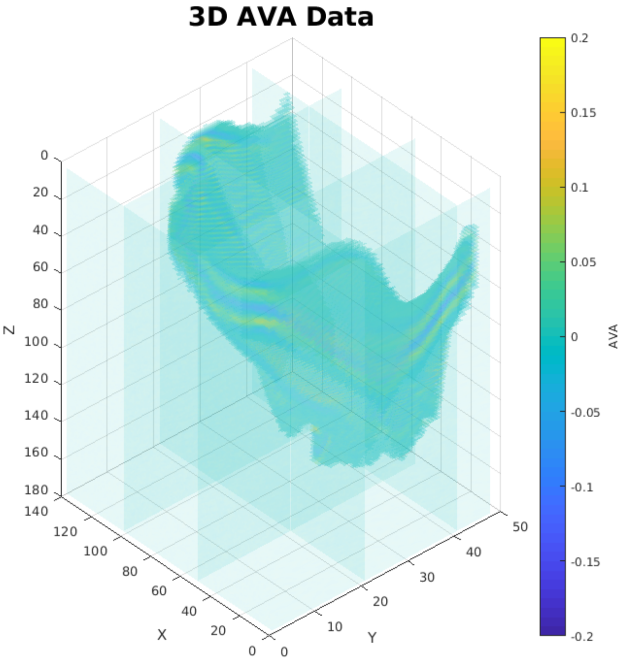


Figure 6: 3D AVA data (observed data): far trace in the base survey (day 1). The color bar in the right indicates the values of AVA data points.

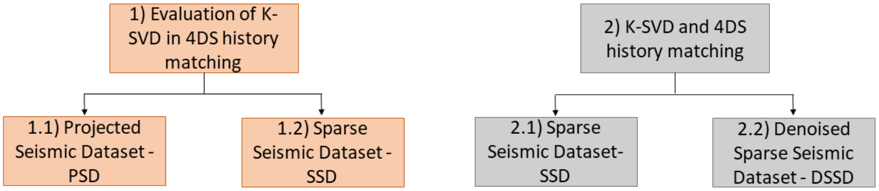

1) Evaluation of K-SVD in 4DS history matching
1.1) Projected Seismic Dataset - PSD
1.2) Sparse Seismic Dataset - SSD
2) K-SVD and 4DS history matching
2.1) Sparse Seismic Dataset- SSD
2.2) Denoised Sparse Seismic Dataset - DSSD

Figure 7: The two main parts in our investigations.

the Projected Seismic Dataset (PSD) and a Sparse Seismic Dataset (SSD) obtained from using the K-SVD algorithm. While in the second one, we aim to improve the data assimilation performance by using the K-SVD algorithm differently, where we first use the K-SVD algorithm to suppress the noise in the 4D seismic dataset and then use the same algorithm again for sparse data representation. For distinction, we call the end product of this procedure (using K-SVD twice) Denoised Sparse Seismic Dataset (DSSD).

For history matching, we use the IES-RML with the formulation shown by the Eq. (32). Note that with this formulation, we do not need to compute the truncated singular value decomposition, as most of the previous works did. In addition, we use the aforementioned three stopping criteria, where the maximum number of iterations is set to 20. Finally, we start with $\beta^0 = 1$, with the reduction and increment factor being 0.9 and 2, respectively.

Through the experiments in the three cases (PSD, SSD, and DSSD), we first evaluate if sparse representation can improve the performance of data assimilation at the expense of increased computational time and information loss. Additionally, we also inspect if denoising 4D seismic datasets before sparse representation can help to further improve the performance of data assimilation.

In the following, we explain in more detail how we perform data assimilation with respect to each approach (PSD, SSD, and DSSD).

Projected Seismic Dataset (PSD)

In this case, we use the full seismic dataset, i.e., the near and far offset of the three surveys, and project them onto the subspace, in the form of $\left(\mathbf{U}^i\mathbf{\Sigma}^i\right)^{\mathrm{T}}\widetilde{\mathbf{D}}^i$, as Eq. (40) suggests, for data-size reduction. In this way, we can use the whole dataset without incurring any computer-memory-related issue in the course of updating uncertain parameters, since we need to store the $\mathbf{K}_{\mathrm{eff}}$, which has a dimension of only $N_{\mathrm{x}} \times N_{\mathrm{e}}$. Note that in this case, we adopt the automatic and adaptive correlation-based scheme [47] for localization, in which the tapering matrix is constructed based on the correlations between model variables and projected data.

To estimate the measurement errors involved in data assimilation, we use the k-means clustering method [66], in which the seismic data are grouped into 3 clusters, one representing the noise and the other two containing positive and negative values that are more likely to be informative signals than noise. We can then calculate the standard

Table 2: K-SVD parameters in SSD.

Parameter	Value
Patch size (n)	8
Step size	8
Iterations (N_{it})	20
Dictionary size (N_{k})	750
Dictionary type	DCT

deviation of the noise cluster, and use it in Eq. (50) and to construct the observation error covariance matrix ($\mathbf{C}_{\mathrm{d}}$) for data assimilation. Note that Obidegwu et al. [56] and Davolio and Schiozer [15] also used k-means, but to cluster acoustic impedance data in softening and hardening effects, and Coleou et al. [14] pointed out other works that used k-means to cluster seismic waveforms for facies classification. Since the noise added consists of white noise, we assume $\mathbf{C}_{\mathrm{d}}$ to be diagonal, i.e., there is no correlation between the observation error. However, if one faces colored noises, it is possible to use the wavelet formulation to determine the noise as in Lorentzen et al. [44]. We chose here to use k-means because of its simplicity and good results reported previously.

Therefore, by projecting the whole 4D seismic dataset onto a subspace, it is possible to avoid the use of sparse representation while also achieving a significant data-size reduction. Note that in this case, the whole 4D seismic data, including the noise itself that does not contain information of the reservoir conditions, will be assimilated into reservoir models. This may downgrade the assimilation performance, as will be shown later.

Sparse Seismic Dataset (SSD)

To sparsely represent the 4D seismic dataset, we use the K-SVD method introduced before to find a representation with as few coefficients as possible while preserving the main characteristics of the data. In the experiments, we adapted the Matlab toolbox developed by Rubinstein et al. [62] to use the K-SVD algorithm. The original toolbox can be found at http://www.cs.technion.ac.il/ ronrubin/software.html. Based on the analysis of Soares et al. [66], we select the parameters involved in the calculation of the sparse matrix, as shown in Table 2. Note that the patch size and step size are equal in all three dimensions, i.e., $8 \times 8 \times 8$.

As we only use the non-zero coefficients during the data assimilation and, consequently, compress the data, it is possible to handle the Kalman gain without projecting them into the subspace of $\left(\mathbf{U}^i\mathbf{\Sigma}^i\right)^{\mathrm{T}}$ as in the previous approach. Therefore, we use the formulation in Eq. (36) in this step.

For the calculation of the $\mathbf{C}_{\mathrm{d}}$, we use the same procedure as before. Note that this is an approximation since we find the error in the original observed data and not in the sparsely represented data. The readers are referred to Raanes et al. [59] for more information about the error statistics and problems with correlated errors.

Denoised Sparse Seismic Dataset (DSSD)

Elad and Aharon [18] originally developed the K-SVD as a denoising tool. In line with this initiative, we apply the K-SVD algorithm in two steps: the first one (denoising) involves using the algorithm to achieve the best-reconstructed image without considering the number of non-zero coefficients N_{nz} retained. In the second step (compression), we apply the algorithm one more time but now focus on getting the best-reconstructed image with as few coefficients as possible. By suppressing noise in the original dataset,

Table 3: K-SVD configuration in the DSSD case.

Parameter	Denoising	Compression
Patch size (n)	8	8
Dimension	2D	3D
Step size	1	8
Iterations (N_{it})	10	50
Dictionary size (N_{k})	256	750
Dictionary type	data	data

we expect to have seismic datasets with better quality, which in turn may help achieve improved assimilation performance.

Due to the different objectives, the way we apply the K-SVD algorithm differs. Table 3 summarizes the configurations of the K-SVD algorithm in these two steps.

For the purpose of denoising, we treat each set of seismic attributes as a collection of 2D images and sequentially apply the K-SVD algorithm to denoise one 2D image at each time. Since the patch dimension for a 2D image is smaller, we believe that handling a 2D image each time helps to accelerate the process, as shown by Soares et al. [66]. Furthermore, as in this step we are not concerned about the number of non-zero coefficients, using a smaller patch size would not affect the efficiency of sparse representation. On the other hand, for the purpose of compression, we directly apply the K-SVD algorithm to 3D seismic datasets.

In the denoising step, we choose to use the maximum training dataset size N_{ts}, as this helps to generate better-reconstructed images. In contrast, in the compression step, we use the minimum N_{ts} instead to obtain fewer non-zero coefficients. For the same reason, the number of iterations is higher in the compression step, as this also helps to reduce the number of non-zero coefficients. In addition, we choose to increase the dictionary size to obtain less N_{nz}.

Instead of using DCT as the initial dictionary, we select random patches from the training dataset to construct our initial dictionary. Note that DCT is a more general approach, while the random patches is more specific for each case, which can help to generate fewer non-zero coefficients in the sparse matrix, as we will show later.

We use the same formulation for the history matching as before, Eq. (36), and to define the $\mathbf{C}_{\mathrm{d}}$, we use the denoised version of the observed data.

Results and Discussions

Results comparison between the PSD and SSD cases

The upper plot in Fig. 8 shows a sample inline seismic attribute, e.g, a vertical cross-section in the y-z plane from the far offset 3D AVA data at the base survey (day 1) (Fig. 6) and with the index number of x being 80. The first step is to define the noise in the seismic dataset and to estimate it, we use the k-means algorithm to cluster the seismic data into three regions (noise, positive, and negative values), as the lower plot in Fig. 8 illustrates. Then, we calculate the standard deviation (σ) of Cluster 1, which represents the noise. Table 4 shows the values of σ for each seismic data and the associated noise levels, which we calculate as in

$$\text{Noise level} = \frac{\text{noise variance}}{\text{pure signal variance}} \times 100\%. \tag{64}$$

Table 4: Seismic noise

Seismic data	Calculated σ	True σ	Noise level (%)
Far trace - base	0.0077	0.0074	32.92
Near trace - base	0.0154	0.0146	33.25
Far trace - monitor 1	0.0076	0.0073	32.89
Near trace - monitor 1	0.0156	0.0149	33.14
Far trace - monitor 2	0.0076	0.0072	32.85
Near trace - monitor 2	0.0156	0.0148	33.08

Luo et al. [49] stated that the noise level in the reference case is 30%. Hence, we observe that with k-means clustering, we achieved noise levels around 33%, which are close to that in the reference case.

In the PSD case, we select the full seismic dataset and project it into the smaller subspace $\left(\mathbf{U}^i\mathbf{\Sigma}^i\right)^{\mathrm{T}}$ as mentioned previously. Therefore, if we consider a random observed data $(\mathbf{d}_j)$ and project it onto $\left(\mathbf{U}^0\mathbf{\Sigma}^0\right)^{\mathrm{T}}$, we would achieve the following projected data

$$\mathbf{d}_j^{\mathrm{proj}} = \left(\mathbf{U}^0\mathbf{\Sigma}^0\right)^{\mathrm{T}}\mathbf{d}_j. \tag{65}$$

Hence, it is possible to reconstruct the signal $(\mathbf{d}_j^{\mathrm{rec}})$ and check how much information we loose as in

$$\mathbf{d}_j^{\mathrm{rec}} = \left(\left(\mathbf{U}^0\mathbf{\Sigma}^0\right)^{\mathrm{T}}\right)^{+}\mathbf{d}_j^{\mathrm{proj}}. \tag{66}$$

The upper plot in Fig. 9 depicts the reconstructed image after the projection and the information loss L is calculated through Eq. (61).

For the SSD approach, we use the characteristics described in Table 2 and calculate the error (Eq. (50)) using σ defined by the k-means. The lower plot in Fig. 9 shows the reconstructed image after applying the K-SVD algorithm in the SSD approach, where we calculate $\mathbf{X}^{\mathrm{rec}}$ through Eq. (60). Note that we use Eq. (60) once we want to show only the characteristics retained by the method.

If we compare the two plots in Fig. 9 with the original AVA dataset (Fig. 8), we notice that they both preserved the structure of the signal. However, we achieve a better-reconstructed image for the SSD case. The explanation for that lies in the fact that the projection can be interpreted as a smooth function. It is possible to see this if one looks at the region with values very close to zero in the original image (upper plot in Fig. 8), and this region has values of zero in the reconstructed image (upper plot in Fig. 9). Hence, this smooth property will also happen in the whole dataset, including the part that we have values that are not too close to zero. In addition, the ensemble may not be able to accommodate all the information contained in the observed data, and the iterative learning process in the SSD helped to generate a better-reconstructed image. Consequently, the projection downgraded the image reconstruction compared to the SSD case.

After the application of the K-SVD algorithm for the entire dataset (far and near offset traces at base, first monitor and second monitor), we achieved a total of 19 055 coefficients, which represents only 0.27% of the entire dataset.

Even though the scope of this work is not to compare different methodologies for sparse representation, we show some differences between the wavelet-based sparse representation [49] and the K-SVD based one. In the methodology developed by Luo et al. [49], there is a need for defining a threshold value (t) that is used to select the

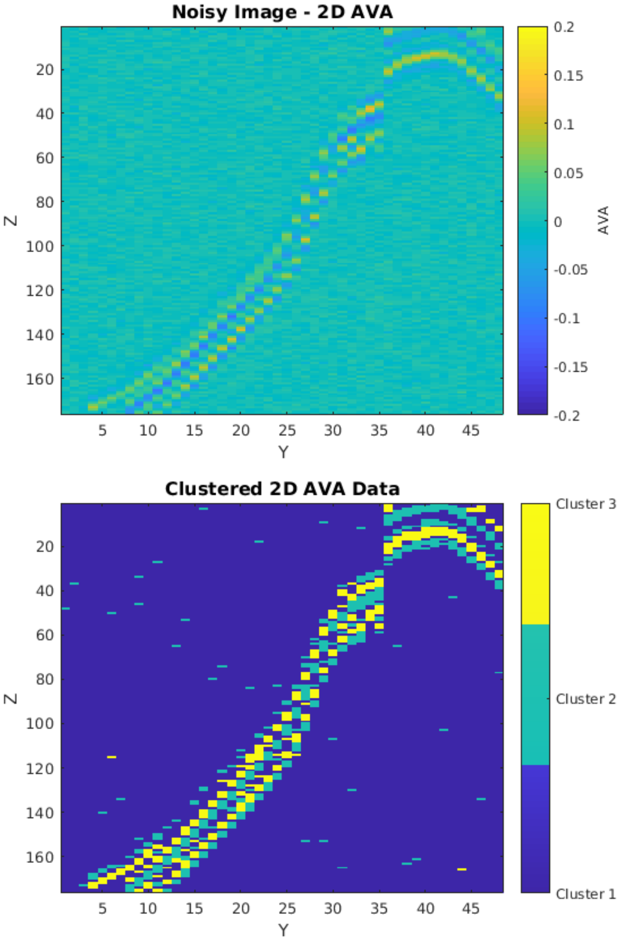


Figure 8: Far trace AVA data - base survey: Noisy image (upper); Clustered image (lower). The horizontal axis represents the y-dimension as in Fig. 6 and the vertical axis represents the z-dimension as in Fig. 6. The color bar in the right indicates the AVA values.

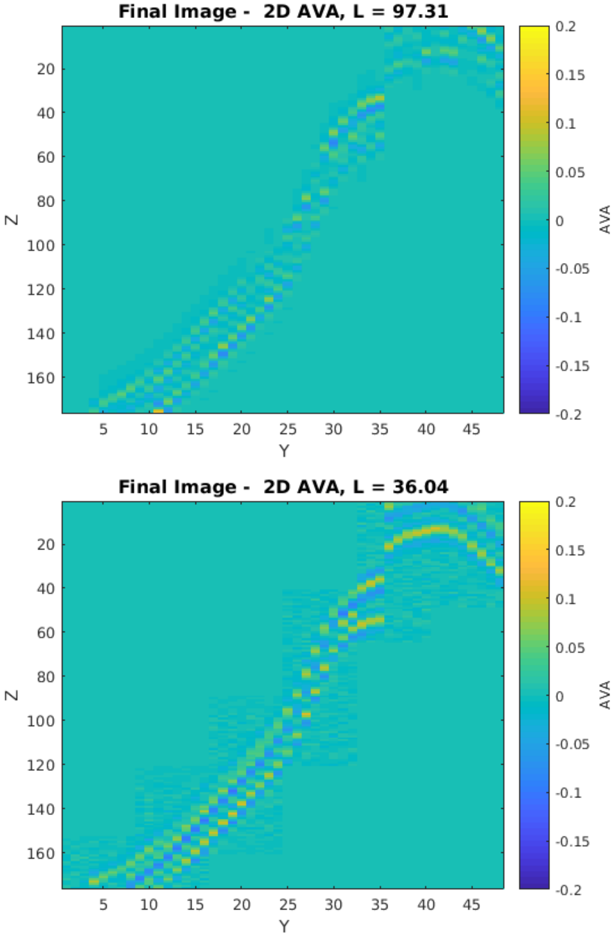


Figure 9: Reconstructed far trace AVA data - base survey: PSD (upper); SSD (lower). The horizontal axis represents the y-dimension as in Fig. 6 and the vertical axis represents the z-dimension as in Fig. 6. The color bar in the right indicates the AVA values.

non-zero coefficients for sparse representation. Fig. 10 depicts the case for $t = 1$ (upper plot) and for $t = 5$ (lower plot). The focus here is on the trade-off between the value of t and the quality of the reconstructed image. When t equals 5, the image does not retain the main features of the original seismic data. However, the number of non-zero coefficients retained was only 3 293 (0.047% of the original seismic dataset). For the case where t is equal to 1, the final reconstructed image is very similar to the original data, but the amount of non-zero coefficients is much larger, up to 178 332 (2.53% of the original dataset).

Therefore, in this case, the K-SVD algorithm tends to generate better-reconstructed images with fewer non-zero coefficients, while for the wavelet-based sparse representation it remains to be a challenge to find a suitable trade-off between the retained characteristics in the reconstructed data and the number of non-zero coefficients.

The analysis of data assimilation performance starts with the calculation of the data mismatch through Eq. (23) without the sum, i.e., we calculate the data mismatch for each ensemble member and plot in a boxplot for every iteration as Fig. 11 depicts. We observe that both approaches tend to reduce the mismatch between the simulated and observed data. In the PSD case (Fig. 11 - upper plot), it is possible to see that the rate of mismatch reduction tends to be slower than that in the SSD case (Fig. 11 - lower plot). Nevertheless, it is difficult to directly compare the data mismatch from both cases as we use different sets of observed data. Note that we could normalize the mismatch, but in the PSD case, most of the observed data concerns the non-reservoir part. Hence, during the forward simulation, we set these values to zero, and consequently, as the observed data in these parts are also very close to zero, the normalized mismatch tends to be much lower for the PSD case. Therefore, it is better to show how the estimated model variables look like in the final ensemble.

We use the Root Mean Square Error (RMSE), as in Luo et al. [49], to measure the distances between the estimated model variables and the reference (true) model.

Fig. 12 shows the RMSE of porosity values for PSD (upper) and SSD (lower). In both cases, RMSE values tend to decrease as the iterations proceed. One can see that using SSD leads to lower RMSE values at the end, i.e., models closer to the reference one.

By inspecting the ensemble mean for the PSD and SSD cases (third and fourth plots in Fig. 13, respectively), one can see that there is a stronger update of the porosity values in the latter case. The ensemble mean obtained in the PSD case remains closer to the mean of the initial ensemble (first plot in Fig. 13), while the one from the SSD case is closer to the reference model (second plot in Fig. 13).

More insights can be gained by inspecting the tapering matrix produced by the localization scheme. Note that the tapering matrix is in the same dimension as the Kalman Gain matrix. Hence, in the PSD case, it is a matrix in the dimension of $N_{\mathrm{x}} \times N_{\mathrm{e}}$. In contrast, in the SSD case, it has the dimension of $N_{\mathrm{x}} \times N_{\mathrm{d}}$, where N_{d} is the number of observations (in this case, the number of non-zero coefficients). Following this perspective, each column of a tapering matrix indicates how an observation element is correlated with the model variables. Since in our experimental settings, the PSD and SSD cases have different amounts of observations, we check the mean of columns and plot the tapering coefficient values $c(z)$ (Eq. (44)) onto the reservoir model gridblocks.

Fig. 14 shows the tapering coefficient values using correlations between porosity and the selected observation element, distributed on the reservoir model gridblocks at Layer

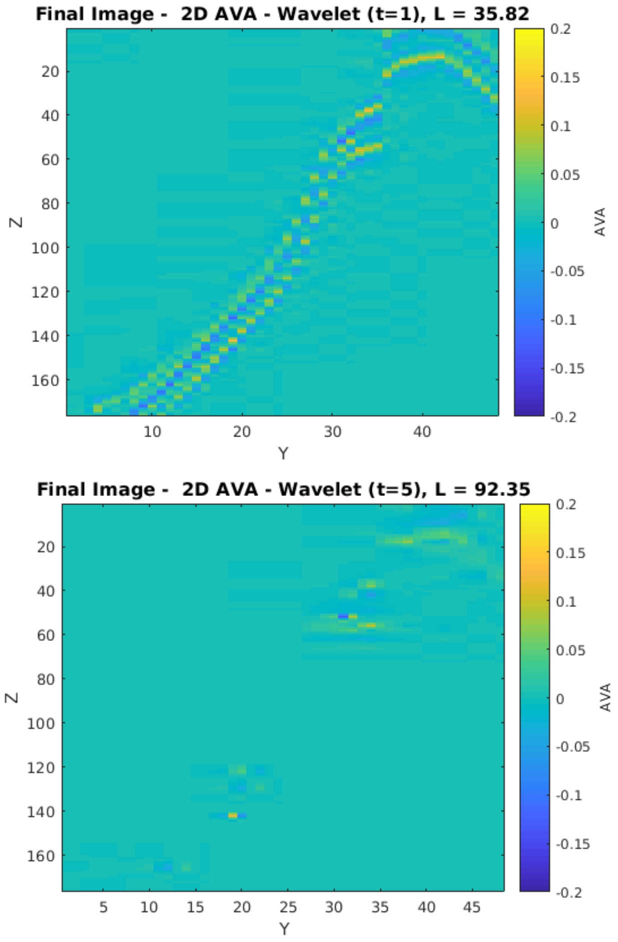


Figure 10: Reconstructed far trace AVA data - base survey - Wavelet based: t=1 (upper); t=5 (lower). The horizontal axis represents the y-dimension as in Fig. 6 and the vertical axis represents the z-dimension as in Fig. 6. The color bar in the right indicates the AVA values.

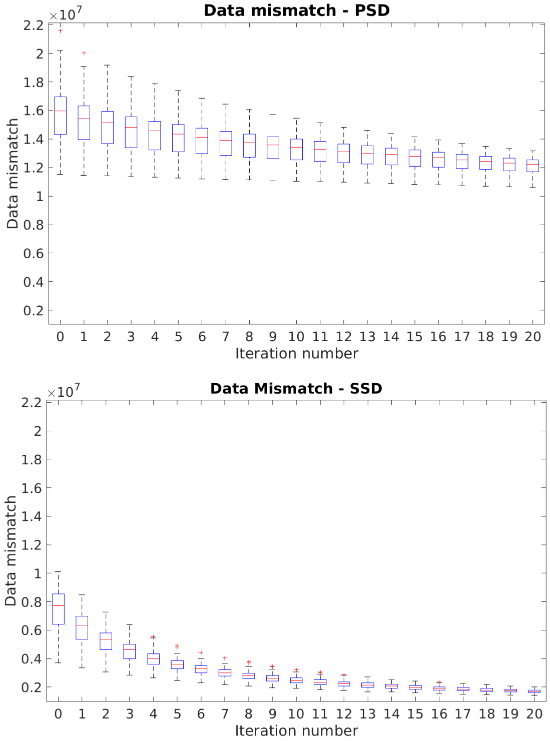


Figure 11: Data mismatch: Projected Seismic Dataset - PSD (upper); and Sparse Seismic Dataset - SSD (lower).

2, in cases of PSD (upper) and SSD (lower). In the PSD case (Fig. 14 - upper), the tapering coefficient values are relatively low. In contrast, in the SSD case (Fig. 14 - lower), the tapering coefficient values tend to be higher. Therefore, the final ensemble of the PSD case tends to remain closer to the initial one, while the one from the SSD case experiences stronger updates and moves closer to the reference model.

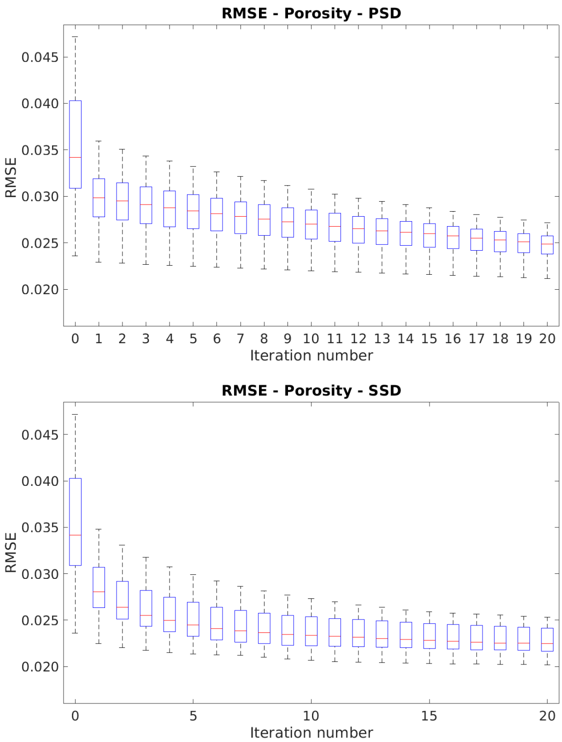
We also examine RMSEs concerning the permeability. Fig. 15 shows that the RMSE of permeability (along the x-direction) does not change as much as porosity since seismic data tend to be less sensitive to permeability in this benchmark case [49]. Similar results are observed for the estimated permeability values along the y- and z-directions. We do not show the permeability maps for succinctness.

In conclusion, projecting observations onto an ensemble subspace and sparse data representation (through K-SVD) may serve as two viable ways to handle big data assimilation problems in practice. In this benchmark case study, it shows that sparse data representation tends to result in better assimilation performance than the subspace-projection based method.

Results comparison between the SSD and DSSD cases

The results from the SSD case are the same as in the preceding subsection. We choose to show some of the results again here for comparison.

In the DSSD case, we first denoise the original dataset using the K-SVD algorithm, whose configuration is indicated in the middle column of Table 3. We use the same estimated noise standard deviations σ as before and apply the K-SVD algorithm to 2D



RMSE - Porosity - PSD

RMSE - Porosity - SSD

Figure 12: RMSE - Porosity: Projected Seismic Dataset - PSD (upper); and Sparse Seismic Dataset - SSD (lower).

images (slices in the x-direction) 139 times. The upper plot in Fig. 16 illustrates the image obtained by applying the K-SVD algorithm for the purpose of denoising. One can see that information loss ($L = 13.61$) is smaller than that in the SSD case ($L = 36.04$). Note that we use Eq. (62) to promote continuity in the reconstructed image.

Denoising of seismic data is a very important topic within the geoscience community. Consequently, there are many works focused on this particular matter. For instance, Baddari et al. [6] developed a non-linear diffusion filter capable of reducing random and Gaussian noise, Xiong et al. [70] proposed a random noise attenuation in the time-frequency domain, and Hennenfent and Herrmann [37] presented the non-uniformly sampled curvelets to denoised seismic images. There's a vast list in the literature with different methodologies, and since it is not the scope of this work to compare different denoising methods, we can refer the readers more works about this topic [40, 64, 34, 7, 72].

Subsequently, we use the k-means clustering method one more time to obtain new values of estimated noise standard deviations σ, which are reported in Table 5. Comparing the results there with those in Table 4, one can see that the new σ values are lower, indicating the impact from the preceding denoising procedure.

Note that we do not use DCT as the initial dictionary in the DSSD case. This is because during the second step (compression), we achieve a relatively large number of non-zero coefficients by using DCT (due to the lower error). However, when we use random patches from our original dataset to construct the initial dictionary (called data

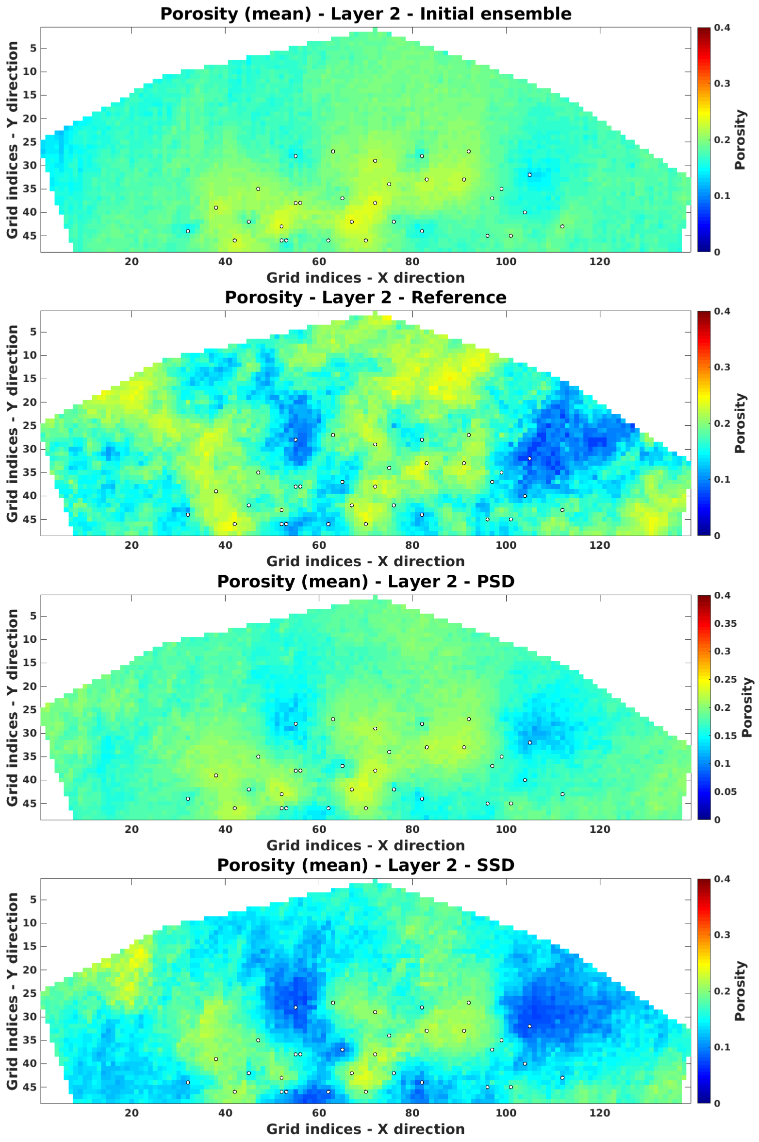

Porosity (mean) - Layer 2 - Initial ensemble
Porosity - Layer 2 - Reference
Porosity (mean) - Layer 2 - PSD
Porosity (mean) - Layer 2 - SSD
Grid indices - Y direction
Grid indices - X direction
5
10
15
20
25
30
35
40
45
20
40
60
80
100
120
Porosity
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4

Figure 13: Porosity maps at Layer 2 of the reservoir. From top to bottom: initial ensemble mean; reference map; final ensemble mean from the PSD case; final ensemble mean from the SSD case.

Table 5: Seismic noise after denoising

Seismic data	Calculated σ
Far trace - base	0.0034
Near trace - base	0.0070
Far trace - monitor 1	0.0033
Near trace - monitor 1	0.0071
Far trace - monitor 2	0.0033
Near trace - monitor 2	0.0070

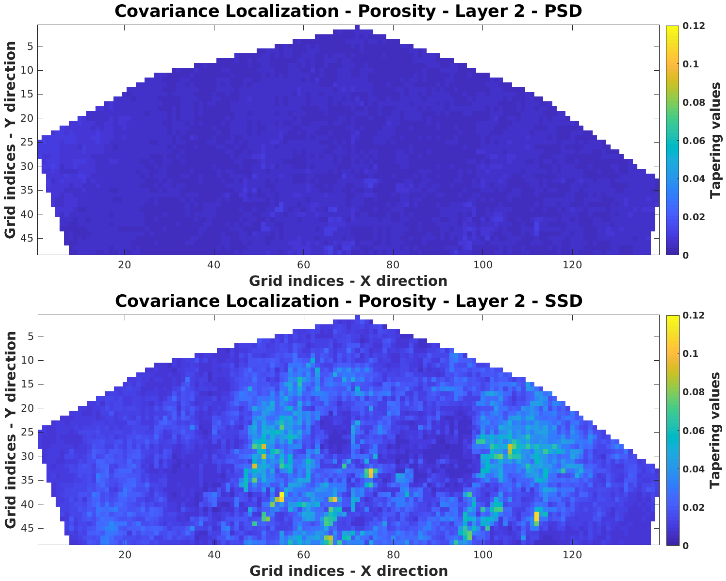


Figure 14: Tapering coefficient values (for porosity) distributed at Layer 2 of the reservoir model gridblocks: Projected Seismic Dataset - PSD (upper); and Sparse Seismic Dataset - SSD (lower).

dictionary hereafter), we can reduce the number of non-zero coefficients. Therefore, to maintain the consistency between the denoising and compression procedures, we prefer to use the data dictionary in both steps. The information loss (L) of the reconstructed image from a sparse representation of the denoised data is slightly higher than that of the denoised data itself (cf. lower plot in Fig. 16). In this sense, we achieve minimal information loss within our experiment settings in the DSSD case.

After the denoising and compression procedures, we obtain a total amount of 38 876 non-zero coefficients, which represents 0.55% of the original dataset. Note that here we end up with about twice the number of non-zero coefficients in the SSD case. However, in terms of information loss, we obtain better-reconstructed images from sparse representations of the original dataset in the DSSD case (cf. the lower plot in Fig. 16).

The upper and lower plots of Fig. 17 illustrate the box plots of data mismatch at different iterations steps in the SSD and DSSD cases, respectively. Since both cases now have observed data concerning only the reservoir part, it is possible to normalize the mismatch and directly compare them. As one can notice, SSD presented lower values for the mismatch. However, the difference between the two cases is not very big. Furthermore, even though the uncertainty in the mismatch is lower in the SSD, this does not reflect in the uncertain model variables (porosity and permeabilities) as the spread (standard deviation) of the final ensemble is very similar for both cases. To complement the analysis, we also compare the estimated reservoir models with the reference one.

Fig. 18 shows the boxplots of RMSE (for porosity) at different iterations steps in the SSD (upper) and DSSD (lower) cases. As one can see, the final RMSE values tend to be close in both cases, with those from the DSSD case tending to be lower. In addition, one can observe that in the SSD case, the RMSE values seem to enter a plateau

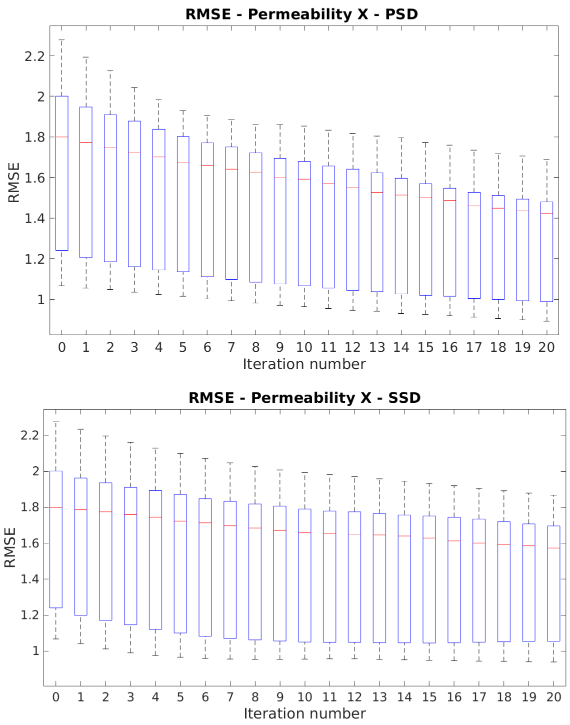


Figure 15: RMSE - Permeability x: Projected Seismic Dataset - PSD (upper); and Sparse Seismic Dataset - SSD (lower).

from the 10th iteration step on, while (although not verified) those in the DSSD case appear to have room for further reduction, following the exhibited trend therein. For RMSEs with respect to the permeabilities, we obtain similar results to those in the SSD case, possibly due to the weak correlations between seismic data and permeabilities, as mentioned before (Fig. 19).

The third and fourth panels of Fig. 20 show the mean porosity maps (of the final ensembles) at Layer 2 of the reservoir in the SSD and DSSD cases, respectively. One can see that both porosity maps capture some of the prominent geological structures in the reference one (second panel of the same figure). Visually, these two estimated porosity maps are similar, although the RMSE metric (cf. Fig. 18) indicates that the one from the DSSD case tends to be better. Similar results are also found in the permeability (along the x-direction) maps, as demonstrated in Fig. 21.

After showing the relative superiority (in terms of RMSE) in using DSSD, we also consider some of the differences between the SSD and DSSD cases from the perspective of computational resources. In the Brugge benchmark case, from the model side, one reservoir model contains 44 550 active gridblocks, and each gridblock has 4 uncertain model variables (porosity and permeabilities along three directions) to estimate. Hence, we have a total of 178 200 uncertain model variables (N_x) in data assimilation. In addition, from the observation side, we have 38 876 data points (N_d) in the DSSD case, resulting in a Kalman gain matrix ($\mathbf{K}$) in the dimension of $178\,200 \times 38\,876$. In contrast, in the SSD case, the Kalman gain matrix ($\mathbf{K}$) is in the dimension of $178\,200 \times 19\,055$ instead, which is somewhat easier to handle in terms of computational time and memory.

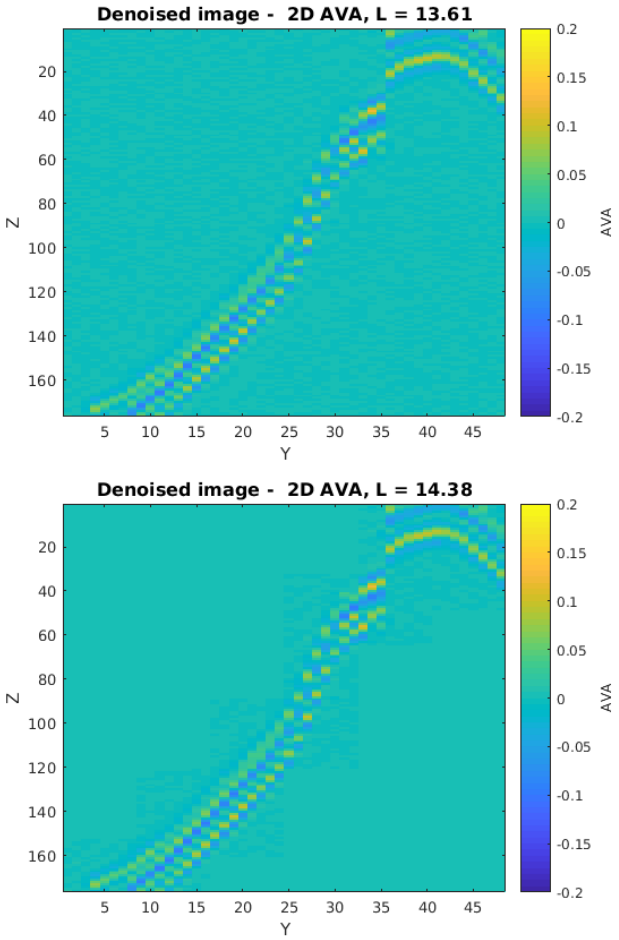


Figure 16: Far trace AVA data - base survey: Denoised data (upper); and Reconstructed data from a sparse representation of the denoised data (lower). The horizontal axis represents the y-dimension as in Fig. 6 and the vertical axis represents the z-dimension as in Fig. 6. The color bar in the right indicates the AVA values.

Regarding the computational time dedicated to sparse data representation, in the DSSD case, it takes about 40 minutes to go through both procedures of denoising and sparse representation. On the other hand, it takes only around 50 seconds in the SSD case. This substantial gap between the computational time is largely due to the time-consuming denoising procedure adopted in the DSSD case, where a minimum step size ($1{\times}1$) is adopted so that it creates a much larger number of overlapping patches to achieve high-quality denoised images. Furthermore, one would need a relatively large RAM memory to handle all the matrices during history matching. For instance, to calculate the Kalman gain $\mathbf{K}$ for the SSD and DSSD cases, one would need about 25 and 50 GB, respectively. This number seems to be very big, but it can be avoided by computing the Kalman gain several times considering different lines of the matrix at each time [20], and one can also use local analysis to avoid these big matrices [8, 24, 13, 63].

Finally, to demonstrate the benefits of using DSSD in the Brugge benchmark, we

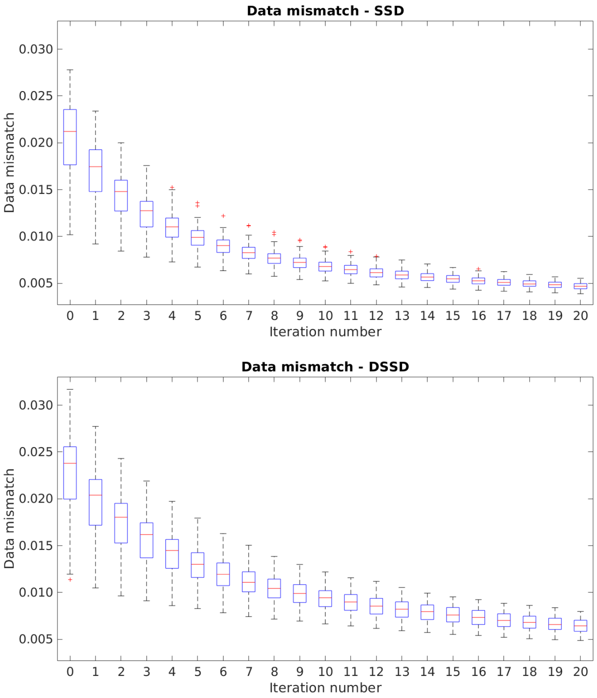


Figure 17: Normalized data mismatch: Sparse Seismic Dataset - SSD (upper); and Denoised Sparse Seismic Dataset - DSSD (lower).

examine the 4D changes of seismic data reconstructed from the original dataset and their sparse representations. Fig. 22 shows the differences between a slice of far offset trace at the monitor survey #1 and the same slice of trace data at the base survey. In other words, we subtract $\mathbf{X}^{\text{rec}}$ at monitor #1 from $\mathbf{X}^{\text{rec}}$ at the base survey in the SSD and DSSD cases. One can see that in the upper-left panel, the difference between the two surveys in the original dataset is very noisy and very different from the reference case (lower-right panel). In the SSD case (upper-right panel), the difference is still big from the reference case, while the differences in the DSSD case (lower-left panel) appear to be more similar.

When we inspect the 4D differences in the SSD case (upper-right panel of Fig. 22) without the denoising procedure, the noise kept in the reconstructed data may sum up and downgrade the quality of the 4D seismic data. In turn, this may deteriorate the performance of data assimilation, if one wishes to use the 4D differences as the observations [2].

In contrast, in the DSSD case (lower-left panel of Fig. 22), by denoising the original seismic data first, we obtain reconstructed images with improved qualities (cf. Fig. 16). Consequently, the difference between the survey is more physically consistent, as we can see the 4D effect better, and the resulting 4D differences may resemble the reference case (lower-right panel) better, which is also reflected by the lower RMSE values.

[2] We note that an alternative way is to first calculate the 4D differences, apply the K-SVD algorithm, and then reconstruct the 4D differences from their sparse representations. However, as aforementioned, in this case, the signal-to-noise ratios in 4D difference data become very low such that it becomes very difficult to extract true seismic signals from the noisy data (upper-left plot in Fig. 22).

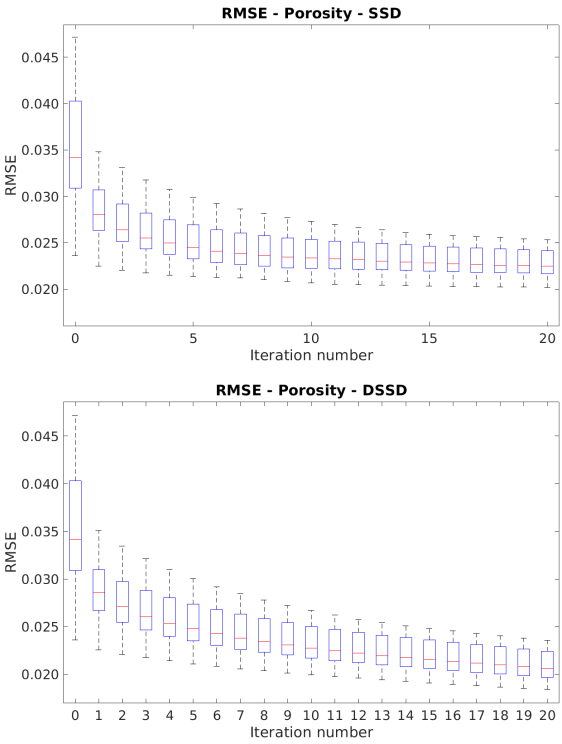


Figure 18: RMSE - Porosity: Sparse Seismic Dataset - SSD (upper); and Denoised Sparse Seismic Dataset - DSSD (lower).

Therefore, despite higher computational time and memory required, adopting DSSD tends to capture the 4D effect better in the Brugge benchmark, resulting in a final ensemble closer to the reference case than using SSD. It is important to mention that for the Brugge benchmark, the original AVA data has a relatively low signal-to-noise ratio. Hence, the denoising process helped to achieve better-reconstructed images and better preservation of the 4D effect. However, some other cases may present a higher signal-to-noise ratio and a good visualization of the 4D effect without denoising the signal. In such cases, since the SSD case is faster, we suggest using SSD instead of DSSD. In other words, one can use different criteria to decide which case to use, one can either have a larger "weight" on the computational resources or on a better final model.

Besides, if we compare the results of the SSD and DSSD cases with that from the previous work [49, 47], in which the authors used the wavelet-based sparse representation, the cases presented here tend to be more convenient in achieving a suitable trade-off between sparse data representation and preservation of data information.

Conclusions

We present a 4D seismic data assimilation framework in which a dictionary learning algorithm (K-SVD) is adopted for sparse data representation, instead of using a wavelet-based sparse representation procedure or deep learning as in previous works. Through

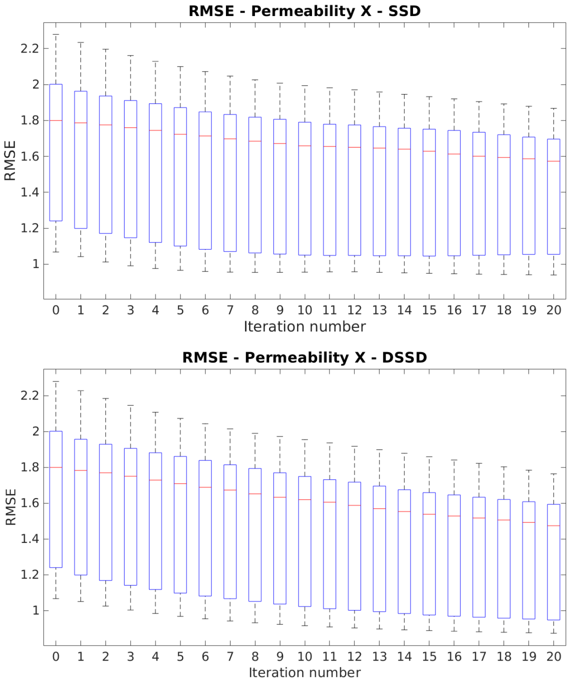
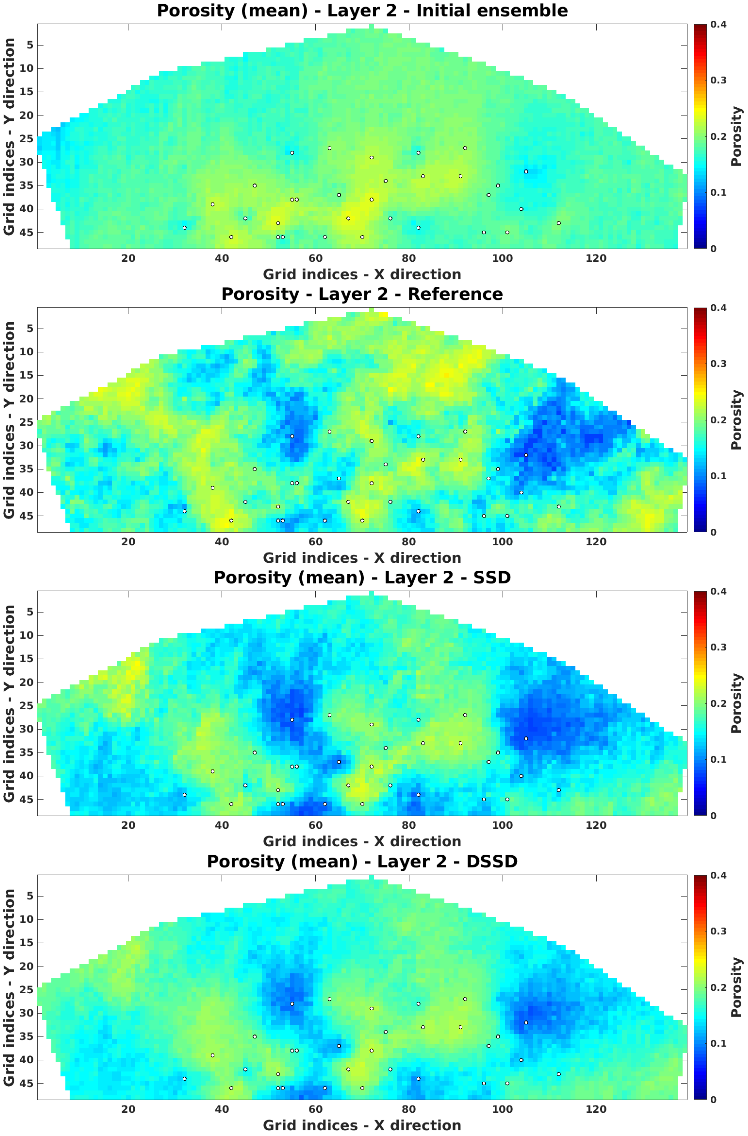


Figure 19: RMSE - Permeability X: Sparse Seismic Dataset - SSD (upper); and Denoised Sparse Seismic Dataset - DSSD (lower).

numerical experiments in the Brugge benchmark case, we show that dictionary-learning based sparse representation can serve as an efficient way to handle big data assimilation problems.

We also consider an alternative way to handle big seismic data by projecting them onto an ensemble subspace. In the investigated benchmark case, it turns out that the projected data tend to exhibit relatively weak correlations with the model variables under estimation. As a result, within our experimental settings, the resulting final ensemble of updated reservoir models remains close to the initial ensemble and thus leads to relatively inferior assimilation performance. In contrast, through dictionary-learning based sparse representation, the corresponding representation coefficients tend to have stronger correlations with the model variables under estimation. This makes it possible for the updated reservoir models to experience more substantial changes and move closer to the reference model.

In an additional investigation, we introduce a denoising procedure before applying dictionary-learning based sparse representation to the 4D seismic. Although incurring higher computational time and memory, our experiments in the Brugge benchmark indicate that this additional procedure helps to improve the performance of data assimilation in terms of the RMSE metric, in cases where the original seismic signal is noisy. However, if the original signal has a high signal-to-noise ratio (low noise), the denoising process might be less useful, since it incurs a higher computational time. Hence, in cases where the original signal is not very noisy and the computational resources are limited, the SSD case may be preferred.



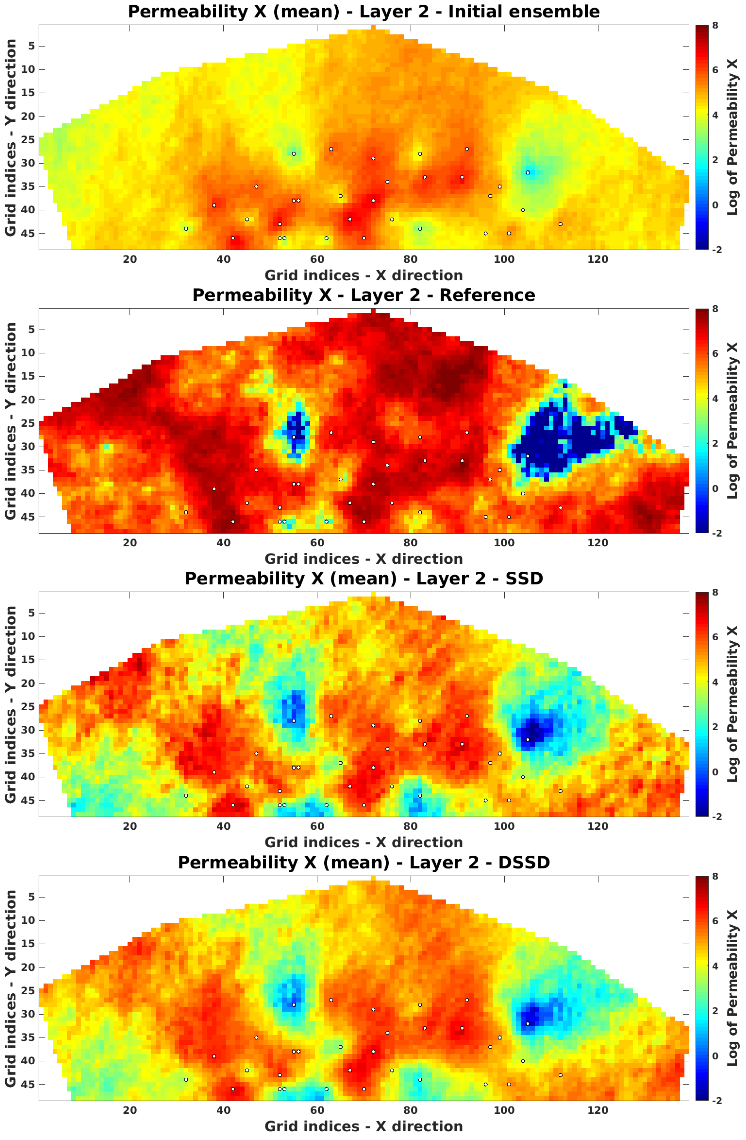
Porosity (mean) - Layer 2 - Initial ensemble

Porosity - Layer 2 - Reference

Porosity (mean) - Layer 2 - SSD

Porosity (mean) - Layer 2 - DSSD

Figure 20: Porosity maps at Layer 2 of the reservoir. From top to bottom: initial ensemble mean; reference map; final ensemble mean from the SSD case ; final ensemble mean from the DSSD case.



Permeability X (mean) - Layer 2 - Initial ensemble

Permeability X - Layer 2 - Reference

Permeability X (mean) - Layer 2 - SSD

Permeability X (mean) - Layer 2 - DSSD

Figure 21: Log of permeability x maps at Layer 2 of the reservoir. From top to bottom: initial ensemble mean; reference map; final ensemble mean from the SSD case ; final ensemble mean from the DSSD case.

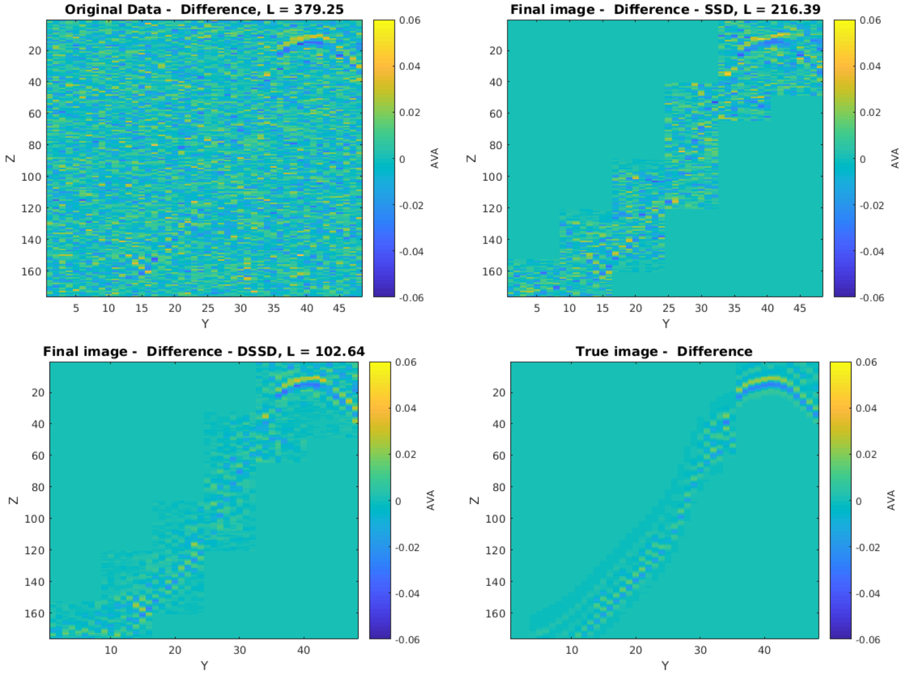


Figure 22: Differences of a slice of far AVA trace data between monitor #1 and base surveys: original dataset (upper left); SSD (upper right); DSSD (lower left); reference case (lower right). The horizontal axis represents the y-dimension as in Fig. 6 and the vertical axis represents the z-dimension as in Fig. 6. The color bar in the right indicates the AVA values.

Finally, even though we reduced the AVA dataset to only 0.55% of the original dataset in the DSSD case, this number still appears too big. Therefore, as our future work, we will consider applying local analysis [23, 63, 13] to handle big datasets more efficiently. In addition, we also plan to test the approaches developed here in a different dataset for further performance validation.

CRediT autorship contribution statement

R.V. Soares: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data curation, Writing - original draft, Visualization. **X. Luo:** Conceptualization, Methodology, Software, Formal analysis, Investigation, Data curation, Writing - review and editing, Supervision. **G. Evensen:** Conceptualization, Methodology, Writing - review and editing, Supervision, Project adminsitration, Funding acquisition. **T. Bhakta:** Methodology, Software, Data curation, Writing - review and editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appear to influence the work reported in this paper.

Acknowledgments

The authors acknowledge financial support from the Research Council of Norway through the Petromaks-2 project DIGIRES (RCN no. 280473) and the industrial partners AkerBP, Wintershall DEA, Vår Energi, Petrobras, Equinor, Lundin, and Neptune Energy. We also thank Schlumberger for providing academic software licenses to ECLIPSE.

References

[1] S. I. Aanonsen, G. Naevdal, D. S. Oliver, A. Reynolds, and B. Valles, *Ensemble Kalman Filter in Reservoir Engineering – A Review*, SPE Journal, SPE-117274-PA, 14(3) (2009), pp. 393–412.

[2] M. Aharon, M. Elad, and A. Bruckstein, *K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation*, IEEE Transactions on Signal Processing, 54 (2006), pp. 4311–4322.

[3] M. Alfonzo and D. S. Oliver, *Seismic data assimilation with an imperfect model*, Comput Geosci, (2019), pp. 889 – 905.

[4] F. L. R. Almeida, A. Davolio, and D. J. Schiozer, *Systematic approach to reduce uncertainties when quantitatively assimilating 4d seismic and well data*, SPE Reservoir Evaluation & Engineering, 23 (2020), pp. 1–12. SPE-187081-PA.

[5] P. Avseth, T. Mukerji, and G. Mavko, *Quantitative seismic interpretation: applying rock physics tools to reduce interpretation risk*, Cambridge Univ. Press, Cambridge, 2010.

[6] K. Baddari, J. Ferahtia, T. Aifa, and N. Djarfour, *Seismic noise attenuation by means of an anisotropic non-linear diffusion filter*, Computers and Geosciences, 37 (2011), pp. 456 – 463.

[7] D. Bonar and M. Sacchi, *Denoising seismic data using the nonlocal means algorithm*, GEOPHYSICS, 77 (2012), pp. A5–A8.

[8] K. Brusdal, J. Brankart, G. Halberstadt, G. Evensen, P. Brasseur, P. J. van Leeuwen, E. Dombrowsky, and J. Verron, *An evaluation of ensemble based assimilation methods with a layered OGCM*, J. Marine. Sys., 40-41 (2003), pp. 253–289.

[9] S. W. Canchumuni, A. A. Emerick, and M. A. C. Pacheco, *History matching geological facies models based on ensemble smoother and deep generative models*, Journal of Petroleum Science and Engineering, 177 (2019), pp. 941 – 958.

[10] Y. Chen and D. S. Oliver, *Ensemble Randomized Maximum Likelihood Method as an Iterative Ensemble Smoother*, Mathematical Geosciences, 44 (2012), pp. 1–26.

[11] Y. Chen and D. S. Oliver, *Levenberg-Marquardt forms of the iterative ensemble smoother for efficient history matching and uncertainty quantification*, Computat Geosci, 17 (2013), pp. 689–703.

[12] Y. Chen and D. S. Oliver, *History Matching of the Norne Full-Field Model With an Iterative Ensemble Smoother*, SPE Reservoir Evaluation & Engineering, 17 (2014), pp. 244–256. SPE-164902-PA.

[13] ——, *Localization and regularization for iterative ensemble smoothers*, Comput Geosci, 21 (2017), pp. 13–30.

[14] T. Coleou, M. Poupon, and K. Azbel, *Unsupervised seismic facies classification: A review and comparison of techniques and implementation*, The Leading Edge, 2 (2012), pp. 942–953.

[15] A. Davolio and D. J. Schiozer, *Probabilistic seismic history matching using binary images*, Journal of Geophysics and Engineering, 15 (2018), pp. 261–274.

[16] D. Devegowda, E. Arroyo-Negrete, A. Datta-Gupta, and S. G. Douma, *Efficient and robust reservoir model updating using ensemble Kalman filter with sensitivity-based covariance localization*, in SPE Reservoir Simulation Symposium, 2007. SPE 106144.

[17] D. L. Donoho and I. M. Johnstone, *Adapting to unknown smoothness via wavelet shrinkage*, Journal of the American Statistical Association, 90 (1995), pp. 1200–1224.

[18] M. Elad and M. Aharon, *Image denoising via sparse and redundant representations over learned dictionaries*, IEEE Transactions on Image Processing, 15 (2006), pp. 3736–3745.

[19] A. Emerick and A. Reynolds, *History Matching a Field Case Using the Ensemble Kalman Filter With Covariance Localization*, SPE Reservoir Evaluation & Engineering, 14 (2011), pp. 443–452. SPE-141216-PA.

[20] A. A. Emerick, *Analysis of the performance of ensemble-based assimilation of production and seismic data*, Journal of Petroleum Science and Engineering, 139 (2016), pp. 219–239.

[21] A. A. Emerick and A. C. Reynolds, *Ensemble smoother with multiple data assimilation*, Computers and Geosciences, 55 (2013), pp. 3–15.

[22] G. Evensen, *Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics*, J. Geophys. Res., 99 (1994), pp. 10,143–10,162.

[23] G. Evensen, *Data Assimilation - The Ensemble Kalman Filter*, Springer, second ed., 2009.

[24] G. Evensen, *Data Assimilation: The Ensemble Kalman Filter*, Springer, 2nd ed., 2009.

[25] ——, *Analysis of iterative ensemble smoothers for solving inverse problems*, Computat Geosci, 22 (2018), pp. 885–908.

[26] G. Evensen, *Accounting for model errors in iterative ensemble smoothers*, Comput Geosci, 23 (2019), pp. 761–775.

[27] G. Evensen, P. Raanes, A. Stordal, and J. Hove, *Efficient implementation of an iterative ensemble smoother for data assimilation and reservoir history matching*, Frontiers in Applied Mathematics and Statistics, 5 (2019), p. 47.

[28] A. Fahimuddin, S. I. Aanonsen, and J.-A. Skjervheim, *4d seismic history matching of a real field case with enkf: Use of local analysis for model updating*, in SPE Annual Technical Conference and Exhibition, 2010. SPE 134894.

[29] G. Gaspari and S. E. Cohn, *Construction of correlation functions in two and three dimensions*, Quarterly Journal of the Royal Metereological Society, 125 (1999), pp. 723–757.

[30] F. Gassmann, *Über die elastizität poröser medien*, Viertel Naturforsch Ges Zürich, 96 (1951), pp. 1 – 23.

[31] T. Goda and K. Sato, *History matching with iterative latin hypercube samplings and parameterization of reservoir heterogeneity*, Journal of Petroleum Science and Engineering, 114 (2014), pp. 61 – 73.

[32] O. Gosselin, S. Aanonsen, I. Aavatsmark, A. Cominelli, R. Gonard, M. Kolasinski, F. Ferdinandi, L. Kovacic, and K. Neylon, *History matching using time-lapse seismic (HUTS)*, in SPE Annual Technical Conference and Exhibition, 2003. SPE-84464-MS.

[33] T. M. Hamill, J. S. Whitaker, and C. Snyder, *Distance-dependent filtering of background error covariance estimates in an ensemble kalman filter*, Monthly Weather Review, 129 (2001), pp. 2776–2790.

[34] J. Han and M. van der Baan, *Microseismic and seismic denoising via ensemble empirical mode decomposition and adaptive thresholding*, GEOPHYSICS, 80 (2015), pp. KS69–KS80.

[35] Z. Hashin and S. Shtrikman, *A variational approach to the theory of the elastic behaviour of multiphase materials*, Journal of the Mechanics and Physics of Solids, 11 (1963), pp. 127 – 140.

[36] M. B. Helgerud, A. C. Miller, D. H. Johnston, M. S. Udoh, B. G. Jardine, C. Harris, and N. Aubuchon, *4d in the deepwater gulf of mexico: Hoover, madison and marshall fields*, The Leading Edge, 30 (2011), pp. 1008–1018.

[37] G. Hennenfent and F. J. Herrmann, *Seismic denoising with nonuniformly sampled curvelets*, Computing in Science Engineering, 8 (2006), pp. 16–25.

[38] D. Johnston, *Practical applications of Time-lapse Seismic Data*, Society of Exploration Geophysicists (SEG), 2013.

[39] V. Kretz, B. Valles, and L. Sonneland, *Fluid front history matching using 4d seismic and streamline simulation*, in SPE Annual Technical Conference and Exhibition, Society of Petroleum Engineers, 2004. SPE-90136-MS.

[40] A. Latif and W. A. Mousa, *An efficient undersampled high-resolution radon transform for exploration seismic data processing*, IEEE Transactions on Geoscience and Remote Sensing, 55 (2017), pp. 1010–1024.

[41] O. Leeuwenburgh and R. Arts, *Distance parameterization for efficient seismic history matching with the ensemble kalman filter*, Comput Geosci, 18 (2014), pp. 535 – 548.

[42] E. Liu and B. Jafarpour, *Learning sparse geologic dictionaries from low-rank representations of facies connectivity for flow model calibration*, Water Resources Research, 49 (2013), pp. 7088–7101.

[43] M. Liu and D. Grana, *Time-lapse seismic history matching with iterative ensemble smoother and deep convolutional autoencoder*, GEOPHYSICS, 85 (2020), pp. M15–M31.

[44] R. Lorentzen, T. Bhakta, D. Grana, X. Luo, R. Valestrand, and G. Naevdal, *Simultaneous assimilation of production and seismic data: application to the norne field*, Comput Geosci, 24 (2020), pp. 907–920.

[45] R. Lorentzen, X. Luo, T. Bhakta, and R. Valestrand, *History matching the full Norne field model using seismic and production data*, SPE Journal, 24 (2019), pp. 1452 – 1467. SPE-194205-PA.

[46] X. Luo, *Ensemble-based kernel learning for a class of data assimilation problems with imperfect forward simulators*, PLOS ONE, 14 (2019), pp. 1–40.

[47] X. Luo and T. Bhakta, *Automatic and adaptive localization for ensemble-based history matching*, Journal of Petroleum Science and Engineering, (2019), p. 106559.

[48] X. Luo, T. Bhakta, M. Jakobsen, and G. Nævdal, *An ensemble 4D-seismic history-matching framework with sparse representation based on wavelet multiresolution analysis*, SPE Journal, 22 (2017), pp. 985 – 1010. SPE-180025-PA.

[49] X. Luo, T. Bhakta, M. Jakobsen, and G. Nævdal, *Efficient big data assimilation through sparse representation: A 3D benchmark case study in petroleum engineering*, PLOS ONE, 13 (2018), pp. 1–32.

[50] X. Luo, R. Lorentzen, R. Valestrand, and G. Evensen, *Correlation-Based Adaptive Localization for Ensemble-Based History Matching: Applied To the Norne Field Case Study*, SPE Reservoir Evaluation & Engineering, 22 (2019), pp. 1084–1109. SPE-191305-PA.

[51] X. Luo, A. Stordal, R. Lorentzen, and G. Nævdal, *Iterative ensemble smoother as an approximate solution to a regularized minimum-average-cost problem: theory and applications*, SPE Journal, 20 (2015), pp. 962–982. SPE-176023-PA.

[52] C. Maschio and D. J. Schiozer, *Probabilistic history matching using discrete Latin Hypercube sampling and nonparametric density estimation*, Journal of Petroleum Science and Engineering, 147 (2016), pp. 98–115.

[53] G. Mavko, T. Mukerji, and J. Dvorkin, *The Rock Physics Handbook: Tools for Seismic Analysis of Porous Media*, Cambridge University Press, 2 ed., 2009.

[54] R. D. Mindlin, *Compliance of elastic bodies in contact*, J. Appl. Mech., ASME, 16 (1949), pp. 259–268.

[55] G. Nævdal, T. Mannseth, and E. Vefring, *Near well reservoir monitoring through ensemble Kalman filter*, in SPE/DOE Improved Oil recovery Symposium, 2002. SPE-75235-MS.

[56] D. Obidegwu, R. Chassagne, and C. MacBeth, *Seismic assisted history matching using binary maps*, Journal of Natural Gas Science and Engineering, 42 (2017), pp. 69 – 84.

[57] D. S. Oliver and Y. Chen, *Recent progress on reservoir history matching: A review*, Comput Geosci, 15 (2011), pp. 185–221.

[58] L. Peters, R. Arts, G. Brouwer, C. Geel, S. Cullick, R. J. Lorentzen, Y. Chen, N. Dunlop, F. C. Vossepoel, R. Xu, P. Sarma, A. H. Alhuthali, and A. Reynolds, *Results of the Brugge Benchmark Study for Flooding Optimization and History Matching*, SPE Reservoir Evaluation & Engineering, (2010). SPE-119094-PA.

[59] P. N. Raanes, A. S. Stordal, and G. Evensen, *Revising the stochastic iterative ensemble smoother*, Nonlin. Processes Geophys, 26 (2019), pp. 325–338.

[60] S. Ravishankar and Y. Bresler, *Mr image reconstruction from highly undersampled k-space data by dictionary learning*, IEEE Transactions on Medical Imaging, 30 (2011), pp. 1028–1041.

[61] F. Roggero, O. Lerat, D. Ding, P. Berthet, C. Bordenave, F. Lefeuvre, and P. Perfetti, *History matching of production and 4d seismic data: Application to the girassol field, offshore angola*, Oil Gas Sci. Technol. - Rev. IFP Energies nouvelles, 67 (2012), pp. 237–262.

[62] R. Rubinstein, M. Zibulevsky, and M. Elad, *Efficient Implementation of the K-SVD Algorithm using Batch Orthogonal Matching Pursuit*, tech. rep., Computer Science Department, Technion - Israel Institute of Technology, 2008.

[63] P. Sakov and L. Bertino, *Relation between two common localization methods for the EnKF*, Computational Geosciences, 15 (2011), pp. 225–237.

[64] H. Shan, J. Ma, and H. Yang, *Comparisons of wavelets, contourlets and curvelets in seismic denoising*, Journal of Applied Geophysics, 69 (2009), pp. 103 – 115.

[65] J.-A. Skjervheim, G. Evensen, J. Hove, and J. Vabø, *An ensemble smoother for assisted history matching*, in SPE Reservoir Simulation Symposium, 2011. SPE-141929-MS.

[66] R. V. Soares, X. Luo, and G. Evensen, *Sparse Representation of 4D Seismic Signal Based on Dictionary Learning*, in SPE Norway One Day Seminar, 2019. SPE-195599-MS.

[67] R. V. Soares, C. Maschio, and D. J. Schiozer, *Applying a localization technique to kalman gain and assessing the influence on the variability of models in history matching*, Journal of Petroleum Science and Engineering, 169 (2018), pp. 110 – 125.

[68] P. Turquais, *Dictionary Learning and Sparse Representations for Denoising and Reconstruction of Marine Seismic Data*, PhD thesis, University of Oslo, Oslo, 2018.

[69] P. J. van Leeuwen and G. Evensen, *Data assimilation and inverse methods in terms of a probabilistic formulation*, Mon. Weather Rev., 124 (1996), pp. 2898–2913.

[70] M. Xiong, Y. Li, and N. Wu, *Random-noise attenuation for seismic data by local parallel radial-trace tfpf*, IEEE Transactions on Geoscience and Remote Sensing, 52 (2014), pp. 4025–4031.

[71] Q. Zhang and B. Li, *Discriminative k-svd for dictionary learning in face recognition*, in 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, June 2010, pp. 2691–2698.

[72] W. Zhu, S. M. Mousavi, and G. C. Beroza, *Seismic signal denoising and decomposition using deep neural networks*, IEEE Transactions on Geoscience and Remote Sensing, 57 (2019), pp. 9476–9488.

NORDIC ECOLABEL
241 731
PRINTED MATTER

Graphic design: Communication Division, UIB / Print: Skipnes Kommunikasjon AS

UNIVERSITAS BERGENSIS

uib.no

ISBN: 9788230855102 (print)
9788230860595 (PDF)