Delft University of Technology
Faculty of Electrical Engineering, Mathematics and Computer Science
Department of Applied Mathematics

Ensemble-Based History Matching for Channelized Petroleum Reservoirs

Thesis submitted to the
Delft Institute of Applied Mathematics
in partial fulfillment of the requirements
for the degree

MASTER OF SCIENCE
APPLIED MATHEMATICS

by

Matei Ţene
M.Tene@student.tudelft.nl

Delft, the Netherlands
20th September, 2013

Delft University of Technology

in collaboration with

TNO Petroleum Geosciences

Ensemble-Based History Matching for Channelized Petroleum Reservoirs

MSc THESIS APPLIED MATHEMATICS

Matei Ţene
M.Tene@student.tudelft.nl

20th September, 2013

Author
Matei Ţene — MSc Student, Risk and Environmental Modelling track
M.Tene@student.tudelft.nl — Department of Applied Probability
Delft University of Technology

Title
Ensemble-Based History Matching for Channelized Petroleum Reservoirs

Committee Members

Prof. Dr. Ir. A.W. Heemink *A.W.Heemink@tudelft.nl*	Promotor Department of Mathematical Physics Delft University of Technology
Dr. Ir. A.G. Chiţu *alin.chitu@tno.nl*	Daily Supervisor Medior Researcher Petroleum Geosciences TNO
Prof. Dr. Ir. J.D. Jansen *J.D.Jansen@tudelft.nl*	Head of Geoscience and Engineering Department Delft University of Technology
Dr. Ir. M.B. van Gijzen *M.B.vanGijzen@tudelft.nl*	Associate Professor Department of Numerical Analysis Delft University of Technology
MSc. Ir. C.M. Mariş *C.M.Maris@tudelft.nl*	PhD Candidate Department of Mathematical Physics Delft University of Technology

To my aunt, Maria Lateş (1946 - 2012)

Abstract

The oil industry is at the backbone of global economy and, as natural resources are becoming scarce, there is a pressing need for efficient extraction strategies. This has led to the development of reservoir models and simulators, able to predict future field behaviour, when paired with accurate geological information. However, the data obtained through in-situ measurements, such as seismic surveys, is insufficient to represent the large number of unknowns (porosity, permeability, pressure and fluid saturation in each grid cell).

In response to this issue, the scientific community designed computer-assisted history matching algorithms, which are able to provide estimates for model parameters by conditioning on the log of observed production data. The Ensemble Kalman Filter, in particular, is becoming the industry standard, because of its ease of implementation and natural ability to handle uncertainty. However, as past studies have pointed out, reservoirs with complex structural features, such as curved or branching channels, raise difficulties because of the higher-order dependencies induced between the state variables. Another important drawback is the appearance of ensemble collapse, which leads to poor estimates and causes the filter to diverge.

The Subspace EnKF is a recently developed history matching framework, able to address both of these issues, by using parameterizations to constrain the ensemble members to different subregions of the parameter space. The main goal of the Master's project was to adapt this framework to 2D channelized petroleum reservoirs, composed of two types of rocks (permeable sand and background shale).

For this purpose, we studied polynomial kernel Principal Components Analysis and proposed a novel analytical solution to the preimage problem. The experiments showed that our method surpasses the fixed-point iterative scheme, suggested in the literature, especially in terms of scalability and computational expense. Next, we paired the resulting KPCA parameterization with the Iterative Ensemble Smoother and the Subspace EnKF. Our comparative history matching experiment revealed that the latter is able to successfully avoid ensemble collapse. Finally, we suggested training set clustering as a means to accommodate the subspace parameterizations to the prior information and conducted a sensitivity study on the Subspace EnKF, which yielded encouraging results.

Contents

Figures

Tables

Mathematical statements

Algorithms

Notation

a, b, c, ..., α, β, ...	scalars
$\mathbf{x}$, $\mathbf{y}$, $\mathbf{z}$, ..., $\boldsymbol{\mu}$, $\boldsymbol{\lambda}$, ...	column vectors
$\mathbf{x}^T$, $\mathbf{y}^T$, $\mathbf{z}^T$, ..., $\boldsymbol{\mu}^T$, $\boldsymbol{\lambda}^T$, ...	row vectors
$\mathbf{X}$, $\mathbf{Y}$, $\mathbf{Z}$, ...	random vectors
$E(\mathbf{X}[i])$	the expected value of random variable $\mathbf{X}[i]$
A, B, C, ..., Σ, Λ, ...	matrices
A^T, B^T, C^T, ..., Σ^T, Λ^T, ...	transposed matrices
$\mathcal{P}$, $\mathcal{S}$, $\mathcal{X}$, ...	sets
$\mathbf{x}[i]$	i-th element of vector $\mathbf{x}$
$A[i,j]$	element on the i-th row and j-th column of matrix A
$A[i,:]$	i-th row of matrix A
$A[:,j]$	j-th column of matrix A
$\det(A)$	determinant of matrix A
$\mathrm{tr}(A)$	trace of matrix A, i.e. the sum of its diagonal elements
$\|\mathbf{x}\|_2$	Euclidean norm of vector $\mathbf{x} \in \mathbb{R}^m$
$\|\mathbf{x}\|_F$	space-specific norm of vector $\mathbf{x} \in F$
$\langle \mathbf{x}, \mathbf{y} \rangle_2$	dot product of vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^m$
$\langle \mathbf{x}, \mathbf{y} \rangle_F$	space-specific inner product of vectors $\mathbf{x}, \mathbf{y} \in F$
$\binom{n}{k}$	binomial coefficient, n choose k, where $n, k \in \mathbb{N}$ and $k \leq n$
$n!$	factorial of $n \in \mathbb{N}$
δ_{ij}	Kronecker delta function, $\delta_{ij} = \begin{cases} 1, & \text{if } i = j \\ 0, & \text{otherwise} \end{cases}$
I_n	$n \times n$ identity matrix, i.e. $I_n[i,j] = \delta_{ij}$
$\mathbf{1_n}$	n-component vector with $\mathbf{1_n}[i] = \frac{1}{n}$
1_n	$n \times n$ matrix with $1_n[i,j] = \frac{1}{n}$
$\boldsymbol{\nabla}_{\mathbf{x}} h$	gradient vector, corresponding to the function $h : \mathbb{R}^m \to \mathbb{R}$, defined as $\boldsymbol{\nabla}_{\mathbf{x}} h\ [i] = \dfrac{\partial h}{\partial \mathbf{x}[i]}$
$J_{\mathbf{x}} \boldsymbol{\xi}$	jacobian matrix, corresponding to the function $g : \mathbb{R}^m \to \mathbb{R}^l$, $g(\mathbf{x}) = \boldsymbol{\xi}$, defined as $J_{\mathbf{x}} \boldsymbol{\xi}\ [i,j] = \dfrac{\partial \boldsymbol{\xi}[i]}{\partial \mathbf{x}[j]}$

m	dimension of the parameter space, i.e. size of the state vector, $\mathbf{x}$
p	dimension of the feature space, i.e. dimension of the image, $\Phi(\mathbf{x})$
l	number of principal components, i.e. size of the projection vector, $\boldsymbol{\xi}$
n_e	ensemble size
n_o	number of observations
n_t	number of kernel PCA training vectors
M	model
H	observation operator; size $n_o \times m$
$\mathbf{x}$	state vector; size m
$\mathbf{x}_{truth}$	vector representing the truth state; size m
$\mathbf{x}^f$, $\mathbf{x}^a$	forecasted and analyzed state vectors, respectively; size m
$\mathbf{x}_i$	i-th ensemble member, $i = 1, \ldots, n_e$; size m
$\overline{\mathbf{x}}$	ensemble mean; size m
$\boldsymbol{\epsilon}$	error vector; size m
P	error covariance matrix; size $m \times m$
P^f, P^a	forecasted and analyzed error covariance matrices, respectively; size $m \times m$
$\mathbf{d}_{meas}$	vector of observations measured from the field; size n_o
$\mathbf{d}_i$	i-th vector of observations predicted by the model, $\mathbf{d}_i = H\mathbf{x}_i$, $i = 1, \ldots, n_e$; size n_o
$\mathbf{p}_i$	i-th vector of perturbations, $i = 1, \ldots, n_e$; size n_o
$\boldsymbol{\Delta}\mathbf{d}_i$	i-th innovation vector, $i = 1, \ldots, n_e$; size n_o
$\mathbf{w}$	observation error vector; size n_o
R	observation error covariance matrix; size $n_o \times n_o$
r_i	standard deviation of the error in the i-th observation, $i = 1, \ldots, n_o$
K_{gain}	Kalman gain matrix; size $m \times n_o$ in $\mathbb{R}^m$ and $l \times n_o$ in $\mathbb{R}^l$
$\mathbf{t}_i$	i-th kernel PCA training vector, $i = 1, \ldots, n_t$; size m
F	feature space; dimension p
$\Phi(\mathbf{x})$	image of $\mathbf{x}$ in F; dimension p
$\overline{\Phi_{\mathbf{t}}}$	mean of the images of the training vectors; dimension p
$\mathfrak{K}(\mathbf{x}, \mathbf{y})$	kernel function, computing the inner product of the images of $\mathbf{x}$ and $\mathbf{y}$ in F
$f(s)$	alternative formulation of $\mathfrak{K}$ in terms of the dot product, $s = \langle \mathbf{x}, \mathbf{y} \rangle_2$
K	training kernel matrix, $K[i,j] = \mathfrak{K}(\mathbf{t}_i, \mathbf{t}_j)$; size $n_t \times n_t$
$\mathbf{v}_i$	i-th principal component, $i = 1, \ldots, l$; dimension p
λ_i	i-th eigenvalue, corresponding to $\mathbf{v}_i$, $i = 1, \ldots, l$
$\boldsymbol{\alpha}_i$	coefficient vector, relating $\mathbf{v}_i$ to the $\{\Phi(\mathbf{t}_j)\}_{j=1,\ldots,n_t}$, $i = 1, \ldots, l$; size n_t
$\boldsymbol{\gamma}$	coefficient vector, relating $\Phi(\mathbf{x})$ to the $\{\Phi(\mathbf{t}_j)\}_{j=1,\ldots,n_t}$; size n_t
$\boldsymbol{\xi}$	projection vector, obtained by projecting $\Phi(\mathbf{x})$ onto the $\{\mathbf{v}_i\}_{i=1,\ldots,l}$; size l
g	parameterization function, $g(\mathbf{x}) = \boldsymbol{\xi}$

Preface

This report is a summary of my research at the Delft Institute of Applied Mathematics between January and August 2013.

My first contact with *data assimilation* was in 2011 when I attended the summer school jointly organized by TU Delft and TNO in Iaşi, Romania. At the time, I was a freshly graduated Bachelor in Computer Science and my mathematical background was rather naive. Still, I discovered a community of people from different fields that were using the same set of simple equations to improve the prediction power of their models.

The mystery was dispelled during the master course on *Environmental Modelling and Data Assimilation* taught by Dr. Peter Wilders and Dr. Remus Hanea and I decided to do my graduation project on the Ensemble Kalman Filter. What followed was a two month internship under the supervision of Cristian Mariş, during which I learned about kernel PCA, a nonlinear parametrization developed in the artificial intelligence community. After that, I was welcomed at TNO Utrecht, in the Petroleum Geosciences department, where I familiarized myself with the geology of petroleum reservoirs. Now, after eight months, I'm happy to see that all the pieces, at first deemed incompatible, have somehow managed to come together.

I was very fortunate to be awarded a full scholarship from the Delft Institute of Applied Mathematics, as international student in the Risk and Environmental Modelling program. This grant was crucial for my stay in the Netherlands, allowing me to concentrate solely on the research topic. I am grateful to my daily supervisor, Dr. Alin Chiţu, for our long series of interesting talks during the daily drive to Utrecht and for his guidance through all the ups and downs of my project work. To Cristian Mariş for his explanations and review of my results. To Prof. Dr. Arnold Heemink for his continued interest in my progress and for consistently providing feedback. To Dr. Olwijn Leeuwenburgh for our discussions during my software implementation. To Dr. Remus Hanea for introducing me to the subject and to Prof. Dr. Dorota Kurowicka for her support throughout the entirety of my Master. I would also like to thank the members of my thesis committee for their patience in reading my report and for providing useful comments. And of course, my friends and family. My work would have been impossible without your love and support!

Matei Ţene
20^{th} September, 2013, Delft

Chapter 1

Introduction

For the last century, the oil industry has gradually consolidated its position at the backbone of world economy. As described by Halliday (2005), petroleum was "the largest commodity, in value terms, traded in the world market" across the 20th century, while Fried et al. (1975) argue that the increase in its price was the main cause for the global recession between 1973-1975.

However, as natural resources start to fall short and most of the oil fields around the world enter the "mature" stages of their life cycle, there is a pressing need for efficient extraction strategies. This has led to the development of computational tools, such as reservoir models and simulators, their results proving to be indispensable in decision-making, from the early phase of field prospection, to the later enhanced oil recovery (EOR) stage. However, since petroleum reservoirs are located at considerable depths in the subsurface, in-situ studies, such as seismic surveys, can only provide a limited amount of a priori information about the oil-bearing geological structures. Hence, there is a high degree of uncertainty regarding the values that need to be supplied as input to the models, in order to obtain results which reflect real field operation. On the bright side, after production begins, there is a constant inflow of data from the so-called *smart wells*, and an ideal scenario would be to implement a closed-loop reservoir management system (Jansen et al. 2005), in which model updates are performed in in real-time.

History matching is a crucial component in this loop, its purpose being to adjust the parameters by conditioning on the log of measured data. Ensemble-based algorithms, in particular, have recently gained popularity in the scientific community, due to their ease of implementation and natural ability to handle uncertainty (Oliver and Y. Chen 2011). However, their theoretical formulation relies on a set of assumptions, and previous studies (see, for example, Sarma and W. Chen 2009; or Ma and Zabaras 2011) showed that reservoirs with complex geological structure (curved features or branching channels) are particularly challenging.

This leads to the following

1.1. Problem statement

Given a joint distribution, which incorporates all the available prior information about the geology of a channelized petroleum reservoir, construct a posterior distribution by conditioning on the history of observed production data, such that the following requirements are fulfilled:

- The estimate provided by the posterior distribution also has a channelized structure.
- Its forecasted production data is in line with the observations.
- And, finally, the uncertainty associated with the posterior model parameters is represented appropriately.

1.2. Research goals

The Subspace Kalman Filter is a history matching framework, recently developed by Sarma and W. Chen (2013) to address the challenges mentioned above. The purpose of the present project is to provide an appropriate formulation and determine the advantages of using the Subspace EnKF for history matching applications. To this end, we focus, specifically, on 2D channelized reservoirs, composed of two types of rocks (channel sand and background shale). In order to reach our end-goal, we will investigate the following topics:

① Propose a parameterization that preserves the structure of channelized reservoirs over sequential assimilation steps.

② Study the effect of the number of subspaces when using the Subspace EnKF for history matching channelized reservoirs.

③ Develop a strategy to form the subspaces which takes into account the prior information about the reservoir.

The report is structured as follows. First, we introduce the basic geological notions concerning petroleum reservoirs, in general, and investigate the flow and well production for channelized fields, in particular. Next, we turn to multi-point geostatistics as a means to represent the prior knowledge and generate the initial ensemble for the Ensemble Kalman Filter data assimilation algorithm. After formulating a suitable state-space representation, we investigate the problem of adaptating the (continuous) response of the EnKF to represent binary categorical variables.

Chapter 3 is dedicated to the kernel PCA parameterization, as a means to address the limitation of the EnKF in handling higher-order moments between state variables. Here, we propose an analytical solution to the preimage problem, as alternative to the fixed-point iterative scheme suggested in the literature.

In chapter 4, we investigate the Iterative Ensemble Smoother and the Subspace EnKF in an attempt to avoid ensemble collapse. Further insight is obtained during the experiments in chapter 5, which is divided in three parts. The first is a study on the KPCA parameterization, outlining the differences between the two preimage problem solutions and the effect of the size of the training set and reservoir grid on their performance. Thereafter, we conduct a comparative history matching experiment on two channelized reservoirs, and finally, we discuss the results of our sensitivity study on the Subspace EnKF, meant to provide answers to the last two research topics.

Chapter 2

Petroleum reservoirs and history matching

As depicted in figure 2.1, a reservoir is a subsurface body of porous rock. However, the factor that makes it noteworthy to multinational oil companies, such as Shell, Chevron or Statoil, is the mass of crude oil and natural gas residing in its pores. These hydrocarbons result from the decomposition of large quantities of biomass during millions of years and at the right levels of temperature and pressure. And since simple autotrophic organisms, such as phytoplankton, are known to have periodic large-scale outbursts, it is not surprising that most reservoirs are formed around ancient sea and river beds.

However, there is one more crucial ingredient that leads to the formation of a petroleum reservoir – the impermeable *cap rock*, which prevents the hydrocarbons from syphoning towards the surface. This creates a so-called *petroleum trap* (see figure 2.1), in which the target fluids are sandwiched between the upper impermeable barrier and the lower, water-bearing rock (the aquifer).

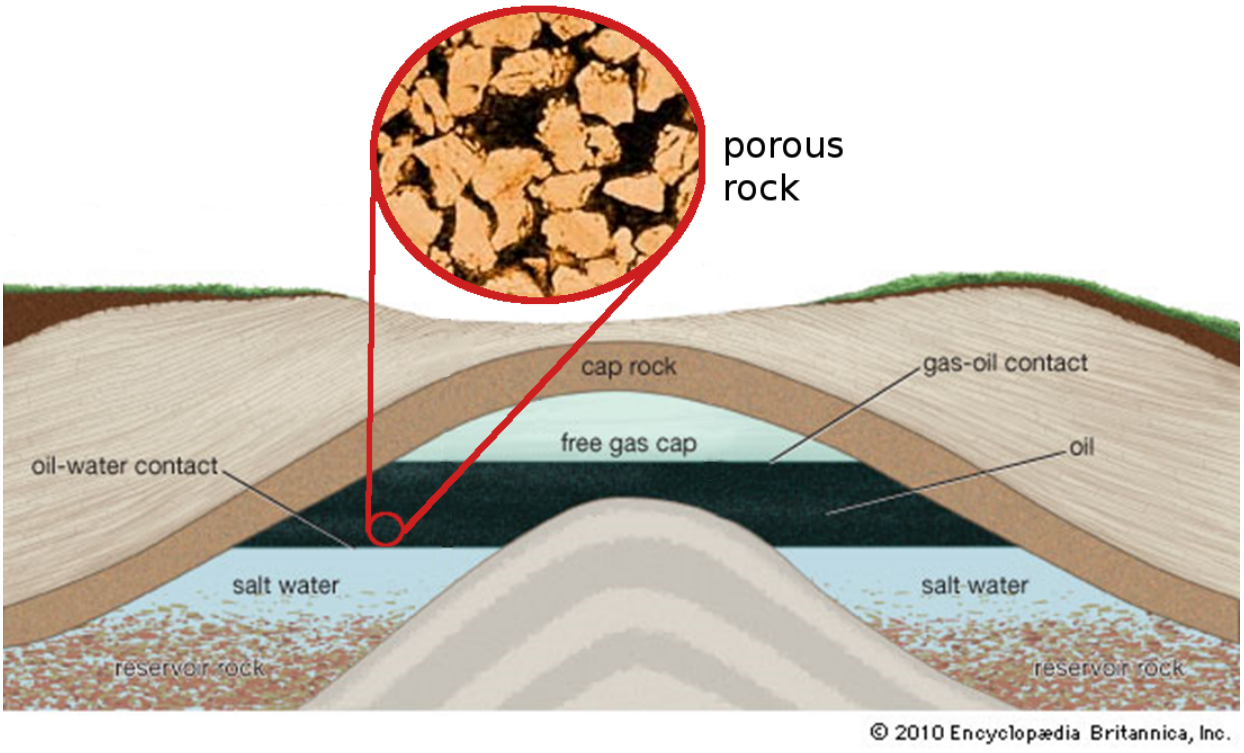


Figure 2.1: Petroleum reservoir [a]

[a]Reservoir image courtesy of *Encylopaedia Britannica*, in accordance with section *1. Terms of Use for Everyone* subsection *Use of Content* of the terms at http://corporate.britannica.com/termsofuse.html, accessed on *22 July 2013*.

2.1. Field production and geology

The fluids trapped within the reservoir are, initially, in thermodynamic equilibrium, and their extraction can be performed by drilling wells that perforate the cap rock. Then, the difference in pressure between the atmosphere (1 atm) and subsurface (hundreds of atm) will induce a driving force which pushes the fluids towards the surface. This is called *primary production* and is depicted in figure 2.2.

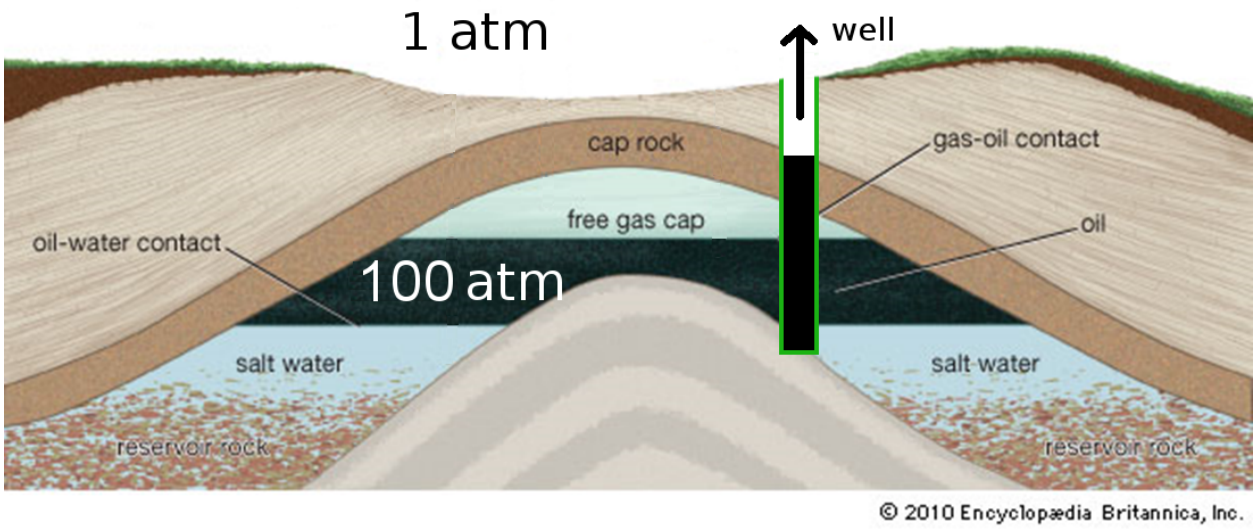


Figure 2.2: Primary production phase

However, according to the laws of physics. the pressure difference will inevitably drop with time and field throughput will start to decline. This is the beginning of *secondary production* (figure 2.3), during which some of the wells are used to inject water into the aquifer, maintaining the pressure gradient. This phase is also known as water-flooding and constitutes the main focus in our application.

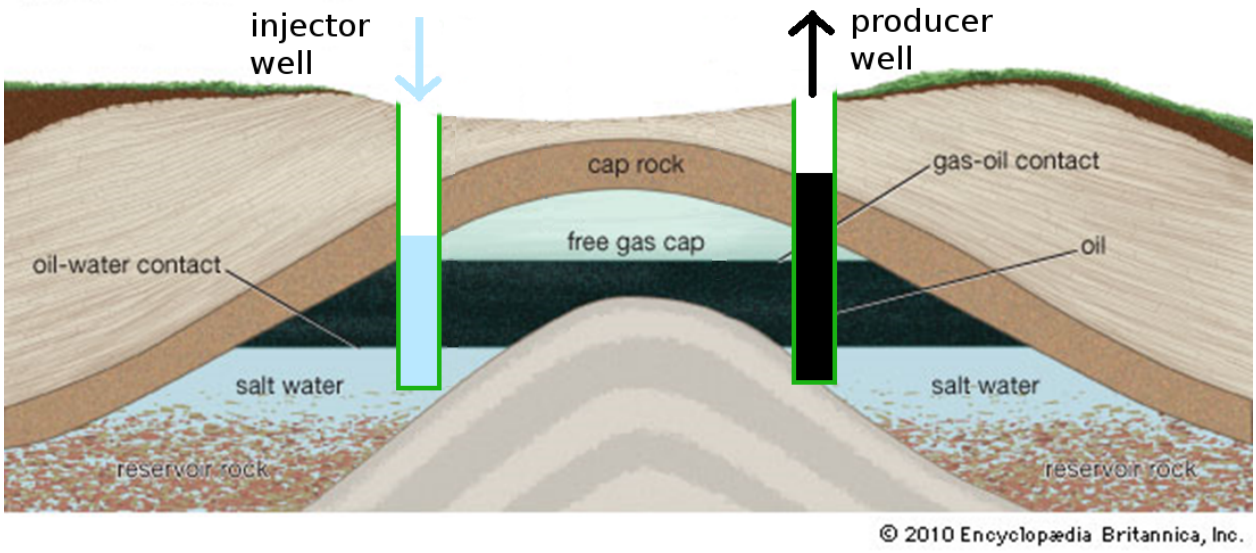


Figure 2.3: Secondary production phase

The behaviour of subsurface fluid flow is determined by two important geological properties. On the one hand, there is *porosity*, which measures the percentage of the rock volume occupied by pores. As such, its value gives an indication of the quantity of hydrocarbons we can expect to produce from the field. On the other hand, we have *permeability*, which is proportional to the amount of effort needed for a fluid to traverse the body of rock. Its value is related to the way the

pores are connected to each other and is the main factor in determining the direction and intensity of the flow.

Our study is focused on a special class of petroleum reservoir, which formed around ancient river beds. As such, their structure exhibits curved channels, which branch out and constitute the main pathways for fluid flow (see figure 2.4 for an example). For simplicity, we assume that our reservoirs are representable in 2D and their composition only has two types of rocks (or *facies*) with known properties (table 2.1): channel sand, which is quite permeable, and background shale, which is essentially impermeable.

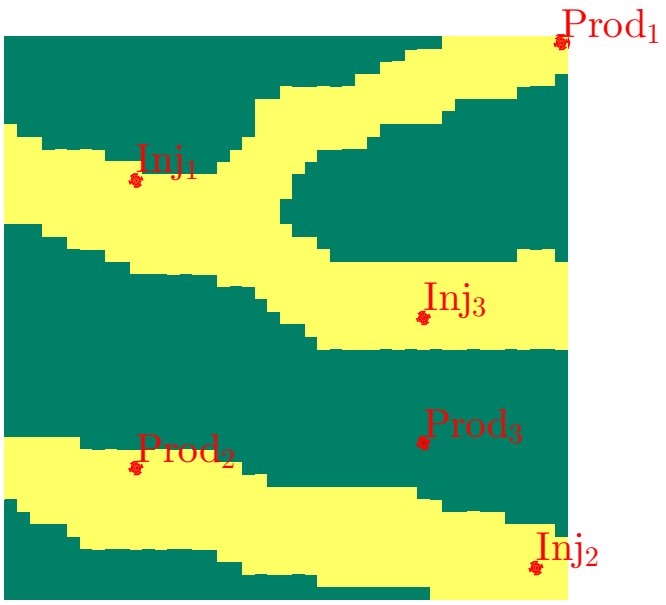


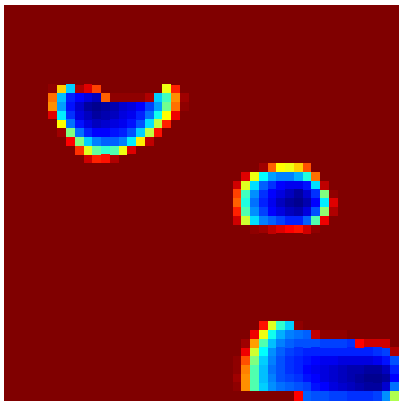
Figure 2.4: *Y-channel reservoir* with well locations

Rock type	Permeability	Porosity
Background shale	0.1 mD	5%
Channel sand	100 mD	20%

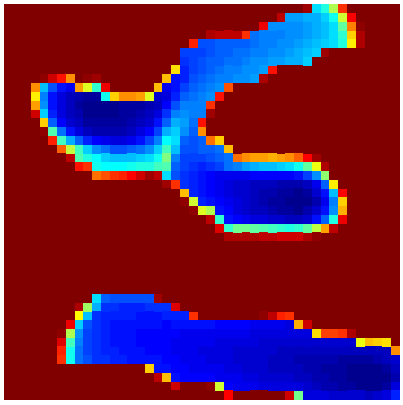
Table 2.1: Geological properties of the two facies that form channelized reservoirs

In order to get a feel for the flow pattern in channelized reservoirs, we will use the MRST reservoir simulator (SINTEF 2013) to monitor the first four years in the production cycle of the *Y-channel reservoir* from figure 2.4. The results are shown in figure 2.5, and, by examining the oil saturation plots (figures 2.5a and 2.5b), we confirm that the flow is, indeed, directed along the channels.

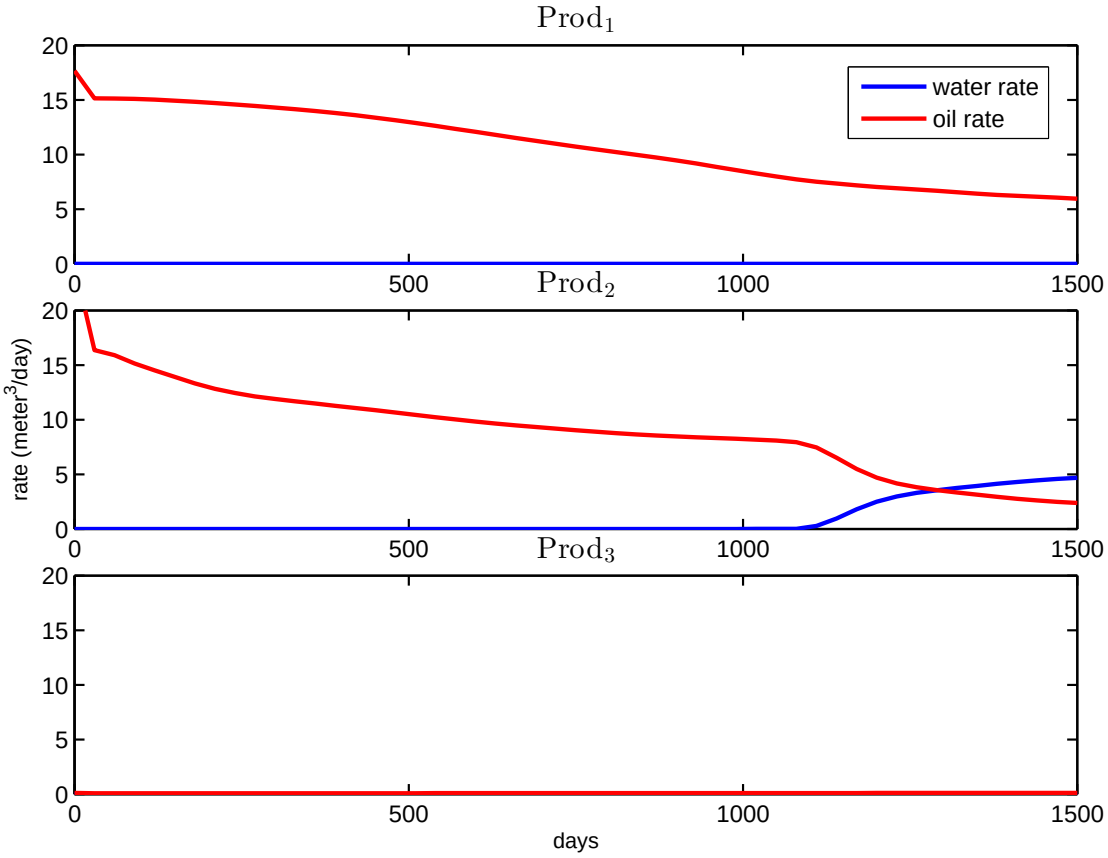
Figure 2.5c shows the production rates in different wells and, first off, we see that $Prod_3$ is essentially inactive due to its unfortunate position in the impermeable background. In $Prod_2$, we notice the intersection of the two curves at ~ 1250 days. This important moment in the production cycle is known as the *water breakthrough*, and it appears when the injected *water front* (figure 2.5b) reaches the location of the well. Our simulation is not long enough to capture the same behaviour in $Prod_1$, whose evolution seems to be slowed down by the two "competing" injectors, Inj_1 and Inj_3.



(a) Oil saturation, 360 days



(b) Oil saturation, 1500 days



(c) Production data

Figure 2.5: Evolution of the *Y-channel reservoir*

2.2. Geostatistics

The main problem in predicting the future production behaviour of a petroleum reservoir, lies in the fact that its geological structure is unknown a priori. The information obtained from field measurements (for example, seismic studies) allows us to get a general idea, but still a high degree of uncertainty remains regarding the permeability and porosity values.

Geostatistics is a mathematical framework, designed to quantify the representation uncertainty of the prior geological information. In order to achieve this, geostatistical methods operate by building a joint probability distribution, which captures the relationships between the values of the rock properties at different locations within the reservoir. An important by-product is that this distribution can be *sampled*, in order to obtain and study different structural scenarios for the field in question. Moreover, the samples can be constrained to satisfy *hard data*, for example, the type of rock at a particular location.

The traditional geostatistical approach is to use a *variogram*, which gives the dependency between a value at a given point and its neighbours. Unfortunately, in order to represent curved channels, we need to take into account groups of more the 2 points, thus, for our application, we need to turn towards *multi-point geostatistics*. The difference is that, instead of considering the values at single locations, multi-point methods build a distribution for structural *features*.

Snesim is a popular multi-point algorithm, introduced by Strebelle (2002), which operates on *training images* (figure 2.6). These images do not (necessarily) depict real reservoirs; their role, however, is to represent all the possible structural features, along with their frequency of appearance. Broadly speaking, snesim extracts patches from the training image, according to a predefined "search ellipsoid", and uses them to build the corresponding geostatistical distribution.

For our purposes, we will run snesim on the channelized reservoir training image (figure 2.6), in order to obtain ensembles of geological scenarios, to be used in history matching algorithms – for clarity, a diagram of the sampling workflow is provided in figure 2.7.

2.3. History matching

History matching, or *data assimilation*, is the procedure of updating the parameters of a model by conditioning on a time-series of observations. The main motivation in doing this is the hope that the posterior model is more accurate in predicting future behaviour of the studied phenomenon (in our case, the production of a channelized petroleum field during water-flooding).

We begin this section by introducing the mathematical framework necessary to prepare a model for history matching. Next, we focus on the Ensemble Kalman Filter algorithm, the numerical aspects related to its implementation and possible approaches in maintaining consistency in the results. Finally, we study its performance on the *Y-channel reservoir* (figure 2.4).

2.3.1. State space representation

There are three main aspects to consider in describing the physical state of a petroleum reservoir:

- the geological structure, given by the permeability and porosity in each point (unknown a priori and, for our purposes, we assume that these properties remain unchanged in time).
- the flow characteristics, captured in the values of the pressure and fluid saturation in each point (practically impossible to measure).
- the production data, i.e. the bottom-hole pressures (BHP) and fluid rates in each well (which constitute our observations).

Given the size of a reservoir and the volume of computations necessary to propagate its state forward in time, we need to define a spatial discretization grid and rely on numerical tools, such

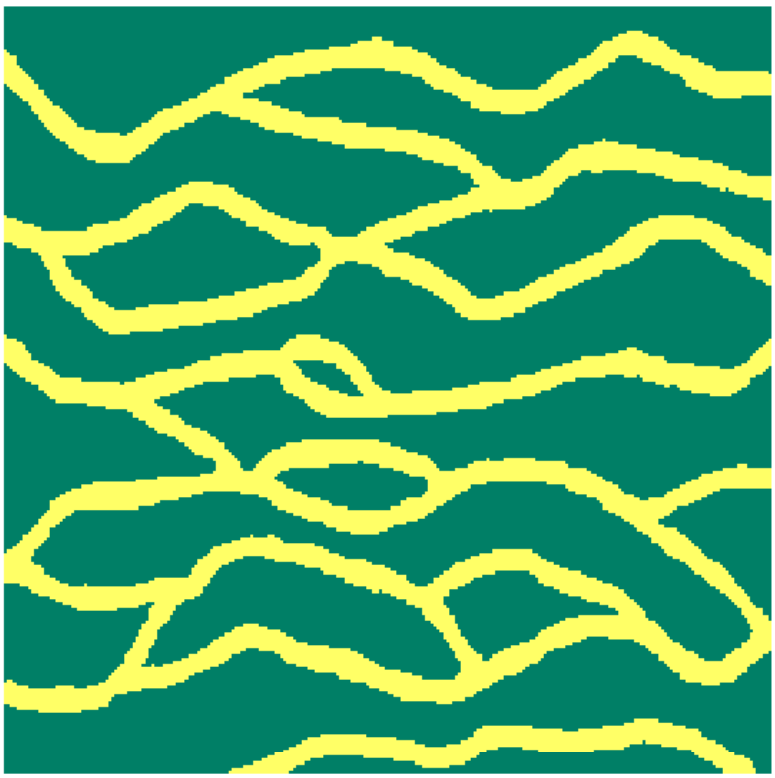


Figure 2.6: Training image

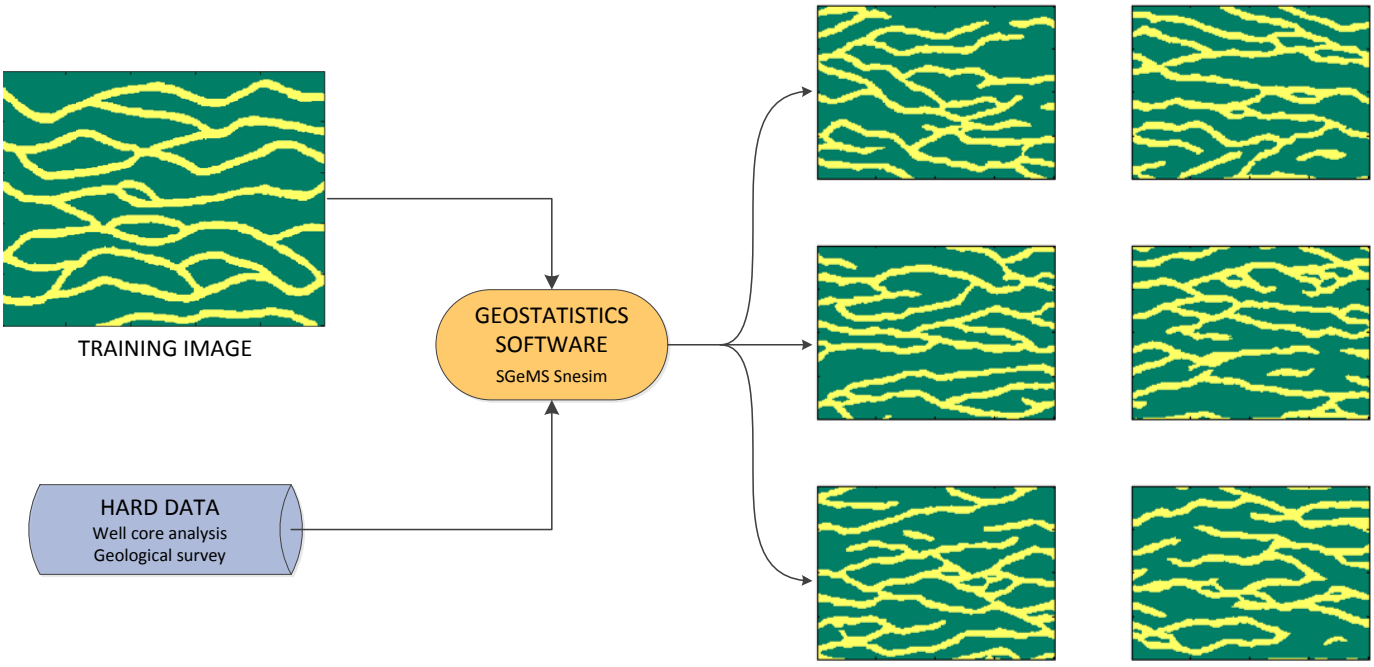


Figure 2.7: Multipoint geostatistics sampling

as the MRST reservoir simulator (SINTEF 2013). To that end, we introduce the so-called *state vector*, which gathers the values of the quantities listed above, at a given time, t,

$$\mathbf{x}^{(t)} = \begin{bmatrix} perm_i \\ poro_i \\ pres_i \\ sat_i \\ rate_j \\ bhp_j \end{bmatrix} \qquad \begin{aligned} i &= 1, \ldots, n_{\text{grid cells}} \\ j &= 1, \ldots, n_{\text{wells}} \end{aligned} \tag{2.3.1}$$

Mathematically, $\mathbf{x}$ is a member of an m-dimensional space, to which we will refer, simply, as the *parameter space*. The number of state variables, m, for a realistic field is, typically, of order 10^6.

With this in mind and if we assume that the reservoir simulator is able to capture reality perfectly, then the history matching problem reduces to that of finding the initial condition, $\mathbf{x}^{(0)}$, such that the forecasted production behaviour matches the observed data. The main uncertainty related to $\mathbf{x}^{(0)}$ lies in the grid cell permeability and porosity values, and we will attempt to estimate them using the algorithm presented in the next paragraph.

However, before moving on, we would like note that, for channelized reservoirs, the permeability and porosity values are fully determined by the type of rock (sand or shale) in the respective grid cell. Hence, we can reduce the size of the state vector by replacing the geological properties with a categorical variable, which gives the type of facies. For example, we can plug the output of snesim (0 for shale and 1 for sand) directly into the state vector; then (2.3.1) becomes

$$\mathbf{x}^{(t)} = \begin{bmatrix} facies_i \\ pres_i \\ sat_i \\ rate_j \\ bhp_j \end{bmatrix} \qquad \begin{aligned} i &= 1, \ldots, n_{\text{grid cells}} \\ j &= 1, \ldots, n_{\text{wells}} \\ facies_i &\in \{0, 1\} \end{aligned} \tag{2.3.2}$$

2.3.2. The Ensemble Kalman filter

We intend to use data assimilation to tune the parameters of our reservoir model, increasing its power to predict the true state, $\mathbf{x}_{truth}$, of channelized petroleum reservoirs. Among the algorithms designed for this purpose, the Kalman Filter (KF), introduced by Kalman (1960), has recently gained popularity in the literature (Oliver and Y. Chen 2011), especially due to its intrinsic ability to handle uncertainty.

Given a distribution, f^{prior}, that incorporates all the available information about the state vector at moment t, and a model, M, that describes how the state changes in time, the aim of the KF is to compute an updated, $f^{posterior}$, by conditioning on the set of observations. The algorithm is sequential and operates in two steps. The first is called the *forecast step*, in which the model is used to propagate the state forward in time,

$$\mathbf{x}^{(t)} = M(\mathbf{x}^{(t-\Delta t)}) \qquad \mathbf{x}^{(t)} \sim f^{prior} \tag{2.3.3}$$

For simplicity, we will drop the time indices and refer to $\mathbf{x}^{(t)}$ as $\mathbf{x}^f$.

The observations are field measurements which provide information about the true state, and are given by

$$\mathbf{d}_{meas} = H\mathbf{x}_{truth} + \mathbf{w} \qquad \mathbf{d}_{meas}, \mathbf{w} \in \mathbb{R}^{n_o} \quad \mathbf{w} \sim \mathbf{W} \tag{2.3.4}$$

Here, H is the observation operator that describes the relationship (assumed linear) between the measured quantities and the state variables in $\mathbf{x}$. The above also accounts for observation error; $\mathbf{w}$ is a realization of a random vector, $\mathbf{W}$, whose distribution describes our uncertainty about the accuracy of $\mathbf{d}_{meas}$.

If observations are available at time t, then we compute the *innovation vector*,

$$\mathbf{\Delta d} = \mathbf{d}_{meas} - H\mathbf{x} \tag{2.3.5}$$

which shows the discrepancy between the observed values and those predicted by the model.

We are now ready to enter the *analysis step*, which updates the state by conditioning on the observations,

$$\mathbf{x}^a = \mathbf{x}^f + K_{gain}\mathbf{\Delta d} \qquad\qquad \mathbf{x}^a \sim f^{posterior} \tag{2.3.6}$$

where K_{gain} is an $m \times n_o$ matrix of coefficients, called the *Kalman gain*. Therefore, we see that the update is linear and its impact depends on the magnitude of the innovation vector – a perfect match with the truth will leave the state unchanged, while a $\mathbf{\Delta d}$ with large magnitude will translate into a heavily modified $\mathbf{x}$.

The Kalman gain is chosen as to minimize the *mean squared error* function,

$$\begin{aligned} &MSE : \mathbb{R}^m \to \mathbb{R} \\ &MSE(\mathbf{x}) = \frac{1}{m}\sum_{i=1}^{m}(\mathbf{x}[i] - \mathbf{x}_{truth}[i])^2 \end{aligned} \tag{2.3.7}$$

Thus, it is the solution to the following optimization problem,

$$\min_{K_{gain}} MSE(\mathbf{x}^a) \tag{2.3.8}$$

which can be equivalently formulated as (Kalman 1960)

$$\min_{K_{gain}} \operatorname{tr}(P^a) \tag{2.3.9}$$

where P is the *error covariance*,

$$P = E\left[\boldsymbol{\epsilon}\boldsymbol{\epsilon}^T\right] \tag{2.3.10}$$

and $\boldsymbol{\epsilon}$ is the *error vector*,

$$\boldsymbol{\epsilon} = \mathbf{x}_{truth} - \mathbf{x} \tag{2.3.11}$$

The derivation is straightforward,

$$\begin{aligned} \boldsymbol{\epsilon}^a &= \mathbf{x}_{truth} - \mathbf{x}^a \\ &\overset{(2.3.6)}{=\!=} \mathbf{x}_{truth} - \mathbf{x}^f - K_{gain}(\mathbf{d}_{meas} - H\mathbf{x}^f) \\ &\overset{(2.3.4)}{=\!=} \mathbf{x}_{truth} - \mathbf{x}^f - K_{gain}(H\mathbf{x}_{truth} + \mathbf{w} - H\mathbf{x}^f) \\ &= (I - K_{gain}H)\boldsymbol{\epsilon}^f - K_{gain}\mathbf{w} \end{aligned} \tag{2.3.12}$$

If we denote by $R = E\left(\mathbf{w}\mathbf{w}^T\right)$, the observation error covariance, then we have

$$\begin{aligned} P^a &= (I - K_{gain}H)P^f(I - K_{gain}H)^T + K_{gain}R{K_{gain}}^T \\ &= P^f - K_{gain}HP^f - P^fH^T{K_{gain}}^T + K_{gain}(HP^fH^T + R){K_{gain}}^T \end{aligned} \tag{2.3.13}$$

where we assumed that the error vector and the observation error are uncorrelated (Kalman 1960).

The solution to (2.3.9) can now be found by setting the gradient of the objective function to $\mathbf{0}$,

$$\frac{d\operatorname{tr}(P^a)}{dK_{gain}} = -2 \cdot (HP^f)^T + 2 \cdot K_{gain}(HP^fH^T + R) = \mathbf{0} \tag{2.3.14}$$

finally, leading to

$$K_{gain} = P^f H^T (H P^f H^T + R)^{-1} \tag{2.3.15}$$

The Ensemble Kalman Filter (EnKF) is a Monte-Carlo variant of the Kalman filter, introduced by Evensen (1994). In this formulation, the prior and posterior distributions are represented by a set of n_e realizations, $\mathbf{x}_1, \ldots, \mathbf{x}_{n_e} \in \mathbb{R}^m$, referred to as *ensemble members*. The *ensemble mean* is, then, an estimation for the true state,

$$\mathbf{x}_{truth} \simeq \overline{\mathbf{x}} = \frac{1}{n_e} \sum_{i=1}^{n_e} \mathbf{x}_i \tag{2.3.16}$$

while the *ensemble covariance* is an approximation for P,

$$P \simeq \frac{1}{n_e - 1} \sum_{i=1}^{n_e} (\mathbf{x}_i - \overline{\mathbf{x}})(\mathbf{x}_i - \overline{\mathbf{x}})^T \tag{2.3.17}$$

The EnKF workflow, illustrated in figure 2.8, uses the same Kalman update equation (2.3.6), this time applied to each forecasted ensemble member, ${\mathbf{x}_i}^f$. The only difference is a correction to the innovation vector (equation (2.3.5)), introduced by Burgers et al. (1998),

$$\mathbf{\Delta d}_i = \mathbf{d}_{meas} + \mathbf{p}_i - H\mathbf{x}_i \tag{2.3.18}$$

Here, the $\mathbf{p}_i$ are an ensemble of perturbations, sampled from $\mathbf{W}$, and can be interpreted as a simulation of the field measurement process.

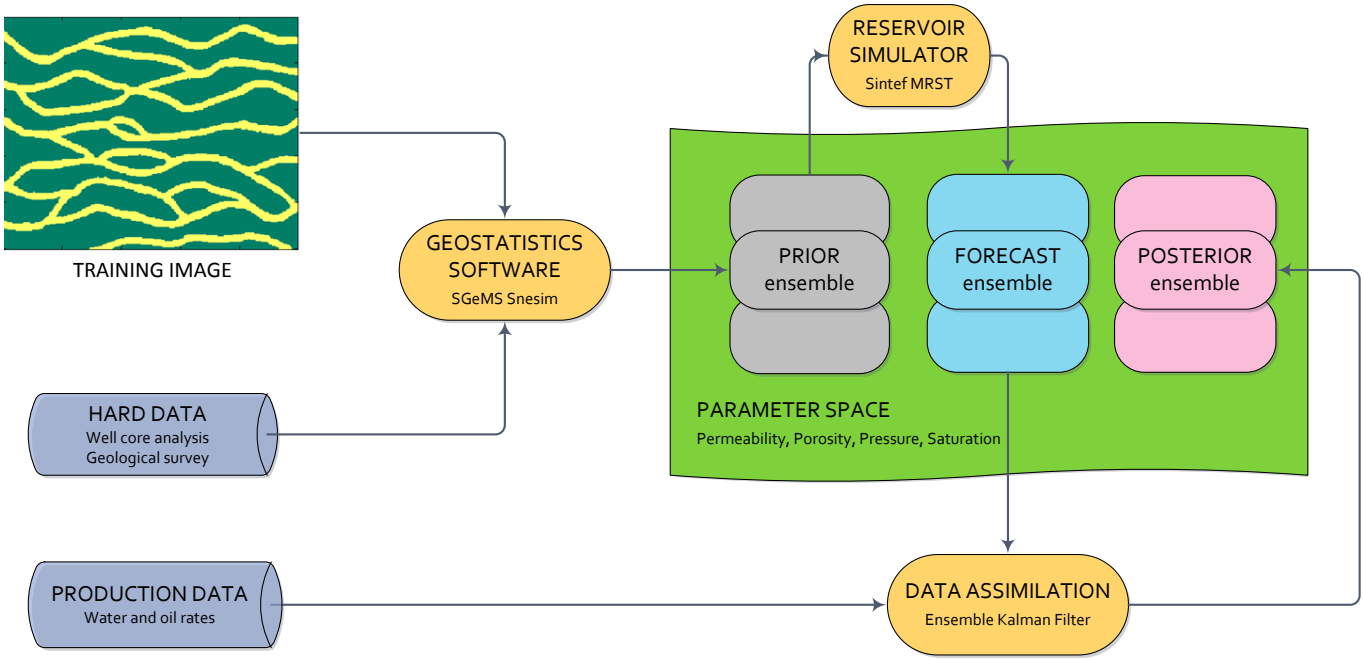


Figure 2.8: EnKF workflow

The derivation of the Kalman filter equations and the performance of the EnKF rely on a set of assumptions. Some of them were already explicitly stated, while the rest is given in (Kalman 1960; and Evensen 1994). We compile a complete list below:

① The distribution of the state vector, $\mathbf{x}$, can be fully described by its mean, $\overline{\mathbf{x}}$, and covariance matrix, P, i.e. f^{prior} and $f^{posterior}$ are multivariate Gaussian.

② The observation operator H is a linear function of the state vector. This can be satisfied by *augmenting* the $\mathbf{x}$ with the model-predicted observations, $\mathbf{d}_i = H\mathbf{x}_i$ (Evensen 2003).

③ $\mathbf{W}$, the distribution of the observation error, is multivariate Gaussian with zero mean and covariance matrix R, and the ensemble of perturbations, $\{\mathbf{p}_i\}_{i=1,\dots,n_e}$, is mean-centered. For our application, we will also assume that $R = diag({r_1}^2, \dots, {r_{n_o}}^2)$.

④ The error vector, $\boldsymbol{\epsilon}$, and the observation error vector, $\mathbf{w}$, are uncorrelated, i.e.

$$E\left(\boldsymbol{\epsilon}\mathbf{w}^T\right) = E\left(\mathbf{w}\boldsymbol{\epsilon}^T\right) = \mathbf{0} \tag{2.3.19}$$

2.3.3. Numerical aspects of the EnKF

Since the numerical implementation is easier if we write the equations in matrix form, we define the following

$$\begin{aligned} X[:,i] &= \mathbf{x}_i \\ D[:,i] &= \mathbf{d}_i = H\mathbf{x}_i \end{aligned} \qquad \forall i = 1, \dots, n_e \tag{2.3.20}$$

Then,

$$\begin{aligned} HX^f &= D \\ H\widetilde{X^f} &= HX^f(I_{n_e} - 1_{n_e}) = \widetilde{D} \end{aligned} \tag{2.3.21}$$

which allows us to rewrite (2.3.15) as

$$\begin{aligned} K_{gain} &\overset{(2.3.17)}{=\!=} \frac{1}{n_e - 1}\widetilde{X^f}\widetilde{X^f}^T H^T \left(\frac{1}{n_e - 1}H\widetilde{X^f}\widetilde{X^f}^T H^T + R\right)^{-1} \\ &= \widetilde{X^f}\widetilde{D}^T \left(\widetilde{D}\widetilde{D}^T + (n_e - 1)R\right)^{-1} \end{aligned} \tag{2.3.22}$$

Also, notice that,

$$\widetilde{X^f}\widetilde{D}^T = X^f(I_{n_e} - 1_{n_e})\widetilde{D}^T = X^f\left(\widetilde{D}(I_{n_e} - 1_{n_e})\right)^T = X^f\widetilde{D}^T \tag{2.3.23}$$

since $\widetilde{D}1_{n_e} = 0$.

If we substitute the above into the Kalman update equation (2.3.6),

$$\begin{aligned} X^a &= X^f + K_{gain}\Delta D \\ &\underset{(2.3.23)}{\overset{(2.3.22)}{=\!=}} X^f + X^f\widetilde{D}^T \left(\widetilde{D}\widetilde{D}^T + (n_e - 1)R\right)^{-1}\Delta D \\ &= X^f\left[I_{n_e} + \widetilde{D}^T \left(\widetilde{D}\widetilde{D}^T + (n_e - 1)R\right)^{-1}\Delta D\right] \end{aligned} \tag{2.3.24}$$

we see that the ${\mathbf{x}_i}^a$, are, in fact, linear combinations of the ${\mathbf{x}_i}^f$ with coefficients taken from the columns of the $n_e \times n_e$ matrix

$$C = I_{n_e} + \widetilde{D}^T \left(\widetilde{D}\widetilde{D}^T + (n_e - 1)R\right)^{-1}\Delta D \tag{2.3.25}$$

According to Evensen (2003), the sum along every column of C is equal to 1, while the sum along the rows gives each member's relative "importance" in the assimilation result. An important thing to note is that C does not use the ensemble, X, directly, which allows us to shorten the state vector by removing the quantities we are not actually interested to update. This includes the production data, which can be obtained directly from the model, and even the flow variables, since we plan to use restarting (see next paragraph), leaving only

$$\begin{aligned} &\mathbf{x}[i] = facies_i \\ &\qquad\qquad \forall i = 1, \dots, n_{\text{grid cells}} \\ &\qquad\qquad facies_i \in \{0, 1\} \end{aligned} \tag{2.3.26}$$

The most demanding step in the EnKF algorithm is computing the inverse of

$$B = \tilde{D}\tilde{D}^T + (n_e - 1)R \tag{2.3.27}$$

which is an $n_o \times n_o$ matrix. By analyzing its structure, we notice that the main contribution comes from $\tilde{D}\tilde{D}^T$, whose rank is $\min(n_o, n_e)$. Therefore, it might happen that B is rank deficient and we have to resort to a pseudo-inverse.

We distinguish two cases, and, in treating them, it is useful to first decompose

$$(n_e - 1)R = QQ^T \tag{2.3.28}$$

and, since R was chosen diagonal,

$$Q = \sqrt{n_e - 1} \cdot diag(r_1, \ldots, r_{n_o}) \tag{2.3.29}$$

Note. For general R, we could obtain Q via Cholesky decomposition or approximate it using the ensemble of perturbations (Evensen 2003),

$$R \simeq \frac{1}{n_e - 1} \sum_{i=1}^{n_e} \mathbf{p}_i {\mathbf{p}_i}^T \tag{2.3.30}$$

leading to

$$Q[:, i] = \mathbf{p}_i \qquad \forall i = 1, \ldots, n_e \tag{2.3.31}$$

Case 1: $n_o \leq n_e$

Emerick and Reynolds (2012) argue that B might suffer from bad scaling, due to observations with values in differing ranges (for example, water cut and fluid rate from the production log of a petroleum well). In Appendix A of the mentioned paper, the authors suggest the following "rescaling" procedure:

$$B = Q\widehat{B}Q^T \tag{2.3.32}$$

where

$$\begin{aligned} \widehat{B} &= Q^{-1} B Q^{-T} \\ &\overset{(2.3.27)}{=\!=} Q^{-1}\tilde{D}\tilde{D}^T Q^{-T} + I_{n_o} \end{aligned} \tag{2.3.33}$$

Therefore, if we compute $\widehat{B}^{-1}$, then, immediately

$$B^{-1} = Q^{-T}\widehat{B}^{-1}Q^{-1} \tag{2.3.34}$$

To this end, let

$$A = Q^{-1}\tilde{D} \tag{2.3.35}$$

and

$$F_1 = [A|I_{n_o}] \tag{2.3.36}$$

be a matrix constructed by appending I_{n_o} to the right of A. Then, it is easy to verify that,

$$\widehat{B} = F_1 {F_1}^T \tag{2.3.37}$$

and we can obtain the inverse by computing the SVD,

$$F_1 = U_1 \Lambda_1 {V_1}^T \tag{2.3.38}$$

In order to avoid rank issues, we will discard the singular vectors corresponding to the least significant singular values (see Golub and van Loan 2012, and also the discussion surrounding (3.3.8)). To that end, it is also worth noting that, since B is a symmetric semi-positive definite matrix (as the sum of two covariance matrices), we are guaranteed that all the singular values are non-negative.

Finally,

$$\widehat{B}^{-1} \overset{(2.3.37)}{=\!=} \left(F_1 {F_1}^T\right)^{-1} \overset{(2.3.38)}{=\!=} U_1 {\Lambda_1}^{-2} {U_1}^T \tag{2.3.39}$$

where we used the fact that V is orthogonal, and, after gathering everything together, (2.3.25) now reads

$$\begin{aligned} C &\overset{\substack{(2.3.34)\\(2.3.39)}}{=\!=} I_{n_e} + \tilde{D}^T Q^{-T} U_1 {\Lambda_1}^{-2} {U_1}^T Q^{-1} \Delta D \\ &= I_{n_e} + \tilde{D}^T \left(Q^{-T} U_1\right) {\Lambda_1}^{-2} \left(Q^{-T} U_1\right)^T \Delta D \end{aligned} \tag{2.3.40}$$

Case 2: $n_o > n_e$

If the observations outnumber the ensemble members, then, according to Mandel (2006), it is advantageous to use the *Sherman-Morrison-Woodbury* inversion formula (which can be consulted in Golub and van Loan 2012),

$$\begin{aligned} B^{-1} &\overset{\substack{(2.3.27)\\(2.3.28)}}{=\!=} \left(\tilde{D}\tilde{D}^T + QQ^T\right)^{-1} \\ &\overset{S-M-W}{=\!=} Q^{-T}Q^{-1} - Q^{-T}Q^{-1}\tilde{D}\left(I_{n_e} + \tilde{D}^T Q^{-T}Q^{-1}\tilde{D}\right)^{-1} \tilde{D}^T Q^{-T}Q^{-1} \\ &\overset{(2.3.35)}{=\!=} Q^{-T}\left[I_{n_o} - A\left(I_{n_e} + A^T A\right)^{-1} A^T\right] Q^{-1} \end{aligned} \tag{2.3.41}$$

In order to solve this, we define

$$F_2 = \left[A^T \middle| I_{n_e}\right] \tag{2.3.42}$$

and compute its SVD,

$$F_2 = U_2 \Lambda_2 {V_2}^T \tag{2.3.43}$$

This allows us to rewrite (2.3.41) as

$$\begin{aligned} B^{-1} &\overset{(2.3.42)}{=\!=} Q^{-T}\left[I_{n_o} - A\left(F_2 {F_2}^T\right)^{-1} A^T\right] Q^{-1} \\ &\overset{(2.3.43)}{=\!=} Q^{-T}\left(I_{n_o} - A U_2 {\Lambda_2}^{-2} {U_2}^T A^T\right) Q^{-1} \end{aligned} \tag{2.3.44}$$

and, finally, by substituting into (2.3.25),

$$C \overset{\substack{(2.3.44)\\(2.3.35)}}{=\!=} I_{n_e} + A^T\left[I_{n_o} - \left(AU_2\right) {\Lambda_2}^{-2} \left(AU_2\right)^T\right] Q^{-1} \Delta D \tag{2.3.45}$$

Notice that, in this case, the rescaling (Emerick and Reynolds 2012) is implicitly applied from the start (2.3.41).

2.3.4. Adapting the assimilation results for simulation

The previous paragraph revealed that the updated ensemble is, in fact, a linear combination of the forecasted state vectors (2.3.24), with coefficients given by matrix C (2.3.25). However, if the model is nonlinear, then the state variables might lose physical consistency. For example, the updated flow variables might not be in line with the new set of rock properties. This can be

overcome by *restarting*, which simply implies running the simulator from time 0 with the updated geological parameters.

Another issue is related to the range of the assimilation results. We learned that each column of C sums up to 1 (Evensen 2003), however, this does not make the Kalman update a convex combination, since it is possible that some of these coefficients become negative. This, in turn, allows the assimilation results to attain unacceptable values, i.e. negative porosities or permeabilities, which are not in line with reality. We study two alternative approaches for dealing with this drawback:

Truncation

A natural approach is to simply replace each outlier with the nearest acceptable value. In order to see the effect, we will perform the twin-experiment described in table 2.2, in which the EnKF is run with an ensemble of 100 members and the truth is the *Y-channel reservoir* from figure 2.4.

Note that the members have hard data constraints regarding the facies at well locations – this is clearly visible in the prior mean and variability (figure 2.9a). We also decided to control all wells based on bottom-hole pressure, since any rate constraint would have created unphysical pressure values in $Prod_3$, due to its position in the nearly-impermeable background shale (see table 2.1). Finally, there is a discrepancy between the observation and assimilation intervals and we treat this asynchronicity in a similar way to the Ensemble Smoother (see paragraph 4.1).

Item	Description
Grid size	$45 \times 45 \times 1$ cells
Physical size of 1 grid cell	15m $\times$ 15m $\times$ 2m
Snesim search ellipsoid	$10 \times 10 \times 1$ grid cells (2D isotropic)
Target marginal distribution for facies	50% shale, 50% sand
Hard data constraints	facies at well locations
Ensemble size	100 members
Initial reservoir pressure	$p_0 = 100$ bars
Initial reservoir saturation	20% water, 80% oil (20% residual oil)
Injector well control	BHP $= 1.5 \cdot p_0$
Producer well control	BHP $= p_0$
Observed variables	water and oil rates in wells
Observation uncertainty	$r = \max(10\% \cdot \text{measured value},\ 0.1)$
Simulation time	34 months
Observation interval	1 month
Assimilation interval	4 months

Table 2.2: Experimental setup for the *Y-channel reservoir*

The posterior mean permeability field and the ensemble variability are shown in figure 2.9b and we see that the result does offer a general idea of the channel locations. However, their structural integrity is fundamentally altered and almost all of the initial variability (figure 2.9a) is lost. Figures 2.10b, 2.10e and 2.10h reveal that this is not a consequence of averaging, since the updated members exhibit similar properties to the posterior mean.

The most likely explanation for this behaviour is that truncation creates inconsistencies in the error covariance matrix, leading to the appearance of spurious correlations. As we will see in chapter 4, they are the main reason for ensemble collapse.

Logistic transform

In order to avoid the side-effects of truncation, we explored the alternative of using the logistic transform. To this end, we introduce the function, $logit : (0,1) \to \mathbb{R}$,

$$logit(s) = \ln\left(\frac{s}{1-s}\right) \qquad \forall s \in (0,1) \tag{2.3.46}$$

and its inverse,

$$logit^{-1}(s) = \frac{1}{1+e^{-s}} \qquad \forall s \in \mathbb{R} \tag{2.3.47}$$

The idea is to use this function as a link between the simulator and the EnKF, by applying (2.3.46) component-wise to the state vector before the assimilation cycle and recovering the updated results via the inverse (2.3.47). This guarantees that the facies values remain in $(0,1)$, and we can interpret them as the "likelihood" that the respective cell is in a channel. Therefore, we can compute the corresponding porosities and permeabilities via the expected value,

$$\begin{aligned} perm_i &= facies_i \cdot perm_{sand} + (1 - facies_i) \cdot perm_{shale} \\ poro_i &= facies_i \cdot poro_{sand} + (1 - facies_i) \cdot poro_{shale} \end{aligned} \qquad i = 1, \ldots, n_e \tag{2.3.48}$$

As can be seen, for example in Myrseth and Omre (2009), this transform can be generally applied to update any bounded variable via the EnKF – we simply need to bring its range to $[0,1]$ via *normalization*. In our case, for permeability,

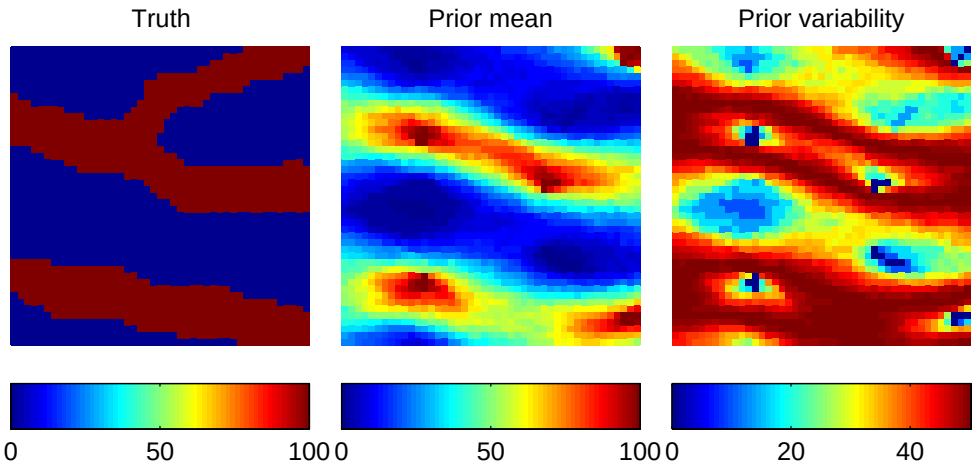
$$facies_i = \frac{perm_i - perm_{shale}}{perm_{sand} - perm_{shale}} \tag{2.3.49}$$

However, note that the *logit* function has singularities for $\{0,1\}$ and we have to apply a small perturbation to the facies values in the initial ensemble. Let *eps* be the smallest floating point value representable in machine precision, then we have

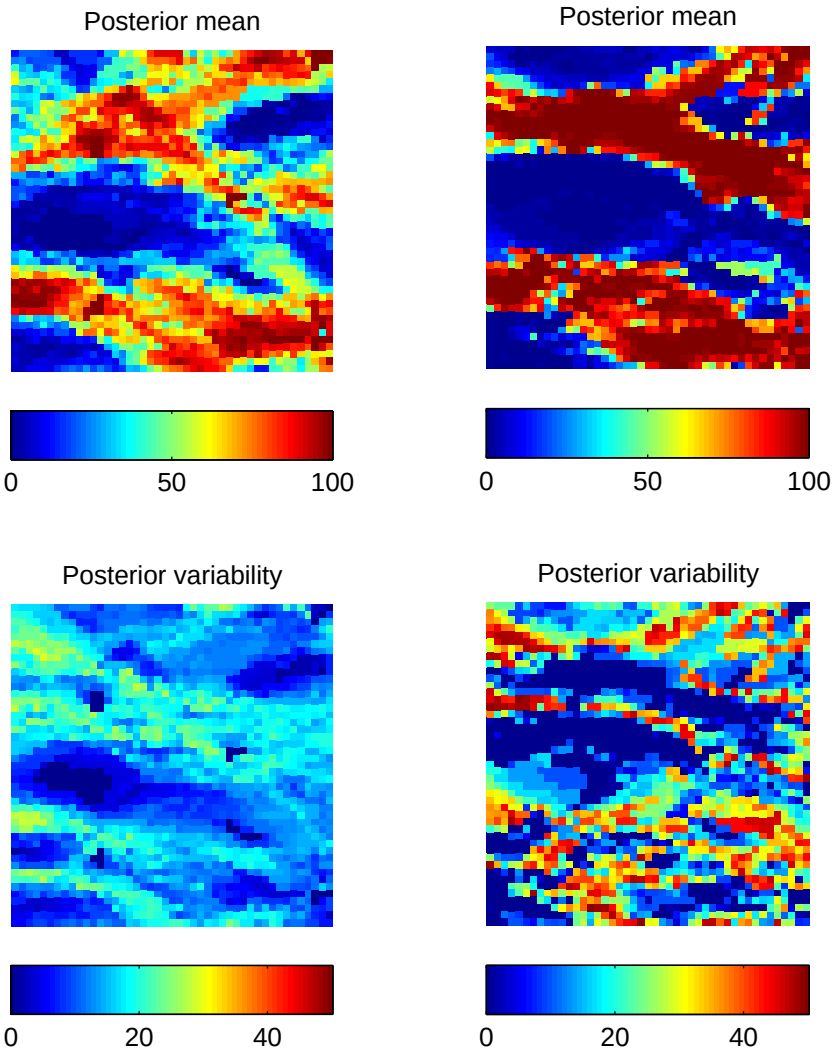
$$\begin{aligned} facies_i(shale) &= eps \\ facies_i(sand) &= 1 - eps \end{aligned} \qquad i = 1, \ldots, n_e \tag{2.3.50}$$

For comparison, we repeated the *Y-channel reservoir* experiment using the logistic transform. We see that the results (figure 2.9c) are a dramatic improvement over those obtained with truncation: the channel structure is much better preserved in the mean and there is considerably more posterior variability than in figure 2.9b. It is also interesting to see the evolution of individual ensemble members; we see that new channels were added (figure 2.10c), while old ones were severed (figures 2.10f) or redirected (figure 2.10i) in order to satisfy the observations, bringing each member's geological structure closer to that of the truth (figure 2.9a).

This leads to the conclusion that the logistic transform is less intrusive on the error covariance and, thus, generally preferable over truncation when handling bounded variables in the EnKF cycle.

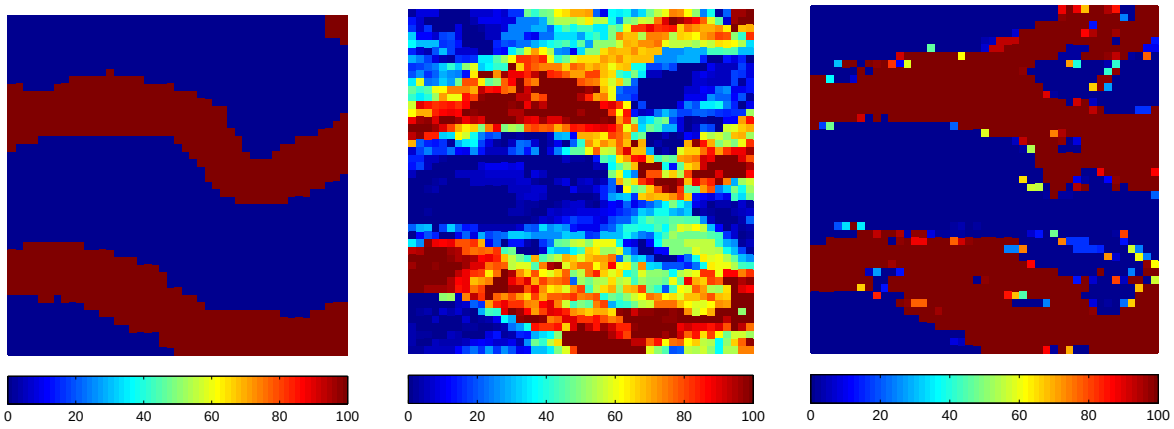


(a) True permeability field and prior information

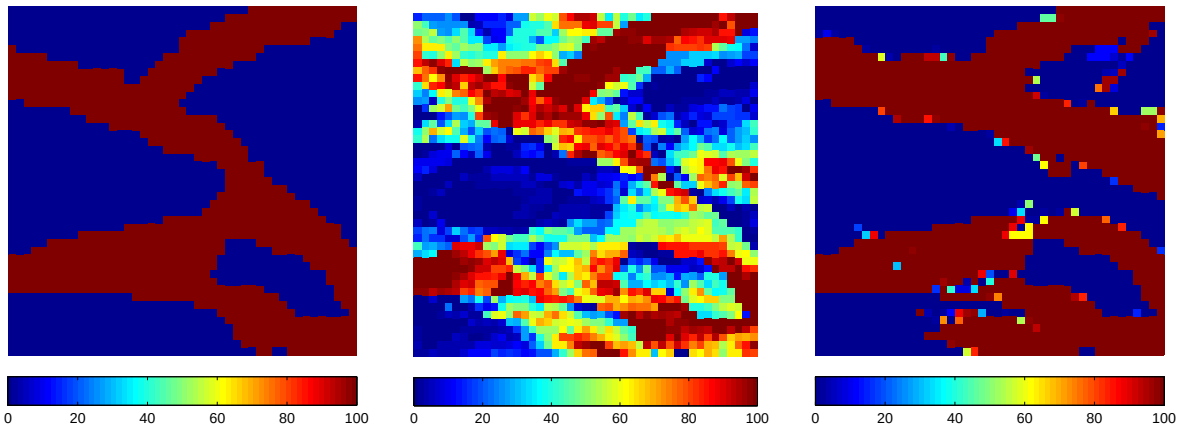


(b) Trucated EnKF results (c) Logistic EnKF result

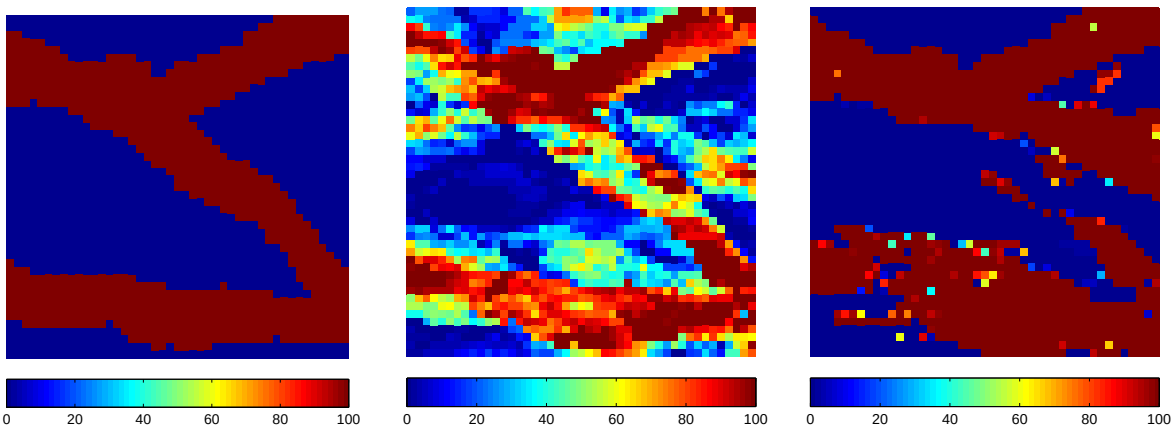
Figure 2.9: Effect of the adaptation methods on the EnKF results



(a) Prior member 3 (b) Truncated EnKF (c) Logistic EnKF



(d) Prior member 12 (e) Truncated EnKF (f) Logistic EnKF



(g) Prior member 67 (h) Truncated EnKF (i) Logistic EnKF

Figure 2.10: Effects of the adaptation methods on individual ensemble members

Chapter 3

Parameterizations and feature spaces

By taking a closer look at the EnKF results from the previous chapter (figure 2.9), we notice that, although the general shape of the channels is captured well, both in the ensemble mean and in individual members, their curvature is not preserved. This is particularly evident for the upper branch of the Y-shaped channel (see figure 2.9c), which plays an important role in predicting future reservoir production, due to the position of $Prod_1$ (figure 2.4). Also, the posterior uncertainty surrounding the channel boundaries (bottom of figure 2.9c) is not big enough to leave room for fundamentally different shapes.

From equation 2.3.24, we recall that the Kalman filter is a linear update, whose computation relies on the error covariance matrix. As such, it is only capable of preserving up to second order statistical moments between the state variables (Sarma, Durlofsky et al. 2008). This is an unfortunate limitation, since, in order to maintain the curved channel structure of our reservoir, we need to take higher-order statistics into account (see definition A.4).

This chapter discusses a possible solution, in the form of nonlinear parameterizations. The bulk of the discussion revolves around the theoretical aspects needed to move the data assimilation framework from the parameter space to a so-called *feature space*. For this purpose, we introduce the polynomial kernel PCA algorithm and show the effect of its application to the EnKF.

3.1. Parameterized EnKF

Using a *parameterization* simply implies performing a change of variable on the state vector before the data assimilation cycle, as illustrated in figure 3.1.

Notice that the parameterization is an intermediate component, separating the parameter space (where the simulator operates) from the, so-called, feature space (where the assimilation takes place). In a sense, we already used this approach when applying the logistic transform (paragraph 2.3.4).

Recall that the elements of the parameter space are the state vectors and their components have physical meaning (permeability, porosity, etc). We will refer to them using latin letters ($\mathbf{x}$, $\mathbf{y}$ or $\mathbf{t}$) and the goal is to map them into the feature space via the (possibly nonlinear) mapping, $\Phi(\mathbf{x})$. Following the convention in the literature (see, for example, Schölkopf, Mika, Burges et al. 1999; Kwok and Tsang 2004; or Honeine and Richard 2011b), we will call $\Phi(\mathbf{x})$ the *image* and $\mathbf{x}$, its *preimage*. However, as explained in paragraph 3.4, we are generally unable to operate directly on the images, but on their lower-dimensional projections, $\boldsymbol{\xi}$, which are tied to the state vectors via the parameterization, $\boldsymbol{\xi} = g(\mathbf{x})$.

As such, $g : \mathbb{R}^m \to \mathbb{R}^l$ is an invertible function, which allows us to perform data assimilation in the feature space. This leads to the following formulation of the parameterized Kalman update

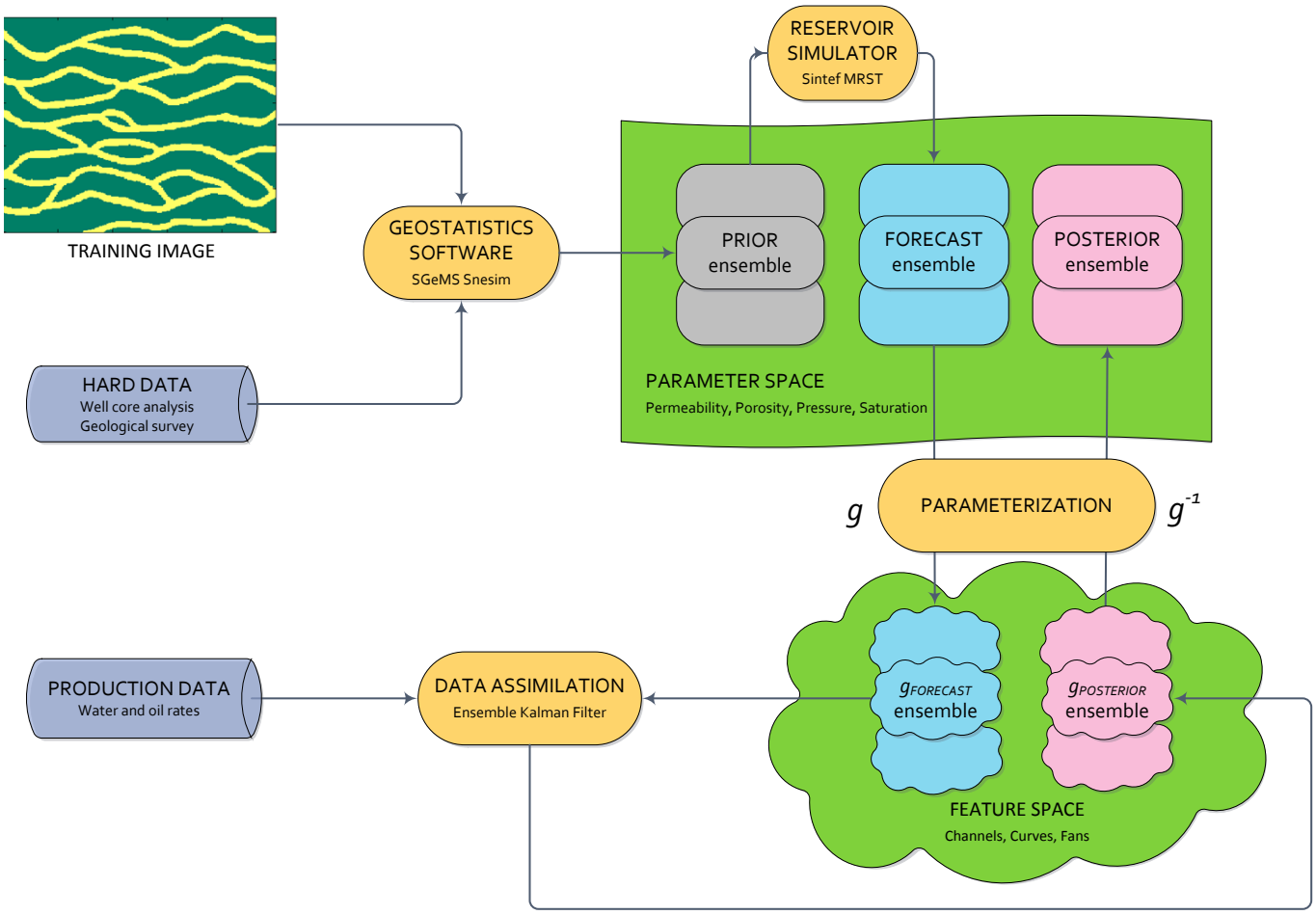


Figure 3.1: Parameterized EnKF workflow

equation (Sarma and W. Chen 2013)

$$\begin{aligned} \boldsymbol{\xi}^f &= g\left(\mathbf{x}^f\right) \\ \boldsymbol{\xi}^a &= \boldsymbol{\xi}^f + K_{gain}\boldsymbol{\Delta}\mathbf{d} \\ \mathbf{x}^a &= g^{-1}\left(\boldsymbol{\xi}^a\right) \end{aligned} \tag{3.1.1}$$

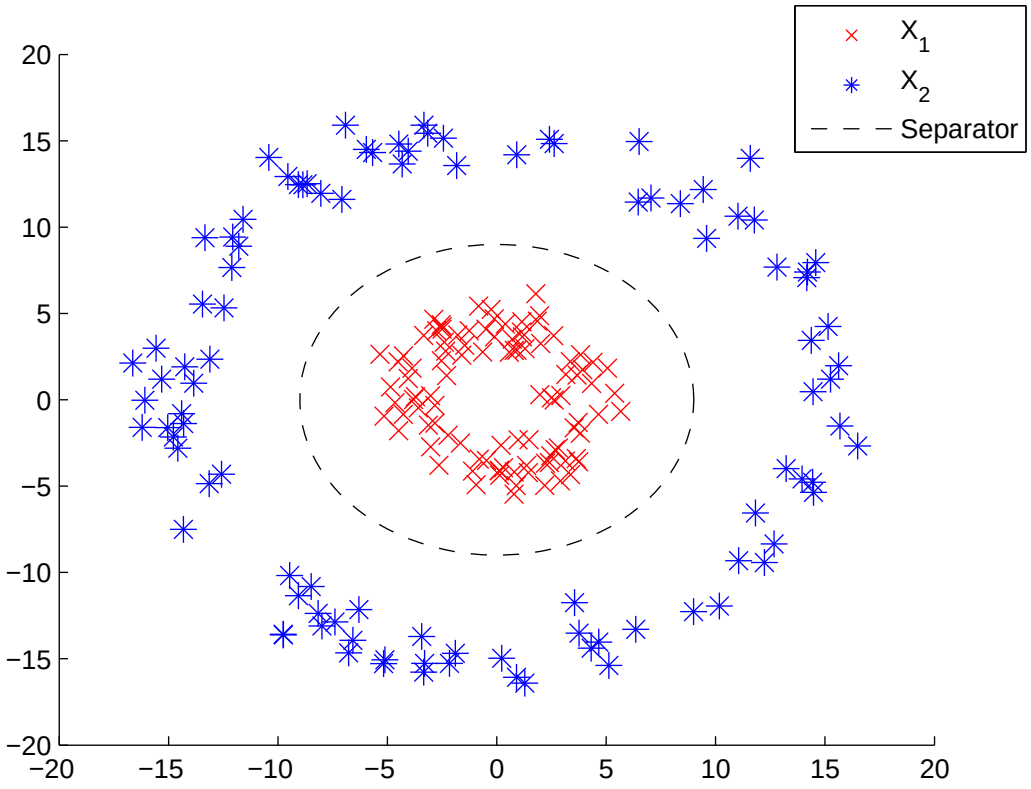
where the Kalman gain is now computed such that it minimizes the mean squared error in the feature space,

$$MSE^*(\mathbf{x}) = g(\mathbf{x}) - g(\mathbf{x}_{truth}) \tag{3.1.2}$$

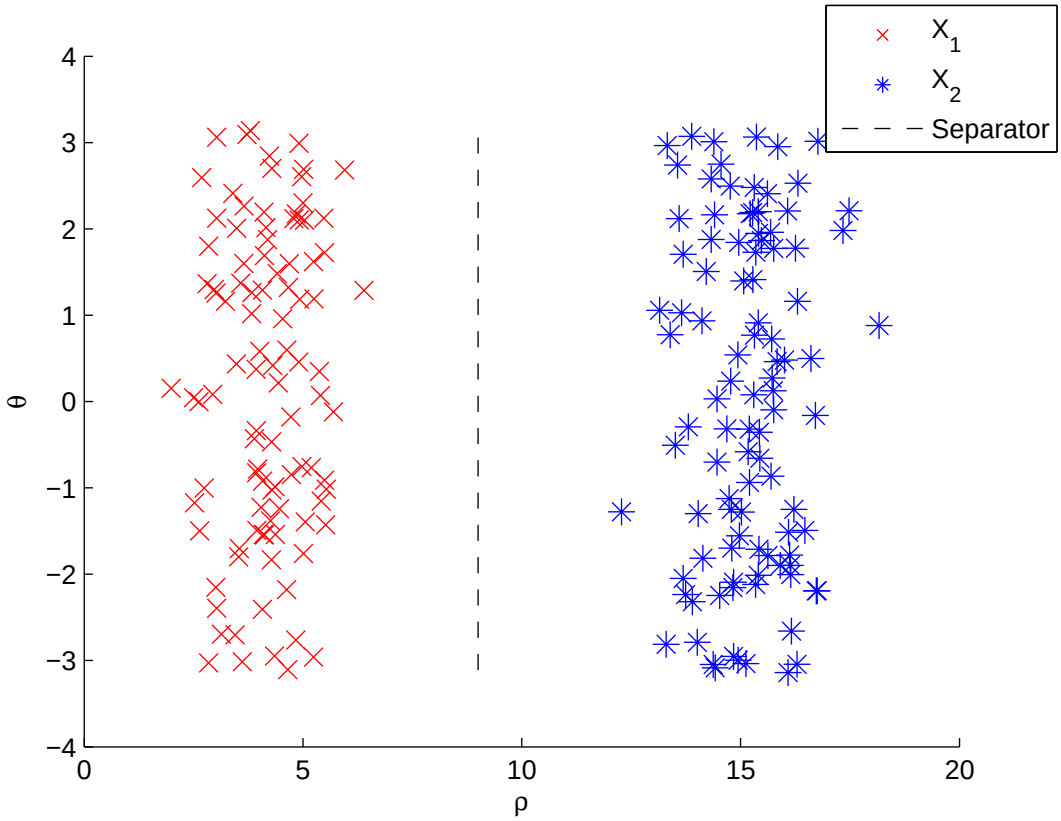
The effect this has on the assimilation results depends on the choice for g and the properties of the underlying feature space. For our application, we desire that the the high-order moments of the facies field are "reflected" in the $\boldsymbol{\xi}$ and that these properties are preserved during linear transformations (such as the Kalman filter update). This will be the focus for the remainder of this chapter.

3.2. Feature spaces

The two classes of points in figure 3.2a are separable by a circle – we call this a *feature* of this particular dataset. If we perform a polar coordinate transformation, as in figure 3.2b, then the circle is mapped to a line and the dataset becomes linearly separable (see paragraph A.2). We will call the polar coordinate space a *feature space* for this dataset, because it emphasizes a key feature of its structure.



(a) Nonlinearly separable in the Cartesian coordinate space



(b) Linearly separable in the polar coordinate space

Figure 3.2: Input vs. feature space

In the general case, let $\Phi : \mathbb{R}^m \to F$ be a (possibly nonlinear) mapping from the parameter space, $\mathbb{R}^m$, to a desired feature space, F. Then F is a p-dimensional space whose coordinates are tied (through Φ) to the initial variables. The value of p depends on the general structure of the dataset and the complexity of the target feature to be identified (Aizerman et al. 1964). It can also happen that $p = \infty$ (see Gaussian kernel, below).

3.2.1. Kernel functions

Aizerman et al. (1964) noted that, especially when $p \gg m$, Φ may become expensive to evaluate and this poses a problem when working with large datasets. It might also happen that Φ doesn't have a closed form and, therefore, the properties of F are not fully known. However, if F is a *Hilbert space* (see, for example, Kreyszig 2007), then many algorithms can be formulated to work solely with inner product evaluations. For this purpose, we define the *kernel function*,

$$\begin{aligned} &\mathfrak{K} : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R} \\ &\mathfrak{K}(\mathbf{x}, \mathbf{y}) = \langle \Phi(\mathbf{x}), \Phi(\mathbf{y}) \rangle_F \end{aligned} \tag{3.2.1}$$

If the kernel function is less expensive to compute than Φ, then any algorithm that only requires inner product evaluations can operate efficiently in F through $\mathfrak{K}$.

From a bottom-up perspective, one can arbitrarily choose a kernel function, disregarding Φ altogether, and work in the associated feature space. Cortes and Vapnik (1995) showed that a choice for $\mathfrak{K}$ is valid (i.e. it corresponds to a Hilbert feature space) if it satisfies Mercer's condition:

Definition 3.1: Positive-definite kernel

(Cortes and Vapnik 1995) Let $\mathcal{X}$ be a nonempty set, $\mathcal{X} \subseteq \mathbb{R}^m$. The symmetric function $\mathfrak{K} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is called a *positive definite* (PD) kernel if it satisfies

$$\sum_{i,j=1}^{n} \mathfrak{K}(\mathbf{x}_i, \mathbf{x}_j) \cdot c_i c_j \geq 0 \tag{3.2.2}$$

for $n \geq 1$, $\forall \mathbf{x}_k \in \mathcal{X}$ and $\forall c_k \in \mathbb{R}$.

Notable positive definite kernel functions are:

- The **linear kernel**

$$\mathfrak{K}_{lin}(\mathbf{x}, \mathbf{y}) = \langle \mathbf{x}, \mathbf{y} \rangle_2 + c \qquad\qquad c \in \mathbb{R} \tag{3.2.3}$$

  is the regular dot product. It can help identify properties related to lines (in $\mathbb{R}^2$) and (hyper)planes (in $\mathbb{R}^m$ with $m \geq 3$).

- The **polynomial kernel**

$$\mathfrak{K}_{poly}(\mathbf{x}, \mathbf{y}) = (a \cdot \langle \mathbf{x}, \mathbf{y} \rangle_2 + c)^d \qquad\qquad a, c \in \mathbb{R},\ d \in \mathbb{N}^* \tag{3.2.4}$$

  operates in a feature space whose coordinates are monomials of degree d of the initial variables. It has widespread use in natural language processing (Chang et al. 2010).

- The **Gaussian kernel**

$$\mathfrak{K}_{gauss}(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{y}\|_2^{\,2}}{2\sigma^2} \right) \qquad\qquad \sigma \in \mathbb{R} \tag{3.2.5}$$

  corresponds to an infinite dimensional feature space of Gaussian functions. This kernel is the most popular member of the *radial basis function* (RBF) class. As such, it can identify

properties related to circles and (hyper)spheres and is frequently used in machine learning models called Support Vector Machines (Schölkopf and Smola 2002).

The value of σ dictates the kernel's sensitivity to the distance between arguments. If σ is too low, it makes the classification unstable and vulnerable to noise. On the opposite side, having σ too large translates to an almost linear behaviour, making the results less responsive to fluctuations in the data (Schölkopf, Mika, Burges et al. 1999).

3.2.2. Operations in feature spaces

The power of kernel functions lies in the fact that they allow us to operate efficiently in high-dimensional spaces. This paragraph gives the kernel-formulation for a few elementary operations.

Computing lengths and distances

As stated before, an algorithm can benefit from the use of the kernel function if it can be rewritten entirely using inner-product evaluations. Fortunately, for lengths and distances this is straightforward, given the properties of the norm in Hilbert spaces (see Kreyszig 2007):

$$\|\Phi(\mathbf{x})\|_F = \sqrt{\langle \Phi(\mathbf{x}), \Phi(\mathbf{x}) \rangle_F} = \sqrt{\mathfrak{K}(\mathbf{x}, \mathbf{x})} \qquad \forall \mathbf{x} \in \mathbb{R}^m \tag{3.2.6}$$

and

$$\begin{aligned} d_F(\Phi(\mathbf{x}), \Phi(\mathbf{y})) &= \|\Phi(\mathbf{x}) - \Phi(\mathbf{y})\|_F \qquad \forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^m \\ &= \sqrt{\langle \Phi(\mathbf{x}) - \Phi(\mathbf{y}), \Phi(\mathbf{x}) - \Phi(\mathbf{y}) \rangle_F} \\ &= \sqrt{\langle \Phi(\mathbf{x}), \Phi(\mathbf{x}) \rangle_F - \langle \Phi(\mathbf{x}), \Phi(\mathbf{y}) \rangle_F - \langle \Phi(\mathbf{y}), \Phi(\mathbf{x}) \rangle_F + \langle \Phi(\mathbf{y}), \Phi(\mathbf{y}) \rangle_F} \\ &= \sqrt{\mathfrak{K}(\mathbf{x}, \mathbf{x}) - 2\mathfrak{K}(\mathbf{x}, \mathbf{y}) + \mathfrak{K}(\mathbf{y}, \mathbf{y})} \end{aligned} \tag{3.2.7}$$

respectively, where we used the symmetry of the kernel function.

The distance function attains a special interpretation in feature spaces – it is a measure of the *dissimilarity* between two samples, in terms of the analyzed features. Therefore, equation (3.2.7) lays the foundation for the kernel-formulation of distance-based clustering algorithms such as K-means (Forgy 1965; and Lloyd 1982).

Mean-centering

Even if a dataset, $\mathbf{t}_1, \ldots, \mathbf{t}_{n_t}$, has mean zero in $\mathbb{R}^m$, there is no guarantee that this property is carried over when mapping into the feature space, F.

Let

$$\overline{\Phi_{\mathbf{t}}} = \frac{1}{n_t} \sum_{i=1}^{n_t} \Phi(\mathbf{t}_i) \tag{3.2.8}$$

be the mean of the images. Then, following (Schölkopf, Smola and Müller 1998, Appendix B), we

define the *mean-centered kernel function* as

$$\begin{aligned}\widetilde{\mathfrak{K}}(\mathbf{t}_i,\mathbf{t}_j) &= \left\langle \widetilde{\Phi}(\mathbf{t}_i), \widetilde{\Phi}(\mathbf{t}_j) \right\rangle_F \\ &= \left\langle \Phi(\mathbf{t}_i) - \overline{\Phi_{\mathbf{t}}}, \Phi(\mathbf{t}_j) - \overline{\Phi_{\mathbf{t}}} \right\rangle_F \\ &= \langle \Phi(\mathbf{t}_i), \Phi(\mathbf{t}_j)\rangle_F - \left\langle \Phi(\mathbf{t}_i), \overline{\Phi_{\mathbf{t}}} \right\rangle_F \\ &\quad - \left\langle \overline{\Phi_{\mathbf{t}}}, \Phi(\mathbf{t}_j) \right\rangle_F + \left\langle \overline{\Phi_{\mathbf{t}}}, \overline{\Phi_{\mathbf{t}}} \right\rangle_F \qquad \forall i,j = 1,\dots,n_t \\ &= \mathfrak{K}(\mathbf{t}_i,\mathbf{t}_j) - \frac{1}{n_t}\sum_{k_1=1}^{n_t} \mathfrak{K}(\mathbf{t}_{k_1},\mathbf{t}_j) \\ &\quad - \frac{1}{n_t}\sum_{k_2=1}^{n_t} \mathfrak{K}(\mathbf{t}_i,\mathbf{t}_{k_2}) + \frac{1}{{n_t}^2}\sum_{k_1=1}^{n_t}\sum_{k_2=1}^{n_t} \mathfrak{K}(\mathbf{t}_{k_1},\mathbf{t}_{k_2})\end{aligned} \tag{3.2.9}$$

We also define the *kernel matrix*,

$$K(i,j) = \mathfrak{K}(\mathbf{t}_i,\mathbf{t}_j) \qquad \forall i,j = 1,\dots,n_t \tag{3.2.10}$$

in order to rewrite equation (3.2.9) compactly as

$$\begin{aligned}\widetilde{K} &= K - 1_{n_t}K - K1_{n_t} + 1_m K 1_{n_t} \\ &= (I_{n_t} - 1_{n_t})K(I_{n_t} - 1_{n_t})\end{aligned} \tag{3.2.11}$$

This result is crucial for the kernel PCA algorithm, given in paragraph 3.4.

3.2.3. The polynomial feature space

The previous discussion did not make any assumptions about the choice of kernel function. However, our aim is to find a feature space in which data assimilation algorithms preserve high-order moments of the state vector. Reviewing definition A.4, the polynomial feature space seems to be a natural choice, as its coordinates are monomials of the initial variables. To illustrate, let's consider a state vector with $m = 2$ variables, $\mathbf{x} = \begin{bmatrix} x_1 & x_2 \end{bmatrix}^T$. Then, the mapping to the space of all monomials of order $d = 2$ is

$$\begin{aligned}&\Phi_2 : \mathbb{R}^2 \to F_2 \\ &\Phi_2(\mathbf{x}) = \begin{bmatrix} {x_1}^2 & {x_2}^2 & x_1x_2 & x_2x_1 \end{bmatrix}^T\end{aligned} \tag{3.2.12}$$

and dot products within this space can be computed using the following polynomial kernel function

$$\mathfrak{K}_2(\mathbf{x},\mathbf{y}) = \langle \mathbf{x},\mathbf{y}\rangle_2^{\,2} = {x_1}^2{y_1}^2 + {x_2}^2{y_2}^2 + 2x_1x_2y_1y_2 \qquad \forall \mathbf{x},\mathbf{y} \in \mathbb{R}^2 \tag{3.2.13}$$

Schölkopf and Smola (2003) showed that (3.2.13) can be generalized for any choice of m and d,

$$\mathfrak{K}_d(\mathbf{x},\mathbf{y}) = \langle \mathbf{x},\mathbf{y}\rangle_2^{\,d} \qquad \forall \mathbf{x},\mathbf{y} \in \mathbb{R}^m \tag{3.2.14}$$

and they also computed the dimension of the corresponding feature space,

$$\binom{m+d-1}{d} = \frac{(m+d-1)!}{d!\cdot(m-1)!} \tag{3.2.15}$$

Sarma et al. introduced the idea of incorporating kernel parameterizations in data assimilation algorithms in order to preserve multipoint geostatistics. Their formulation is applicable to the adjoint method (Sarma, Durlofsky et al. 2008) as well as the EnKF (Sarma and W. Chen 2009) and it uses the following kernel function

$$\mathfrak{K}_{1\dots d}(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{d} \langle \mathbf{x},\mathbf{y}\rangle_2^{\,i} \qquad \forall \mathbf{x},\mathbf{y} \in \mathbb{R}^m, \quad \forall d \in \mathbb{N}^* \tag{3.2.16}$$

which corresponds to the space of all monomials up to order d. By using equation (3.2.15), the dimensionality of this feature space is

$$p = \sum_{i=1}^{d} \binom{m+i-1}{i} \tag{3.2.17}$$

thus, prohibitively large even for relatively small values of m and d. However, there is a high degree of redundancy – some dimensions are permutations of the same monomial – and the next paragraph discusses a way to select only the most relevant subset of these dimensions.

3.3. Principal component analysis

In 1901, Pearson described a method of computing the best line fit for a set of points in the Euclidean plane, an idea which was later developed into the Principal Component Analysis (PCA) algorithm by Hotelling (1933). It is a transformation that exposes a dataset's internal structure from a perspective that best explains the variance. The principal components are an ordered set of vectors which form the orthonormal basis of this new perspective, such that

- The first principal component is the direction of the biggest spread among the data points.
- The second principal component accounts for the biggest spread, after the contribution of the first component has been removed.
- The i-th principal component accounts for the biggest spread, after the contributions of the first $i-1$ components have been removed.

PCA operates on datasets with zero mean. If this is not the case, then the first principal components will be perturbed by the mean and will not accurately show the directions of greatest variance. Therefore, in order to get informative results, the dataset, $\mathbf{t}_1, \dots \mathbf{t}_{n_t}$, needs to be first centered (A.1.2) around its mean, $\overline{\mathbf{t}}$.

Formally, the above can be summarized as

Definition 3.2: Principal Component Analysis

(Jolliffe 2005) Let $\widetilde{\mathbf{t}}_1, \dots \widetilde{\mathbf{t}}_{n_t}$ be mean-centered samples of the random vector $\mathbf{X} \in \mathbb{R}^m$. *Principal component analysis* is an orthogonal linear transformation, $W \in \mathbb{R}^{m \times l}$ with $l \leq m$, such that

$$\min_{W} \sum_{i=1}^{n_t} \left\| \widetilde{\mathbf{t}}_i - WW^T \widetilde{\mathbf{t}}_i \right\|_2^2 \tag{3.3.1}$$

Hotelling (1933) showed that the principal components are the eigenvectors, $\mathbf{v}_j$, of the covariance matrix, S (A.1.5). Thus, they are obtained by solving the following eigenproblem (see, for example, Golub and van Loan 2012)

$$\begin{aligned} &S\mathbf{v}_j = \lambda_j \mathbf{v}_j \\ &\|\mathbf{v}_j\|_2 = 1 \end{aligned} \qquad \forall j = 1, \dots, m \tag{3.3.2}$$

where λ_j are the corresponding eigenvalues.

Covariance matrices are symmetric positive semidefinite by definition (Johnson and Wichern 2002). This has two implications for PCA. First, the eigenvalues and eigenvectors can be computed through singular value decomposition (see Golub and van Loan 2012). Second, all of the eigenvalues are real nonnegative numbers and, according to Hotelling (1933), the magnitude of λ_j is proportional to the "importance" of the associated principal component, $\mathbf{v}_j$.

An important observation is given in the following

Corollary 3.3: Linearity of PCA

The principal components are a linear combination of the training dataset.

Proof. We substitute the definition of S (A.1.5) into (3.3.2),

$$\begin{aligned}\lambda_j \mathbf{v}_j &= \left(\frac{1}{n_t - 1}\sum_{i=1}^{n_t} \tilde{\mathbf{t}}_i \tilde{\mathbf{t}}_i^{\,T}\right)\mathbf{v}_j \\ &= \frac{1}{n_t - 1}\sum_{i=1}^{n_t} \tilde{\mathbf{t}}_i \left(\tilde{\mathbf{t}}_i^{\,T}\mathbf{v}_j\right) \qquad \forall j = 1, \dots, m \\ &= \frac{1}{n_t - 1}\sum_{i=1}^{n_t} \left\langle \tilde{\mathbf{t}}_i, \mathbf{v}_j \right\rangle_2 \tilde{\mathbf{t}}_i\end{aligned} \tag{3.3.3}$$

leading to

$$\begin{aligned}\mathbf{v}_j &= \sum_{i=1}^{n_t} \boldsymbol{\alpha}_j[i] \cdot \tilde{\mathbf{t}}_i \\ \boldsymbol{\alpha}_j[i] &= \frac{\left\langle \tilde{\mathbf{t}}_i, \mathbf{v}_j \right\rangle_2}{(n_t - 1)\lambda_j}\end{aligned} \qquad \forall j = 1, \dots, m \tag{3.3.4}$$

□

Notice that we used the term *training dataset*. This is because the principal components form an orthonormal basis in $\mathbb{R}^m$, with orientation given by the $\mathbf{t}_i$. However, once they are obtained, we can express any $\mathbf{x} \in \mathbb{R}^m$, in terms of its projections onto the $\mathbf{v}_j$:

$$\tilde{\mathbf{x}} = \sum_{j=1}^{m} \left\langle \tilde{\mathbf{x}}, \mathbf{v}_j \right\rangle_2 \cdot \mathbf{v}_j = \sum_{j=1}^{m} \boldsymbol{\xi}[j] \cdot \mathbf{v}_j \qquad \boldsymbol{\xi} \in \mathbb{R}^m \tag{3.3.5}$$

where $\tilde{\mathbf{x}} = \mathbf{x} - \bar{\mathbf{t}}$ is centered around $\bar{\mathbf{t}}$, the mean of the training dataset.

The projection vector, $\boldsymbol{\xi} \in \mathbb{R}^m$, can be written compactly as

$$\boldsymbol{\xi} = V^T \tilde{\mathbf{x}} = V^T(\mathbf{x} - \bar{\mathbf{t}}) \tag{3.3.6}$$

Figure 3.3 shows an example where the Cartesian coordinates are sampled from $m = 2$ positively correlated variables. Notice that the datapoints are, in effect, rotated such that the direction of the highest variance lies on the horizontal axis.

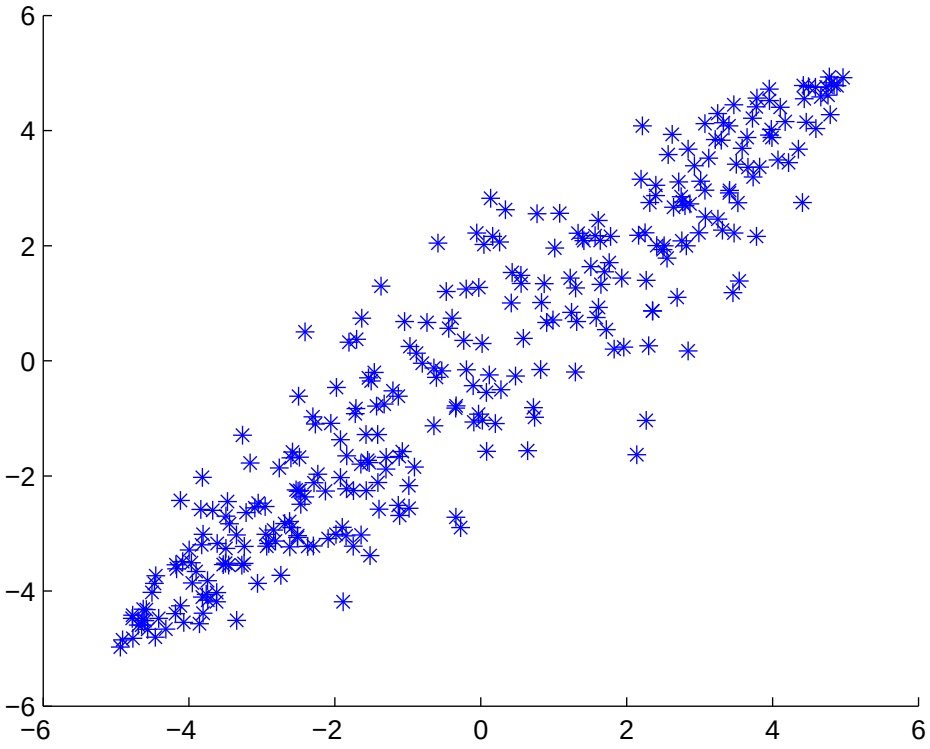
According to Hotelling (1933), most of the information is, usually, captured by the first few principal components. We can, thus, define a truncated version of PCA, which retains only the most significant $l \leq m$ components, leading to dimensionality reduction (see paragraph A.3), i.e.

$$\tilde{\mathbf{x}} \simeq \sum_{j=1}^{l} \boldsymbol{\xi}[j] \cdot \mathbf{v}_j \qquad \boldsymbol{\xi} \in \mathbb{R}^l,\ l \leq m \tag{3.3.7}$$

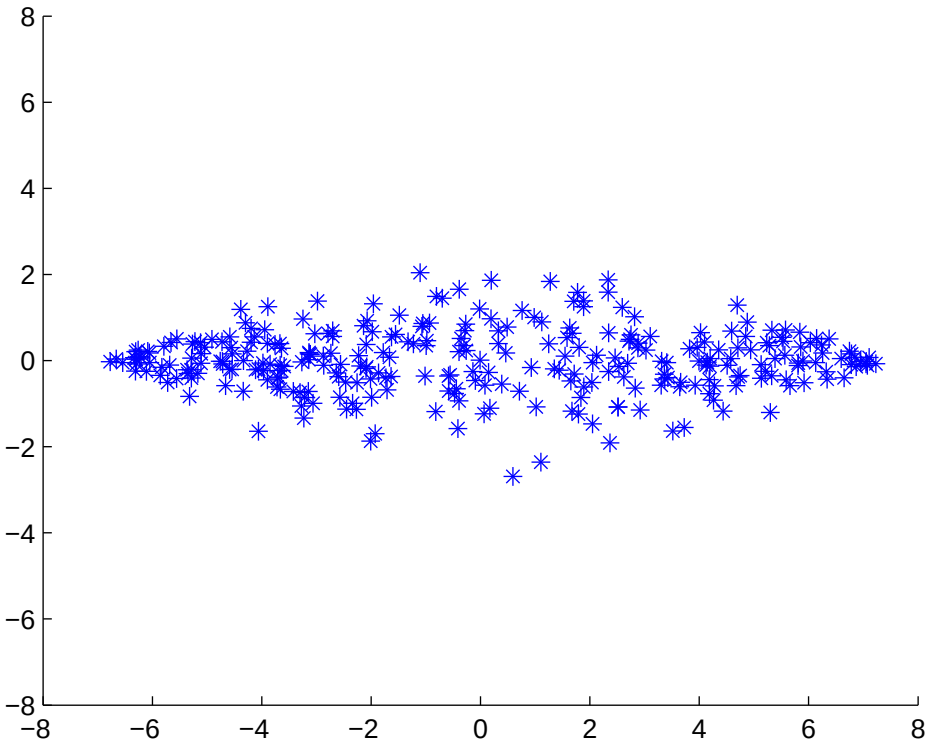
l can either be fixed – for example, a value of 2 or 3 allows visualization of the results – or determined dynamically, based on the dataset at hand. A detailed discussion of the different ways to do this can be consulted in (Jolliffe 2005); we will only mention the simplest and most straightforward criterion:

If we consider the eigenvalues in decreasing order, $\lambda_1 \geq \cdots \geq \lambda_m$, then, in order to preserve $q\%$ of the initial variance, l needs to be chosen as the smallest number satisfying

$$\frac{\sum_{j=1}^{l} \lambda_j}{\sum_{j=1}^{m} \lambda_j} \geq \frac{q}{100} \tag{3.3.8}$$



(a) original dataset



(b) projection onto the principal components

Figure 3.3: The PCA transform

Algorithm 3.1: Principal Component Analysis

Input: The training dataset $\mathbf{t}_1, \ldots, \mathbf{t}_{n_t} \in \mathbb{R}^m$

Output: The transformation matrix W

① Center the dataset around its mean, $\bar{\mathbf{t}}$, (A.1.2) to obtain $\tilde{\mathbf{t}}_1, \ldots, \tilde{\mathbf{t}}_{n_t}$.

② Compute the covariance matrix, S (A.1.5)

③ Compute the SVD of S

$$S = V \Lambda V^T \tag{3.3.9}$$

where $\Lambda = diag(\lambda_1, \ldots, \lambda_m)$ and $V[:, j] = \mathbf{v}_j$.

④ Order the eigenvectors in decreasing order of the associated eigenvalues and store the first $l \leq m$ of them in the columns of matrix W.

⑤ Use W to perform transformations

$$\boldsymbol{\xi} = W^T(\mathbf{x} - \bar{\mathbf{t}}) \qquad \forall \mathbf{x} \in \mathbb{R}^m \tag{3.3.10}$$

3.4. Kernel PCA

As stated before, we intend to use PCA in the feature space in order to work with a manageable subset of its dimensions. A crucial result in this direction was developed by Schölkopf, Smola and Müller (1998), who showed that PCA can be formulated solely in terms of inner product evaluations. Here, we will only give an outline of the key steps from their argument.

As usual, consider the training dataset $\mathbf{t}_1, \ldots, \mathbf{t}_{n_t} \in \mathbb{R}^m$ and let F be the desired feature space, with $\mathfrak{K}$ the corresponding kernel function. Then, by mapping into F, we obtain $\Phi(\mathbf{t}_1), \ldots, \Phi(\mathbf{t}_{n_t})$. After mean-centering, the covariance matrix reads

$$S = \frac{1}{n_t - 1} \sum_{i=1}^{n_t} \tilde{\Phi}(\mathbf{t}_i) \tilde{\Phi}(\mathbf{t}_i)^T \tag{3.4.1}$$

and the principal components are its eigenvectors, $\mathbf{v}_j$, satisfying

$$\begin{aligned} &S\mathbf{v}_j = \lambda_j \mathbf{v}_j \\ &\|\mathbf{v}_j\|_2 = 1 \end{aligned} \qquad \forall j = 1, \ldots, p \tag{3.4.2}$$

where λ_j are the corresponding eigenvalues.

Similar to (3.3.4), we have

$$\mathbf{v}_j = \sum_{i=1}^{n_t} \boldsymbol{\alpha}_j[i] \cdot \tilde{\Phi}(\mathbf{t}_i) \qquad \boldsymbol{\alpha}_j \in \mathbb{R}^{n_t} \qquad \forall j = 1, \ldots, p \tag{3.4.3}$$

which allows the eigenproblem (3.4.2) to be rewritten as (see Schölkopf, Smola and Müller 1998, for the proof):

$$\begin{aligned} &\mu_i \boldsymbol{\alpha}_i = \tilde{K} \boldsymbol{\alpha}_i \\ &\|\boldsymbol{\alpha}_i\|_2 = \frac{1}{\sqrt{\mu_i}} \end{aligned} \qquad \forall i = 1, \ldots, n_t \tag{3.4.4}$$

where $\widetilde{K}$ is the mean-centered kernel matrix (3.2.11) and

$$\mu_i = (n_t - 1)\lambda_i \qquad \forall i = 1, \ldots, n_t \tag{3.4.5}$$

its eigenvalues.

Projections onto the principal components are done through the kernel function

$$\begin{aligned} \boldsymbol{\xi}[j] &= \left\langle \mathbf{v}_j, \widetilde{\Phi}(\mathbf{x}) \right\rangle_F \\ &\overset{(3.4.3)}{=\!=} \sum_{i=1}^{n_t} \boldsymbol{\alpha}_j[i] \cdot \left\langle \widetilde{\Phi}(\mathbf{t}_i), \widetilde{\Phi}(\mathbf{x}) \right\rangle_F \qquad \forall j = 1, \ldots, n_t \quad \forall \mathbf{x} \in \mathbb{R}^m \\ &= \sum_{i=1}^{n_t} \boldsymbol{\alpha}_j[i] \cdot \widetilde{\mathfrak{K}}(\mathbf{t}_i, \mathbf{x}) \end{aligned} \tag{3.4.6}$$

The new method is called *kernel PCA* and it performs PCA in the feature space through the eigendecomposition of the kernel matrix. Since the kernel function satisfies Mercer's condition (3.2.2), K is positive semidefinite, hence, just as before, we have $\mu_i \geq 0$ for $\forall i = 1, \ldots, n_t$ and the eigenvectors can be obtained through SVD.

Given the high dimensionality of the feature space, we opt to keep only the $l \ll p$ most significant components. The equivalent of (3.3.7) then becomes

$$\widetilde{\Phi}(\mathbf{x}) \simeq \sum_{i=1}^{l} \boldsymbol{\xi}[i] \cdot \mathbf{v}_i \qquad \boldsymbol{\xi} \in \mathbb{R}^l,\ l \leq n_t \ll p \tag{3.4.7}$$

and l is chosen according to the criterion presented in paragraph 3.3.

Algorithm 3.2: Kernel Principal Component Analysis

Input: The training dataset $\mathbf{t}_1, \ldots, \mathbf{t}_{n_t} \in \mathbb{R}^m$

Output: The transformation matrix W

① Compute the kernel matrix K (3.2.10).

② Mean-center the kernel matrix (3.2.11) to obtain $\widetilde{K}$.

③ Compute the SVD of $\widetilde{K}$

$$\widetilde{K} = A \Sigma A^T \tag{3.4.8}$$

where $\Sigma = diag(\mu_1, \ldots, \mu_m)$ and $A[:, i] = \boldsymbol{\alpha}_i$.

④ Normalize the $\boldsymbol{\alpha}_i$ according to (3.4.4).

⑤ Order the eigenvectors in decreasing order of the associated eigenvalues and store the first $l \leq n_t$ of them in the columns of matrix W.

⑥ Use W to perform transformations

$$\begin{aligned} \boldsymbol{\xi} &= W^T \widetilde{\mathbf{k}} & \forall \mathbf{x} &\in \mathbb{R}^m \\ \widetilde{\mathbf{k}}[i] &= \widetilde{\mathfrak{K}}(\mathbf{t}_i, \mathbf{x}) & \forall i &= 1, \ldots, n_t \end{aligned} \tag{3.4.9}$$

The algorithm is generic, in the sense that any kernel function can be used. Schölkopf, Smola and Müller (1998) showed that the results for the linear kernel (3.2.3) with parameter $c = 0$ are equivalent to regular PCA. However, since the size of the kernel matrix is equal to the number of the training dataset, the linear kernel PCA algorithm is more efficient than the one presented in paragraph 3.3 when $n_t \ll m$.

As an example, figure 3.4 shows the effect of Gaussian kernel PCA with $l = 2$ on the circularly separable dataset presented earlier (figure 3.2a). Through a radial basis function transform, the algorithm was able to identify the two classes of points, making them linearly separable.

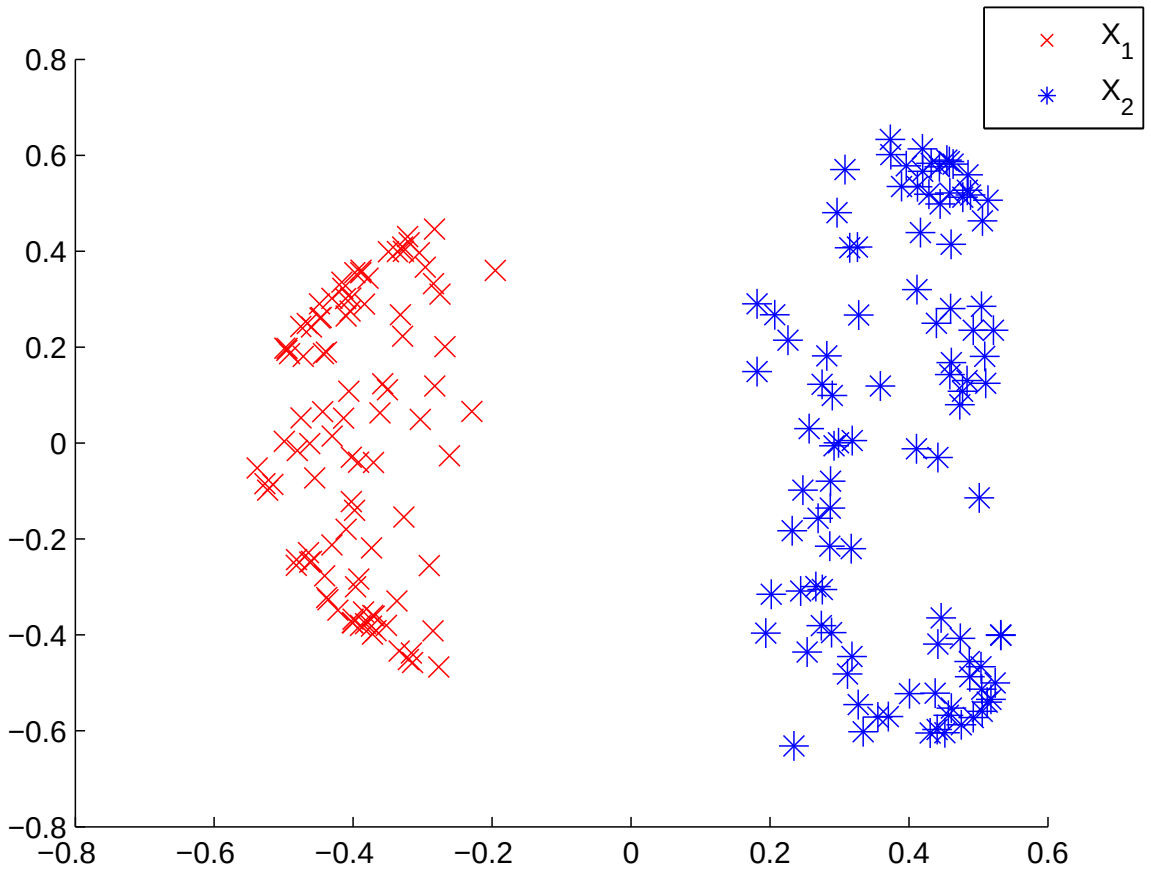


Figure 3.4: Example kernel PCA result using the Gaussian kernel with $\sigma = 7.89$ and $l = 2$

3.5. The preimage problem

We would now like to use polynomial kernel PCA as a parameterization for the EnKF, in the hope that it will help preserve high-order geostatistics throughout the assimilation cycle. We, thus, need to provide a way to compute preimages, i.e. obtain the corresponding facies fields of the update results. However, except for very special cases, the substantial difference in dimensionality between the parameter and feature spaces makes this generally impossible (see Schölkopf, Mika, Burges et al. 1999, for an argument regarding the Gaussian kernel). Fortunately, the polynomial kernel (3.2.4) with odd degree is one such special case, but the same does not hold for (3.2.16).

To overcome this issue, the literature provides methods to compute approximate solutions. These involve, either solving an optimization problem via fixed-point iterations (Schölkopf, Mika, Burges et al. 1999), multidimensional scaling based on distances (Kwok and Tsang 2004) or learning the inverse mapping by constructing a basis in the feature space (Honeine and Richard 2011a). A complete survey of these methods can be consulted in (Honeine and Richard 2011b).

In the following paragraphs, we will review the fixed-point iterative scheme used to compute

preimages in (Sarma, Durlofsky et al. 2008), and propose an alternative kernel function, which admits analytical solutions to the preimage problem. An extensive comparison between the two methods will be conducted in paragraph 5.1.

3.5.1. Fixed-point iterative scheme

Since the inverse kernel PCA problem does not have an exact solution in the general case, Schölkopf, Mika, Burges et al. (1999) propose a method to compute approximate preimages. Let $\boldsymbol{\Phi}_{\mathbf{x}} \in F$ be the image corresponding to a given projection vector $\boldsymbol{\xi} \in \mathbb{R}^l$ (for example, obtained through data assimilation). Then we can formulate the following optimization problem,

$$\begin{aligned} &\underset{\mathbf{x}\in\mathbb{R}^m}{\operatorname{argmin}}\, \rho(\mathbf{x}) \\ &\rho(\mathbf{x}) = \left\|\Phi(\mathbf{x}) - \boldsymbol{\Phi}_{\mathbf{x}}\right\|_F^{\,2} \\ &\qquad = \langle \Phi(\mathbf{x}), \Phi(\mathbf{x})\rangle_F - 2\langle \Phi(\mathbf{x}), \boldsymbol{\Phi}_{\mathbf{x}}\rangle_F + \langle \boldsymbol{\Phi}_{\mathbf{x}}, \boldsymbol{\Phi}_{\mathbf{x}}\rangle_F \end{aligned} \qquad \boldsymbol{\Phi}_{\mathbf{x}} \in F \text{ fixed} \tag{3.5.1}$$

A key result is the following

Proposition 3.4: Linearity of the image space

Any image, $\Phi(\mathbf{x})$, lies in the feature subspace spanned by the images of the training dataset, $\Phi(\mathbf{t}_i)$, i.e.

$$\Phi(\mathbf{x}) = \sum_{i=1}^{n_t} \boldsymbol{\gamma}[i] \cdot \Phi(\mathbf{t}_i) \qquad \forall \mathbf{x} \in \mathbb{R}^m \tag{3.5.2}$$

Its proof will be provided in the next paragraph. We will not use the expression for the $\boldsymbol{\gamma}[i]$ from (Sarma, Durlofsky et al. 2008), as their derivation assumed that the projection vectors, $\boldsymbol{\xi} \in \mathbb{R}^l$, are normally distributed with zero mean and covariance equal to $diag(\lambda_1, \ldots, \lambda_l)$. This assumption was contested by Ma and Zabaras (2011), who attempt to use polynomial chaos expansions in order to approximate the true distributions.

We can now rewrite the optimization problem (3.5.1) as

$$\rho(\mathbf{x}) \overset{(3.5.2)}{=\!=} \mathfrak{K}(\mathbf{x},\mathbf{x}) - 2\sum_{i=1}^{n_t} \boldsymbol{\gamma}[i] \cdot \mathfrak{K}(\mathbf{t}_i, \mathbf{x}) + \sum_{i=1}^{n_t}\sum_{j=1}^{n_t} \boldsymbol{\gamma}[i] \cdot \boldsymbol{\gamma}[j] \cdot \mathfrak{K}(\mathbf{t}_i, \mathbf{t}_j) \quad \boldsymbol{\Phi}_{\mathbf{x}} \in F \text{ fixed} \tag{3.5.3}$$

which can be solved by setting the gradient of the objective function to $\mathbf{0}$,

$$\frac{d\rho}{d\mathbf{x}} = \frac{d\mathfrak{K}(\mathbf{x},\mathbf{x})}{d\mathbf{x}} - 2\sum_{i=1}^{n_t} \frac{d\mathfrak{K}(\mathbf{t}_i,\mathbf{x})}{d\mathbf{x}} = \mathbf{0} \tag{3.5.4}$$

Applying the above to the kernel in (3.2.16), Sarma, Durlofsky et al. (2008) developed the following fixed-point iterative scheme.

$$\mathbf{x}^k = \frac{\sum_{i=1}^{n_t} \boldsymbol{\gamma}[i] \left(\sum_{j=1}^{d} j \left\langle \mathbf{t}_i, \mathbf{x}^{k-1}\right\rangle_2^{\,j-1}\right) \mathbf{t}_i}{\sum_{i=1}^{n_t} \boldsymbol{\gamma}[i] \left(\sum_{j=1}^{d} j \left\langle \mathbf{t}_i, \mathbf{x}^{k-1}\right\rangle_2^{\,j-1}\right)} \qquad k \in \mathbb{N}^* \tag{3.5.5}$$

The starting solution, $\mathbf{x}^0$, dictates the region of the parameter space where the search is performed, therefore, a poor choice might drive the result into a local optimum (Kwok and Tsang 2004). In the next paragraph we will give an alternative solution that does not suffer from this drawback.

3.5.2. Analytical method

The derivation in this paragraph relies heavily on the following

Proposition 3.5: Sufficient condition for the existence of a preimage

(Schölkopf, Mika, Smola et al. 1998) A feature space admits an exact solution to the preimage problem if its kernel is an invertible function of the dot product.

Proof. Consider the canonical basis in the parameter space

$$\mathbf{e}_i[j] = \delta_{ij} \qquad \forall i, j = 1, \dots, m \tag{3.5.6}$$

Then, the coordinates of any datapoint can be written as dot products with the canonical vectors $\mathbf{e}_i$

$$\mathbf{x} = \sum_{i=1}^{m} \langle \mathbf{x}, \mathbf{e}_i \rangle_2 \cdot \mathbf{e}_i \qquad \forall \mathbf{x} \in \mathbb{R}^m \tag{3.5.7}$$

and, if the kernel function satisfies

$$\mathfrak{K}(\mathbf{x}, \mathbf{y}) = f\left(\langle \mathbf{x}, \mathbf{y} \rangle_2\right) \qquad \forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^m \tag{3.5.8}$$

with $f : \mathbb{R} \to \mathbb{R}$ invertible, then we can write

$$\begin{aligned} \mathbf{x}[i] &= \langle \mathbf{x}, \mathbf{e}_i \rangle_2 \\ &= f^{-1}\left(\mathfrak{K}(\mathbf{x}, \mathbf{e}_i)\right) \\ &= f^{-1}\left(\langle \Phi(\mathbf{x}), \Phi(\mathbf{e}_i) \rangle_F\right) \end{aligned} \qquad \forall \mathbf{x} \in \mathbb{R}^m \tag{3.5.9}$$

□

Even though this is a known result, the works consulted in our literature study do not provide the full derivation of a closed-form expression for the analytical preimage solution. Therefore, we feel compelled to elaborate below.

The key lies in the, yet undefended, result from proposition 3.4. We begin its

Proof of Proposition 3.4.
by expanding (3.4.3) as

$$\begin{aligned} \mathbf{v}_j &= \sum_{i=1}^{n_t} \boldsymbol{\alpha}_j[i] \cdot \widetilde{\Phi}(\mathbf{t}_i) \\ &= \sum_{i=1}^{n_t} \boldsymbol{\alpha}_j[i] \cdot \left(\Phi(\mathbf{t}_i) - \overline{\Phi_{\mathbf{t}}}\right) \\ &= \left(\sum_{i=1}^{n_t} \boldsymbol{\alpha}_j[i] \cdot \Phi(\mathbf{t}_i)\right) - \left(\sum_{i=1}^{n_t} \boldsymbol{\alpha}_j[i]\right) \cdot \overline{\Phi_{\mathbf{t}}} \qquad \forall j = 1, \dots, l \\ &= \left(\sum_{i=1}^{n_t} \boldsymbol{\alpha}_j[i] \cdot \Phi(\mathbf{t}_i)\right) - \sum_{k=1}^{n_t} \frac{1}{n_t} \left(\sum_{i=1}^{n_t} \boldsymbol{\alpha}_j[i]\right) \cdot \Phi(\mathbf{t}_k) \\ &= \sum_{i=1}^{n_t} \left(\boldsymbol{\alpha}_j[i] - \overline{\boldsymbol{\alpha}_j}\right) \cdot \Phi(\mathbf{t}_i) \end{aligned} \tag{3.5.10}$$

where $\overline{\boldsymbol{\alpha}}_j$ is the mean of the elements in vector $\boldsymbol{\alpha}_j$.

This allows us to write

$$\begin{aligned}\Phi(\mathbf{x}) &= \overline{\Phi_{\mathbf{t}}} + \widetilde{\Phi}(\mathbf{x}) \\ &\overset{(3.4.7)}{=\!=} \overline{\Phi_{\mathbf{t}}} + \sum_{j=1}^{l} \boldsymbol{\xi}[j] \cdot \mathbf{v}_j \\ &\overset{(3.5.10)}{=\!=} \overline{\Phi_{\mathbf{t}}} + \sum_{j=1}^{l} \boldsymbol{\xi}[j] \left(\sum_{i=1}^{n_t} (\boldsymbol{\alpha}_j[i] - \overline{\boldsymbol{\alpha}_j}) \cdot \Phi(\mathbf{t}_i) \right) \qquad \forall \mathbf{x} \in \mathbb{R}^m \\ &= \overline{\Phi_{\mathbf{t}}} + \sum_{i=1}^{n_t} \left(\sum_{j=1}^{l} (\boldsymbol{\xi}[j] \cdot \boldsymbol{\alpha}_j[i] - \boldsymbol{\xi}[j] \cdot \overline{\boldsymbol{\alpha}_j}) \right) \cdot \Phi(\mathbf{t}_i)\end{aligned} \tag{3.5.11}$$

where we assumed equality in (3.4.7), since the contributions of the $(p-l)$ trailing components are attributed to noise and, thus, discarded by the kernel PCA algorithm.

For simplicity, let's denote

$$\boldsymbol{\beta}[i] = \sum_{j=1}^{l} \boldsymbol{\xi}[j] \cdot \boldsymbol{\alpha}_j[i] \qquad \forall i = 1, \ldots, n_t \tag{3.5.12}$$

Then,

$$\begin{aligned}\sum_{j=1}^{l} \boldsymbol{\xi}[j] \cdot \overline{\boldsymbol{\alpha}_j} &= \sum_{j=1}^{l} \boldsymbol{\xi}[j] \left(\frac{1}{n_t} \sum_{i=1}^{n_t} \boldsymbol{\alpha}_j[i] \right) \\ &= \frac{1}{n_t} \sum_{j=1}^{l} \sum_{i=1}^{n_t} \boldsymbol{\xi}[j] \cdot \boldsymbol{\alpha}_j[i] \\ &= \frac{1}{n_t} \sum_{i=1}^{n_t} \left(\sum_{j=1}^{l} \boldsymbol{\xi}[j] \cdot \boldsymbol{\alpha}_j[i] \right) \\ &\overset{(3.5.12)}{=\!=} \frac{1}{n_t} \sum_{i=1}^{n_t} \boldsymbol{\beta}[i] \\ &= \overline{\boldsymbol{\beta}}\end{aligned} \tag{3.5.13}$$

leading to

$$\begin{aligned}\Phi(\mathbf{x}) &= \overline{\Phi_{\mathbf{t}}} + \sum_{i=1}^{n_t} \left(\boldsymbol{\beta}[i] - \overline{\boldsymbol{\beta}} \right) \Phi(\mathbf{t}_i) \\ &\overset{(3.2.8)}{=\!=} \left(\frac{1}{n_t} \sum_{i=1}^{n_t} \Phi(\mathbf{t}_i) \right) + \sum_{i=1}^{n_t} \left(\boldsymbol{\beta}[i] - \overline{\boldsymbol{\beta}} \right) \Phi(\mathbf{t}_i) \qquad \forall \mathbf{x} \in \mathbb{R}^m \\ &= \sum_{i=1}^{n_t} \left(\frac{1}{n_t} + \boldsymbol{\beta}[i] - \overline{\boldsymbol{\beta}} \right) \Phi(\mathbf{t}_i)\end{aligned} \tag{3.5.14}$$

Thus, any $\Phi(\mathbf{x})$ lies in the linear span of the images of the training dataset, with coefficients

$$\boldsymbol{\gamma}[i] = \frac{1}{n_t} + \boldsymbol{\beta}[i] - \overline{\boldsymbol{\beta}} \qquad \forall i = 1, \ldots, n_t \tag{3.5.15}$$

□

We now have all the necessary ingredients. By combining (3.5.2) and (3.5.9), the preimage solution, finally, reads

$$\begin{aligned}\mathbf{x}[i] &= f^{-1}\left(\sum_{j=1}^{n_t}\boldsymbol{\gamma}[j]\cdot\langle\Phi(\mathbf{t}_j),\Phi(\mathbf{e}_i)\rangle_F\right)\\ &= f^{-1}\left(\sum_{j=1}^{n_t}\boldsymbol{\gamma}[j]\cdot\mathfrak{K}(\mathbf{t}_j,\mathbf{e}_i)\right)\end{aligned}\qquad\forall\mathbf{x}\in\mathbb{R}^m \tag{3.5.16}$$

The kernel function (3.2.16), used by Sarma, Durlofsky et al. (2008) to preserve multipoint geostatistics, is a d-th degree polynomial of the dot product,

$$f_{1\dots d}(s)=\sum_{i=1}^{d}s^i=\frac{1-s^{d+1}}{1-s}-1=\frac{s(1-s^d)}{1-s}\qquad\forall s\in\mathbb{R} \tag{3.5.17}$$

where we used the finite geometric series formula. Unfortunately, $f_{1\dots d}$ is not invertible, and we propose the following alternative kernel – a slightly modified version of the *inhomogeneous polynomial kernel* (Schölkopf and Smola 2003)

$$\begin{aligned}\mathfrak{K}_{1\dots d}{}^*(\mathbf{x},\mathbf{y}) &= (\langle\mathbf{x},\mathbf{y}\rangle_2+1)^d-1\\ &= \sum_{i=1}^{d}\binom{d}{i}\langle\mathbf{x},\mathbf{y}\rangle_2{}^i\end{aligned}\qquad\forall\mathbf{x},\mathbf{y}\in\mathbb{R}^m\quad d\in\mathbb{N}^*\text{ odd} \tag{3.5.18}$$

which corresponds to

$$\begin{aligned}f_{1\dots d}{}^*(s) &= (s+1)^d-1\\ f_{1\dots d}{}^{*-1}(s) &= \sqrt[d]{s+1}-1\end{aligned}\qquad\forall s\in\mathbb{R} \tag{3.5.19}$$

If the order, d, is chosen as an even number, then the inverse function will attain complex values for negative dot products, hence the above holds only for odd d. This does not impede our application significantly, however, the same cannot be immediately said about the extra coefficients, $\binom{d}{i}$, that differentiate (3.5.18) from (3.2.16). Their effect was studied extensively in the experiments from paragraph 5.1 and their results allow us to conclude that the analytical method is preferable over the fixed-point iterative scheme, in general. Thus, for the remainder of this report, we will use analytical preimages for our parameterizations.

3.6. KPCA-EnKF

Equations (3.4.9) and (3.5.16) allow us to integrate the polynomial kernel PCA transform into the parameterized EnKF workflow (figure 3.1) by defining $g : \mathbb{R}^m \to \mathbb{R}^l$ as

$$\begin{aligned} g(\mathbf{x}) &= W^T \widetilde{\mathbf{k}} && \forall \mathbf{x} \in \mathbb{R}^m \\ g^{-1}(\boldsymbol{\xi}) &= f^{-1}(K_t \boldsymbol{\gamma}) && \forall \boldsymbol{\xi} \in \mathbb{R}^l \end{aligned} \tag{3.6.1}$$

where f^{-1} is applied component-wise and

$$\begin{aligned} &\widetilde{\mathbf{k}}[j] = \widetilde{\mathfrak{K}}(\mathbf{t}_j, \mathbf{x}) && \forall i = 1, \ldots, m \\ &K_t[i,j] = \mathfrak{K}(\mathbf{e}_i, \mathbf{t}_j) = f(\mathbf{t}_j[i]) && \forall j = 1, \ldots, n_t \\ &\boldsymbol{\gamma} = \mathbf{1}_{\mathbf{n_t}} + (I_{n_t} - 1_{n_t}) W \boldsymbol{\xi} \end{aligned} \tag{3.6.2}$$

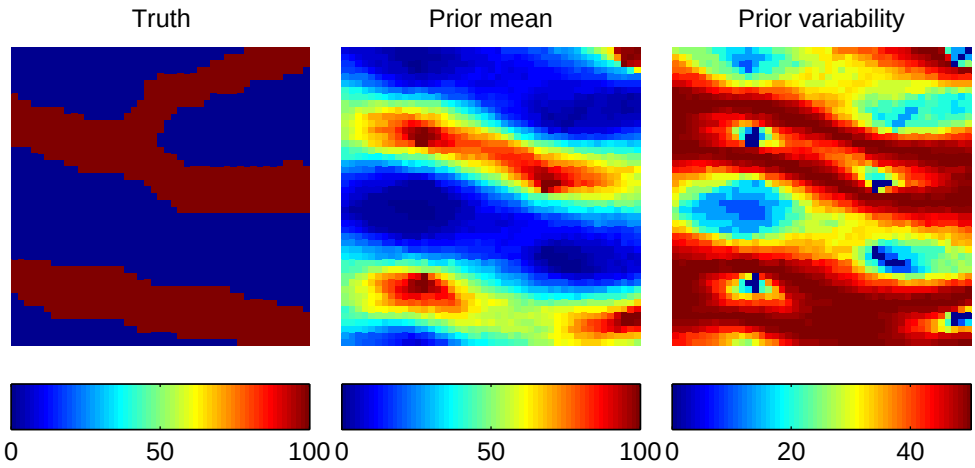
For comparison, we will now re-run the *Y-channel reservoir* experiment (see table 2.2) using the parameterization setup described in table 3.1.

Item	**Description**
Training set size	$n_t = 1500$ samples
Kernel degree	$d \in \{1, 3, 5\}$
Parameterization variance threshold	$q = 90\%$
Adaptation method	*logit*

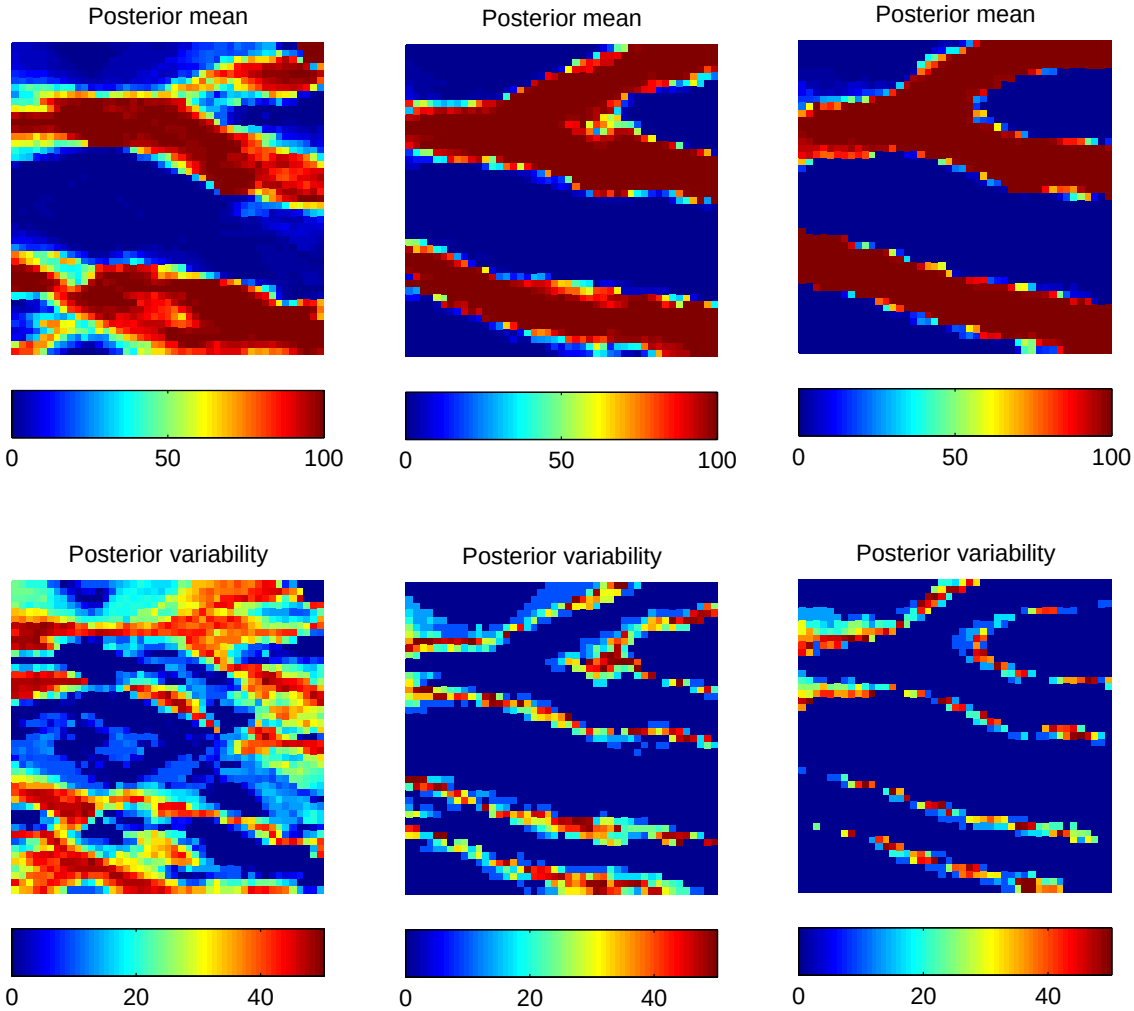
Table 3.1: Parmeterization setup for the example reservoir

The results are depicted in figure 3.5, and we see that, for kernel degree 1 (figure 3.5b), we obtain a posterior mean similar to that of the classic EnKF (figure 2.9). This was to be expected, since, for $d = 1$, the kernel function computes the regular dot product and, thus, the feature and parameter spaces coincide. However, as the degree of the kernel increases, so do the curved features start to develop (see figure 3.5d).

Unfortunately, the better accuracy is accompanied by a dramatic decrease in ensemble variability (see the bottom plots in figure 3.5). This is called *ensemble collapse* and will be studied in more detail during the following chapter.



(a) True permeability field and prior information



(b) $d = 1$ (c) $d = 3$ (d) $d = 5$

Figure 3.5: KPCA-EnKF results with different degrees of the polynomial kernel function

Chapter 4

Preventing ensemble collapse

This chapter explores the possibilities to avoid the issue of ensemble collapse, which appears when using the kernel PCA parameterization in EnKF. In (Evensen 2009, Chapter 15), the author points out that the main cause for the dramatic loss in variability is the use of a limited number of ensemble members when building the error covariance matrix.

Specifically, a small ensemble size results in a coarse approximation of P^f and the appearance of artificial (or spurious) correlations between the state variables and the observations. As a result, each update step produces an over-reduction of the ensemble variability. In dealing with this issue, Evensen (2009) discusses methods, such as localization or inflation, which imply direct modifications on the covariance matrix.

However, since we are using a nonlinear parameterization, it is not obvious how to adapt these methods to our application (in the case of localization, this might even turn out to be impossible). Instead, we will focus on two recent developments – the Ensemble Smoother with multiple data assimilations (Emerick and Reynolds 2012) and the Subspace EnKF (Sarma and W. Chen 2013).

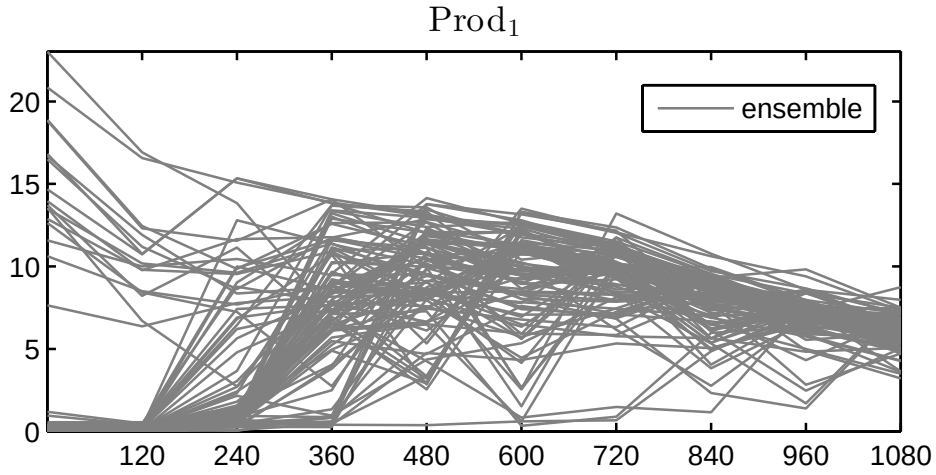
4.1. Ensemble smoother

In order to learn more about the apparition and development of ensemble collapse, we will investigate some additional data from the previous experiment. Figure 4.1 illustrates the convergence behaviour of the KPCA-EnKF by plotting the forecasted oil rate in $Prod_1$ for all members after each update.

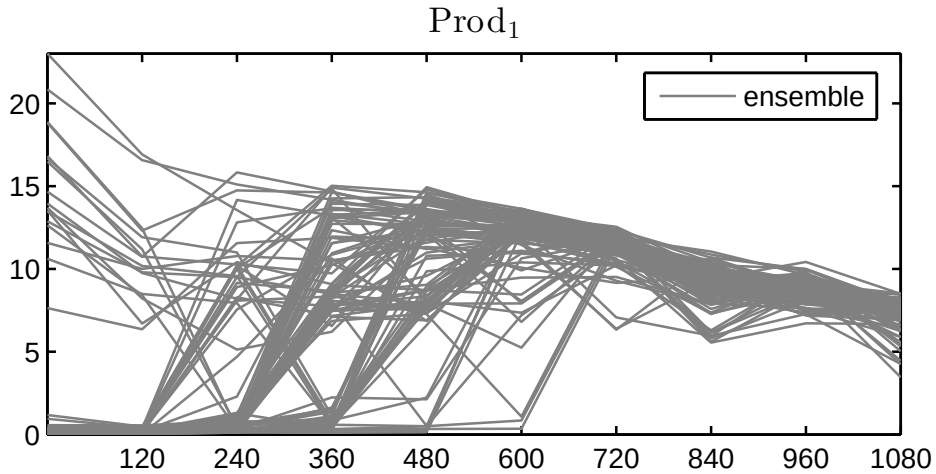
We notice that for $d = 1$ (figure 4.1a) the ensemble maintains its spread until the end of the simulation. In contrast, for the cases when $d = 3$ and $d = 5$ (figures 4.1b and 4.1c), where we experienced collapse, there is an abrupt decrease in spread after the 6th-7th update and, even though new observations become available at subsequent time steps, they seem to have little impact.

The Ensemble Smoother (EnS) is a data assimilation algorithm proposed by van Leeuwen and Evensen (1996). In its formulation, the EnKF's sequential updates are replaced with a single one, which uses all available observations at once. This is, essentially, done by pasting together the state vectors at different times, on one side, and the corresponding innovation vectors, on the other, and then plugging them into the Kalman update equation (2.3.6). Recently, Emerick and Reynolds (2012) extended this method by allowing for multiple iterations. More specifically, they prove that one Kalman update is equivalent to n sequential iterations of (2.3.6) using the same observations, while replacing R with nR.

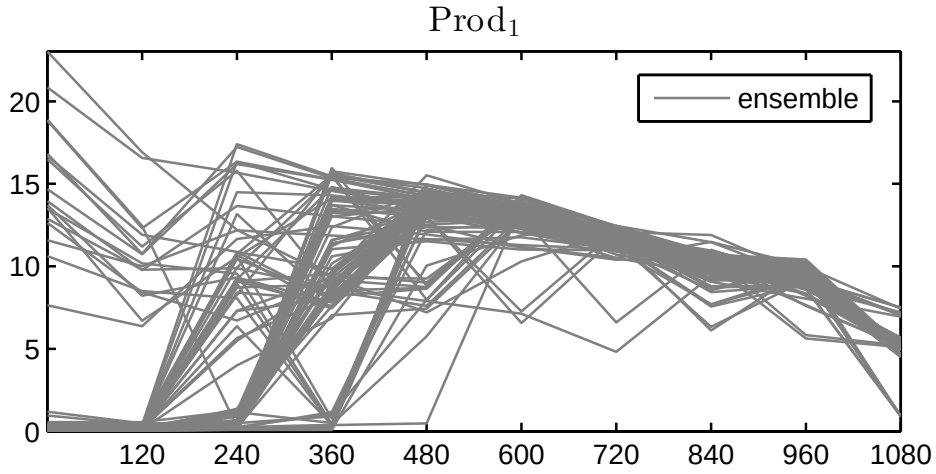
We will now test this Iterative EnS algorithm, together with the KPCA parameterization (table 3.1), in the hope of preventing collapse on the *Y-channel reservoir*. For this purpose, we used a number of 9 iterations, which is equal to the number of sequential updates from the previous experiment (table 2.2). The results are shown in figure 4.2 and, indeed, we notice an increase in variability around the channel boundary, for both $d = 3$ (figure 4.2b) and $d = 5$ (figure 4.2c).



(a) $d = 1$

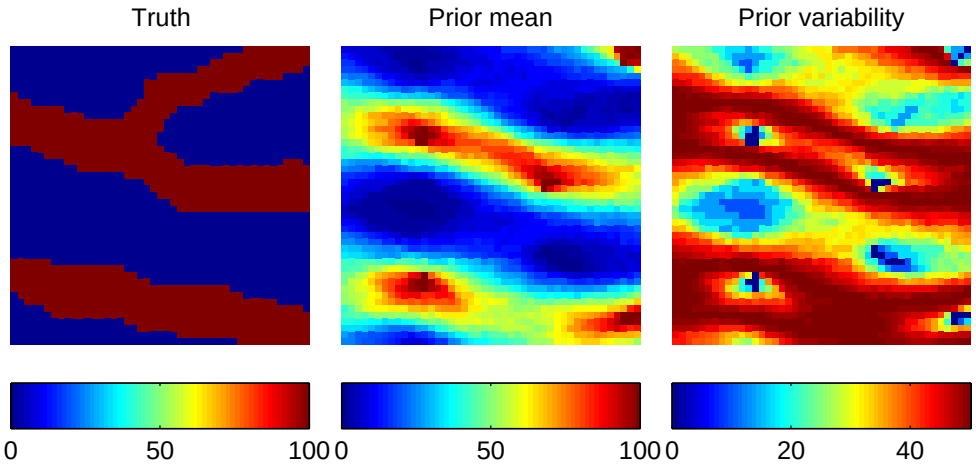


(b) $d = 3$

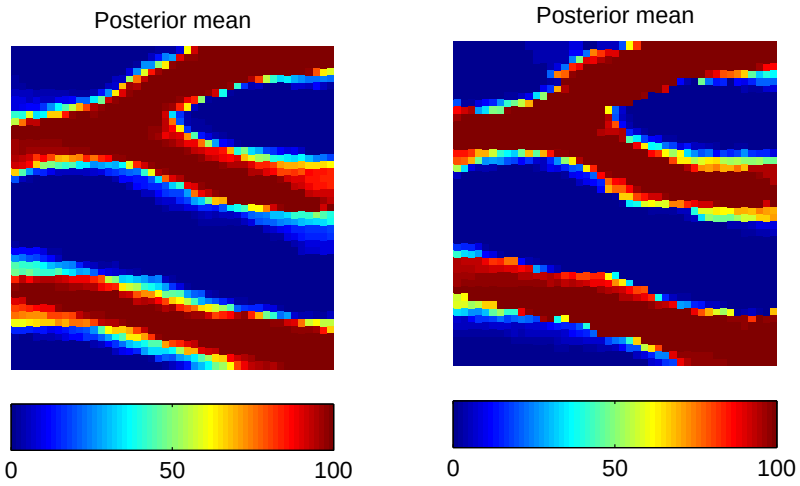


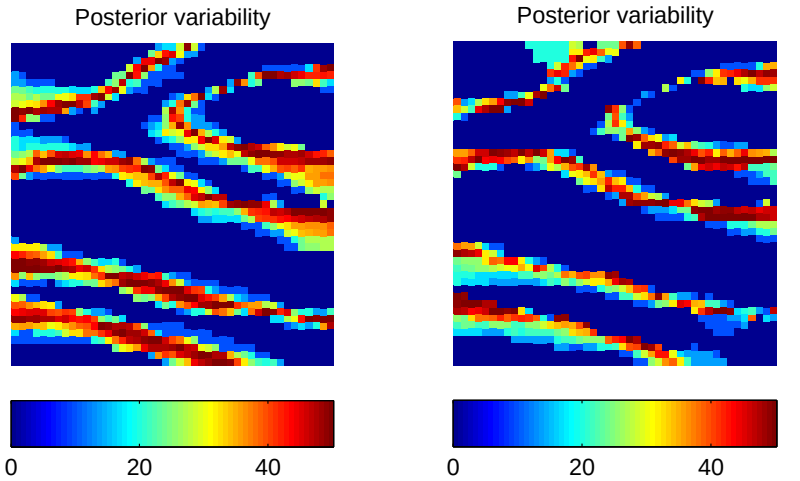
(c) $d = 5$

Figure 4.1: Convergence behaviour of the KPCA-EnKF with different kernel degrees (oil rate in $Prod_1$)



(a) True permeability field and prior information





(b) $d = 3$ (c) $d = 5$

Figure 4.2: Iterative EnS results with different degrees of the polynomial kernel function

Encouraged by these results, we will prefer the Iterative KPCA-EnS over the KPCA-EnKF in the comparative experiments conducted in paragraph 5.2. However, having variability only around channel boundaries might not be sufficient if the general structure is not in line with the observations. This is the reason why, in the next paragraph, we will study an alternative approach in preventing collapse.

4.2. Subspace EnKF

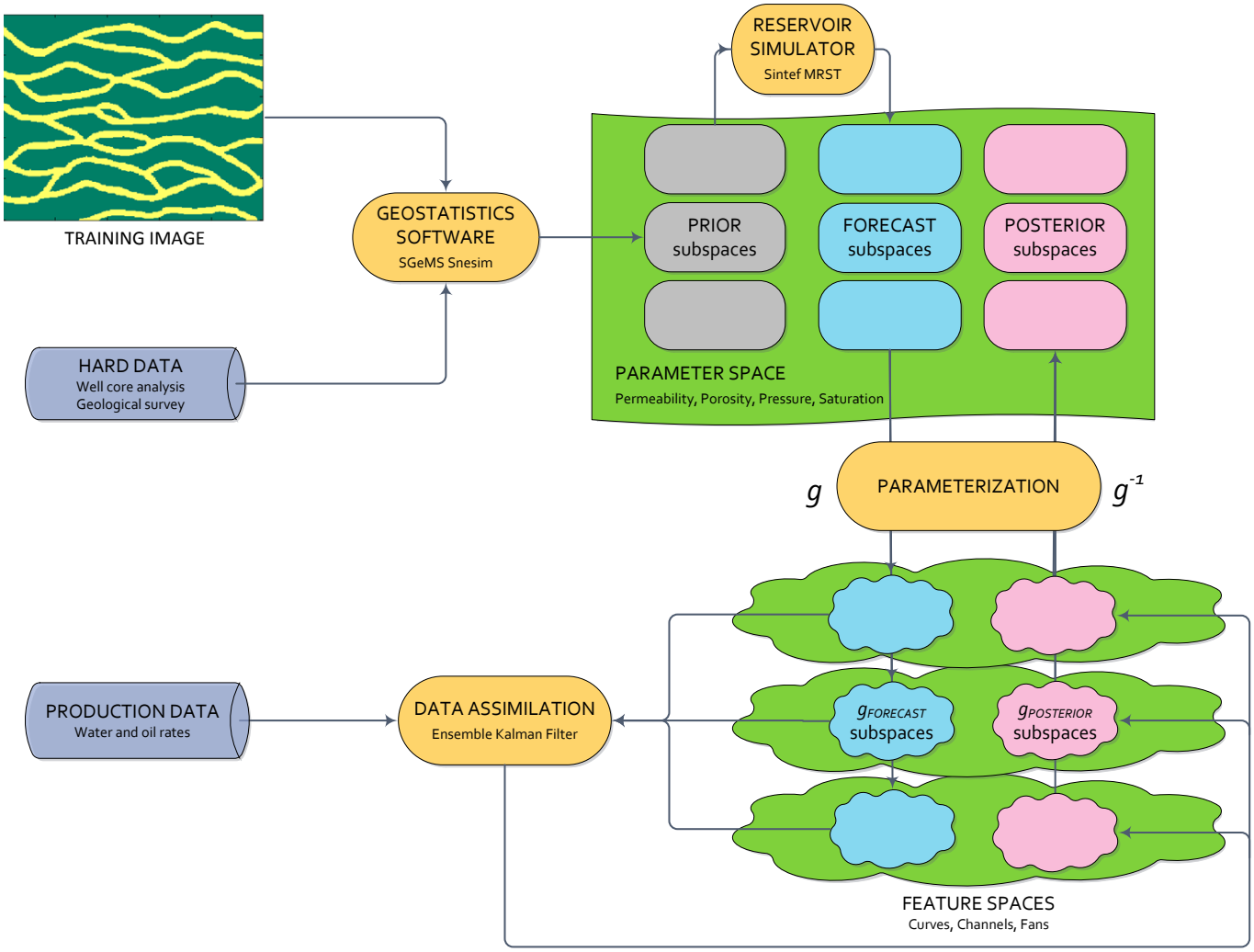


Figure 4.3: Subspace EnKF workflow

Regardless of which history matching method we choose, the objective is to obtain a posterior that minimizes the MSE function (see paragraph 2.3.2). Let's denote by $h = \text{tr}(P)$, with $\nabla_{\mathbf{x}} h$ its gradient vector. Then, we can approximate the solution to $\text{argmin}_{\mathbf{x}}\, h$ by employing the steepest descent iterative method, introduced by Cauchy (1847) and refined by Curry (1944),

$$\mathbf{x}^k = \mathbf{x}^{k-1} - s \cdot \nabla_{\mathbf{x}} h \qquad s > 0, \quad k \in \mathbb{N}^* \tag{4.2.1}$$

Sarma and W. Chen (2013) observed a similarity between this equation, with step size $s = 1$, and the Kalman update (2.3.6), i.e. they used the following

Conjecture 4.1: Equivalence between the EnKF and the steepest descent method

The EnKF update is equivalent to an iteration of the steepest descent method with unit step, where the gradient is approximated from the ensemble, i.e.

$$K_{gain}\mathbf{\Delta d} \simeq -\mathbf{\nabla_x} h \tag{4.2.2}$$

to develop a new data assimilation framework, called the *Subspace EnKF*. However, this fact was not sufficiently confirmed during our literature study (the closest match was the result by Sayed and Kailath (1994), showing the equivalence between the Kalman filter and the Recursive Least Squares method) and, unfortunately, providing a proof is outside the scope of this thesis. Hence, we will use the above as an assumption.

As illustrated in figure 4.3, the attractiveness of the Subspace EnKF lies in the ability to define a different parameterization for each (group of) ensemble members. The assimilation results will, therefore, lie in the region of the parameter space spanned by the preimages of the corresponding parameterization. Suppose these subspaces are known to be disjunct, then collapse is guaranteed not to occur, regardless of the volume of observations or the number of assimilation steps.

However, since the Kalman equations (2.3.6) are only applicable when all members originate from the same space, a naive approach would be to partition the ensemble into groups and run a separate Kalman filter for each of them. Still, the reduced size of each group would increase the ill-posedness of the problem, and instead, for the Subspace EnKF, the Kalman gain is computed in such a way as to retain information from the complete ensemble.

To see this, we will turn our attention to the parameterized Kalman update equation (3.1.1), in which the Kalman gain is computed using the projection vectors, $\boldsymbol{\xi}$. Hence, the objective function is expressed in terms of the covariance of the feature space images, however, since we want an update that takes into account all ensemble members, regardless of parameterization, we have to reconsider and choose the K_{gain} that minimizes h in the parameter space. Then, the Subspace EnKF update equation can be formulated as

$$\begin{aligned}
\boldsymbol{\xi}^a &\overset{(4.2.2)}{=\!=} \boldsymbol{\xi}^f - \mathbf{\nabla}_{\boldsymbol{\xi}} h \\
&\overset{chain}{=\!=} \boldsymbol{\xi}^f - \left(J_{\boldsymbol{\xi}}\mathbf{x}\right)^T \mathbf{\nabla_x} h \\
&\overset{(4.2.2)}{=\!=} \boldsymbol{\xi}^f + \left(J_{\boldsymbol{\xi}}\mathbf{x}\right)^T K_{gain}\mathbf{\Delta d}
\end{aligned} \tag{4.2.3}$$

where $J_{\boldsymbol{\xi}}\mathbf{x}\ [j,k] = \dfrac{\partial \mathbf{x}[j]}{\partial \boldsymbol{\xi}[k]}$ is the jacobian matrix and we used the *chain rule*,

$$\mathbf{\nabla}_{\boldsymbol{\xi}} h\ [k] = \frac{\partial h}{\partial \boldsymbol{\xi}[k]} \overset{chain}{=\!=} \sum_{j=1}^{m} \frac{\partial h}{\partial \mathbf{x}[j]} \cdot \frac{\partial \mathbf{x}[j]}{\partial \boldsymbol{\xi}[k]} \qquad \forall k = 1, \ldots, l \tag{4.2.4}$$

What remains is to derive the expression for $J_{\boldsymbol{\xi}}\mathbf{x}$. In the case of kernel PCA, this is

$$\begin{aligned}
J_{\boldsymbol{\xi}}\mathbf{x}\,[j,k] &= \frac{\partial \mathbf{x}[j]}{\partial \boldsymbol{\xi}[k]} \\
&\overset{(3.5.9)}{=\!=} \frac{\partial f^{-1}\left(\langle \Phi(\mathbf{x}), \Phi(\mathbf{e}_j)\rangle_F\right)}{\partial \boldsymbol{\xi}[k]} \\
&\overset{chain}{=\!=} f^{-1'}\left(\langle \Phi(\mathbf{x}), \Phi(\mathbf{e}_j)\rangle_F\right) \cdot \frac{\partial \langle \Phi(\mathbf{x}), \Phi(\mathbf{e}_j)\rangle_F}{\partial \boldsymbol{\xi}[k]} \\
&\overset{(3.5.2)}{=\!=} f^{-1'}\left(\sum_{i=1}^{n_t} \boldsymbol{\gamma}[i] \cdot \mathfrak{K}(\mathbf{t}_i, \mathbf{e}_j)\right) \cdot \frac{\partial \langle \Phi(\mathbf{x}), \Phi(\mathbf{e}_j)\rangle_F}{\partial \boldsymbol{\xi}[k]} \\
&\overset{(3.4.7)}{=\!=} f^{-1'}\left(\sum_{i=1}^{n_t} \boldsymbol{\gamma}[i] \cdot \mathfrak{K}(\mathbf{t}_i, \mathbf{e}_j)\right) \cdot \langle \mathbf{v}_k, \Phi(\mathbf{e}_j)\rangle_F \\
&\overset{(3.5.10)}{=\!=} f^{-1'}\left(\sum_{i=1}^{n_t} \boldsymbol{\gamma}[i] \cdot \mathfrak{K}(\mathbf{t}_i, \mathbf{e}_j)\right) \cdot \left(\sum_{i=1}^{n_t} (\boldsymbol{\alpha}_k[i] - \overline{\boldsymbol{\alpha}_k}) \cdot \mathfrak{K}(\mathbf{t}_i, \mathbf{e}_j)\right) \\
&= f^{-1'}\left(\sum_{i=1}^{n_t} \boldsymbol{\gamma}[i] \cdot f\left(\mathbf{t}_i[j]\right)\right) \cdot \left(\sum_{i=1}^{n_t} (\boldsymbol{\alpha}_k[i] - \overline{\boldsymbol{\alpha}_k}) \cdot f\left(\mathbf{t}_i[j]\right)\right)
\end{aligned} \tag{4.2.5}$$

where the $\mathbf{t}_i$ represent the training dataset.

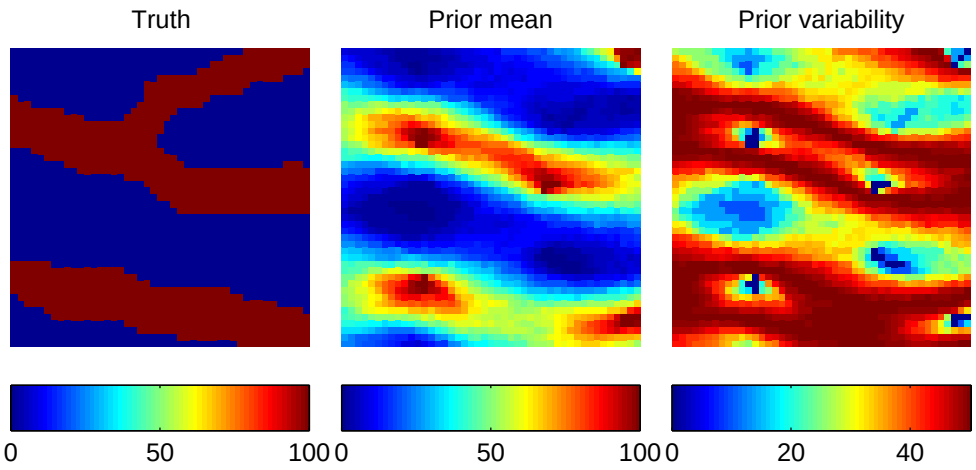
For the kernel (3.5.18), we have

$$f_{1\ldots d}{}^{*-1'}(s) = \frac{1}{d\sqrt[d]{(s+1)^{d-1}}} \tag{4.2.6}$$

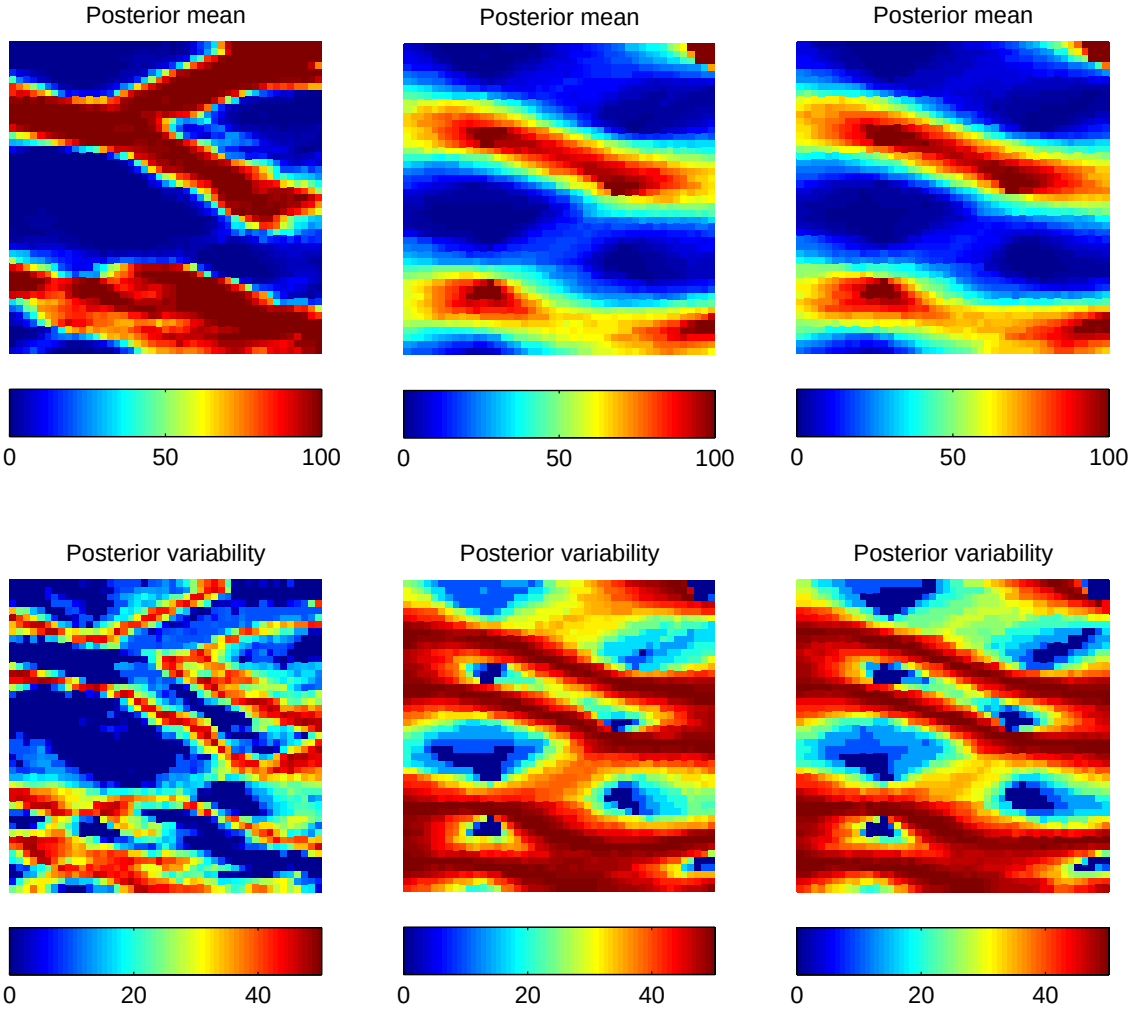
For comparison, we will now use the Subspace EnKF to history match the *Y-channel reservoir* (see table 2.2) with 5 subspaces and different values for the kernel degree. The results are illustrated in figure 4.4 and we notice that, for $d = 1$ (figure 4.4b), we obtain a channelized structure with a good amount of posterior variability, covering not only the shale-sand borders, but also the channel body and its possible branches (see the bottom part of the plot).

It is, therefore, surprising that these properties are not inherited for $d = 3$ (figure 4.4c) and $d = 5$ (figure 4.4d). Actually, when comparing the results to the prior ensemble (figure 4.4a), the update seems to have little to no effect in these cases. While investigating this issue, we identified that the culprit is the jacobian matrix, $J_{\boldsymbol{\xi}}\mathbf{x}$, its norm decreasing drastically as d increases. As a result, the magnitude of the updates become insignificant for $d > 1$, leaving the prior ensemble almost intact. This is not in line with the results obtained by Sarma and W. Chen (2013) and the only difference is that, in the latter, the authors apply PCA to the ensemble before using the kernel parameterization, a step which we chose to avoid, in order to better understand the sole effect of kernel PCA.

Despite this evident drawback, the results obtained for $d = 1$ are promising, and we will investigate the Subspace EnKF more thoroughly in chapter 5.



(a) True permeability field and prior information



(b) $d = 1$ (c) $d = 3$ (d) $d = 5$

Figure 4.4: Subspace EnKF results with different degrees of the polynomial kernel function

Chapter 5

Experiments

This chapter details the experiments carried out during our study and discusses their results. As can be consulted in table 5.1, we used a multi-core hardware platform running MATLAB and its Parallel Toolbox. The starting point for our implementation was the data assimilation module, developed at TNO (Leeuwenburgh 2012), for the MRST reservoir simulator (see https://www.sintef.no/Projectweb/MRST/ and http://www.sintef.no/Projectweb/MRST/Modules/). We customized the module code to the application at hand and introduced parallelization for more efficiency. The KPCA training sets and the initial ensemble for the EnKF were generated using the snesim algorithm and the image from figure 2.6. Snesim is part of the SGeMS software suite (http://sgems.sourceforge.net/), which was interfaced with MATLAB using the mGstat package (http://mgstat.sourceforge.net/).

Item	Specification
Processor	Intel Core i7 2670QM
# of cores	4
RAM	8 GB, 1333 MHz
Operating system	Windows 8 64-bit
MATLAB version	R2013a 64-bit
MRST version	2013a
SGeMS version	v2.5b
mGstat version	0.991

Table 5.1: Hardware and software setup for the experiments

All results presented in this chapter were obtained from so-called *twin-experiments*, in which the "truth" state is known and used to simulate the observation process. This allows for both a qualitative and a quantitative comparison between canonical methods (the EnKF) and those newly introduced in the literature (KPCA-EnS and Subspace EnKF), with the modifications suggested in the previous chapters.

The exposition is split into three parts. First, we focus on the polynomial kernel PCA parameterization and begin by comparing the results of the fixed-point iterative scheme, used by Sarma, Durlofsky et al. (2008), and the analytical method introduced in paragraph 3.5.2. We also study their sensitivity on the size of the training set and of the reservoir grid, and investigate their behaviour when paired with the logistic transform (paragraph 2.3.4).

Next, we perform a comparison between the performance of the classic EnKF, the KPCA-EnS with $d = 3$ and the Subspace EnKF with $d = 1$ on two channelized reservoirs. Finally, we conduct sensitivity studies on the Subspace EnKF.

5.1. Study on the polynomial kernel parameterization

The experiments in this paragraph aim to differentiate between the two preimage solutions for the polynomial kernel PCA parameterization – the fixed-point iterative scheme, used by Sarma, Durlofsky et al. (2008), and the analytical method, developed in paragraph 3.5.2. To this end, we designed the following twin-experiment

Algorithm 5.1: Preimage problem experiment

Input: A set of $n = (n_t + n_e)$ channelized fields, generated using snesim.

Output: The fields obtained from the n_e test samples after an image-preimage cycle.

① Use the first n_t samples to train the parameterization (i,e. obtain the transformation matrix using algorithm 3.2).

② Apply the parameterization on the n_e test samples, $\mathbf{x}_i$, in order to obtain the projection vectors, $\boldsymbol{\xi}_i = g(\mathbf{x}_i)$.

③ Use the preimage algorithm to reverse the parameterization, i.e. compute $\mathbf{x}_i{}' = g^{-1}(\boldsymbol{\xi}_i)$.

④ Compare $\mathbf{x}_i{}'$ with the original $\mathbf{x}_i$.

5.1.1. Comparison between the analytical and iterative solutions to the preimage problem

We begin by showing a few example results, in order to get a feel for the effect of each of the two preimage solution variants. Table 5.2 details the experimental setup and, as can be seen, we sampled a set of $n = 504$ channelized fields with 45×45 grid cells. We used the training image in figure 2.6 without imposing any hard data constraints. $n_t = 500$ samples were used as training set for the parameterization and, in building W (see algorithm 3.2), we retained the principal components that account for 90% of the variance (i.e. $q = 90$ in (3.3.8)).

Item	Description
Grid size	$45 \times 45 \times 1$ cells
Snesim search ellipsoid	$10 \times 10 \times 1$ grid cells (2D isotropic)
Target marginal distribution for facies	50% shale, 50% sand
Hard data contraints	none
Training set size	$n_t = 500$ samples
Parameterization variance threshold	$q = 90\%$
Starting solution for the iterative scheme	$\mathbf{x}^0 = \mathbf{0}$
Iterative scheme stopping threshold	0.1

Table 5.2: Experimental setup for the comparative preimage problem experiment

For the iterative scheme, we used $\mathbf{x}^0 = \mathbf{0}$ as initial solution and the following stopping criterion

$$\left\| \mathbf{x}^k - \mathbf{x}^{k-1} \right\|_2 \leq 0.1 \qquad k \in \mathbb{N}^* \tag{5.1.1}$$

which was chosen after investigating the convergence behaviour (figure 5.1). The main observation is that the difference, in $\|\|_2$-norm, between the solutions of successive iterations is decreasing

exponentially. Also, by comparing plots 5.1a and 5.1b, we see that the number of iterations required for a converged result seems to increase with the order of the kernel function.

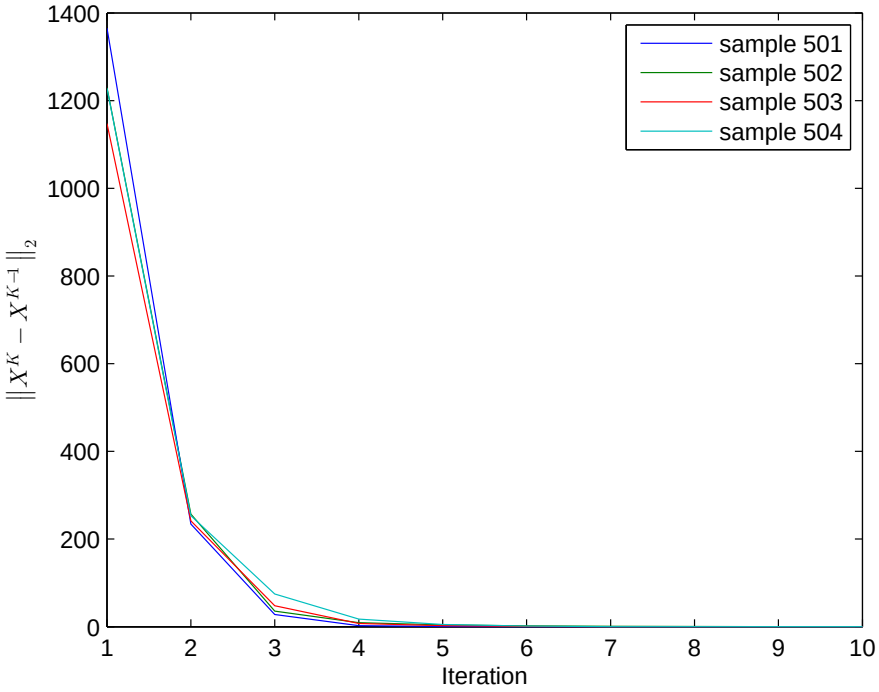
Figure 5.2a shows the results obtained with sample 501 and polynomial kernels of order 3. We see that both methods produce grid cells with values outside $[0, 1]$. The issue is most likely caused by the loss of information during the image-preimage cycle, together with the scope of the parameterization, which spans the whole real axis. We also notice that the iterative scheme is less affected, since its output is a linear (but not convex) combination of the training samples, with normalized coefficients (3.5.5).

As revealed in paragraph 2.3.4, these outliers are inconsistent with the physics and will raise difficulties in reservoir simulation. But, more importantly for the current experiment, they cause a shift in colormaps, which impedes our comparison of the preimage results. Therefore, we will constrain subsequent results to $[0, 1]$, via truncation. Another approach would be to apply the logistic transform to the samples before we feed them to the parameterization, and paragraph 5.1.4 shows the impact of this alternative.

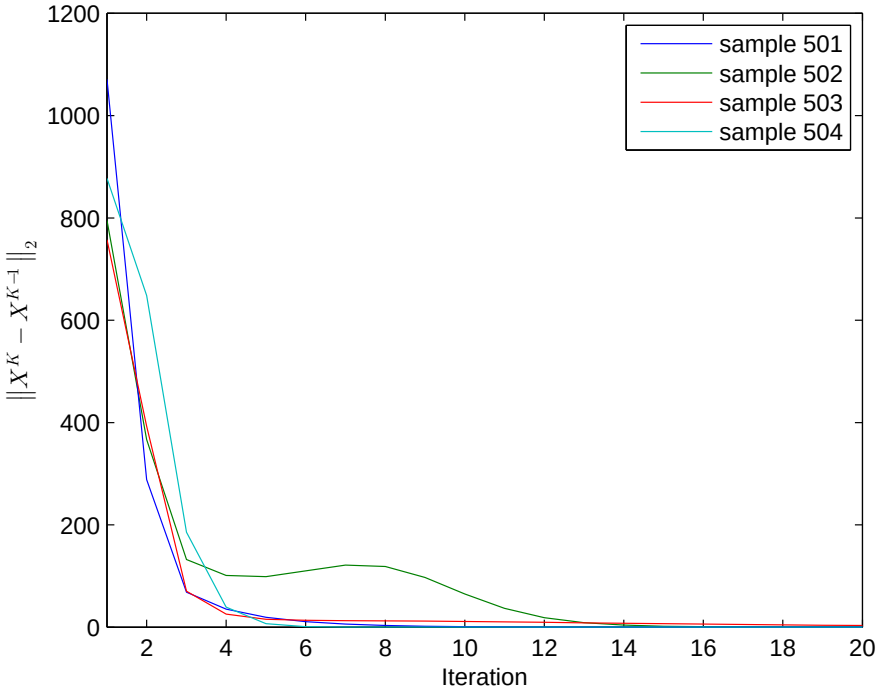
We now turn to figures 5.2 to 5.5, which illustrate the (truncated) outputs of both methods when faced with the $n_e = 4$ test samples and different degrees of the polynomial kernel, $d \in \{3, 5, 7, 9\}$. We notice that with the increase in d, the analytical solution produces nearly uniform fields, while, for the iterative scheme, the search "collapses" onto one of the training samples (figures 5.3c and 5.5c). Therefore, both preimage methods fail for higher values of d.

On the opposite side of the spectrum, when $d = 3$ the analytical method is generally able to reproduce the details of the channel shape better than the iterative scheme (for example, the connection on the right edge of figure 5.4b or the top tail in figure 5.5b).

We now have a general idea of each method's performance. More in-depth properties will be revealed during the sensitivity studies following this paragraph.

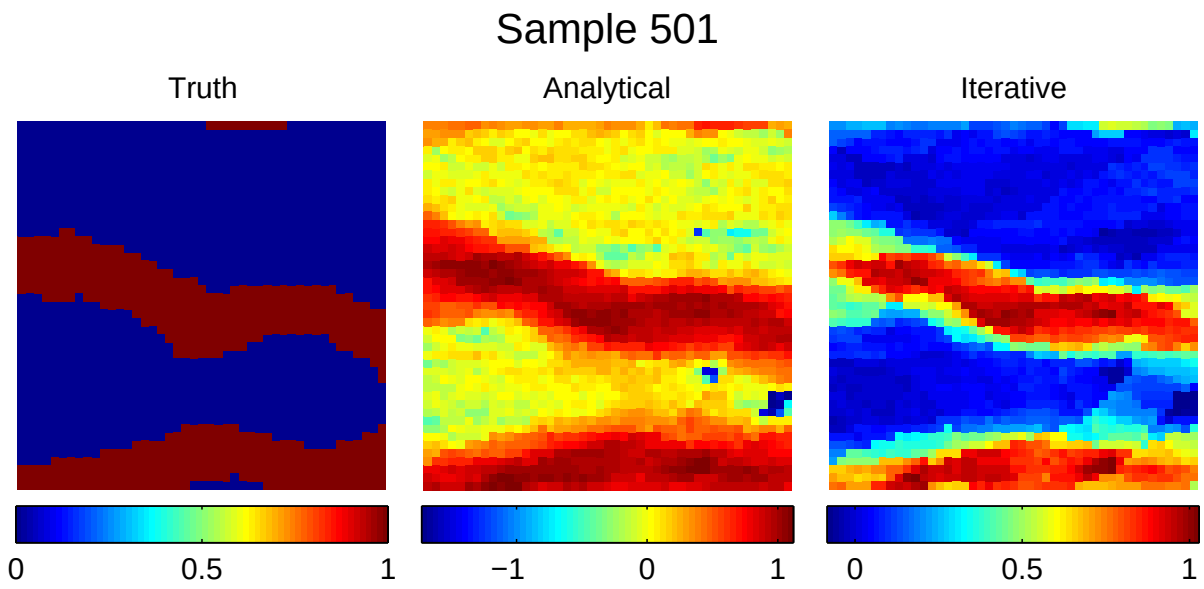


(a) order $d = 3$

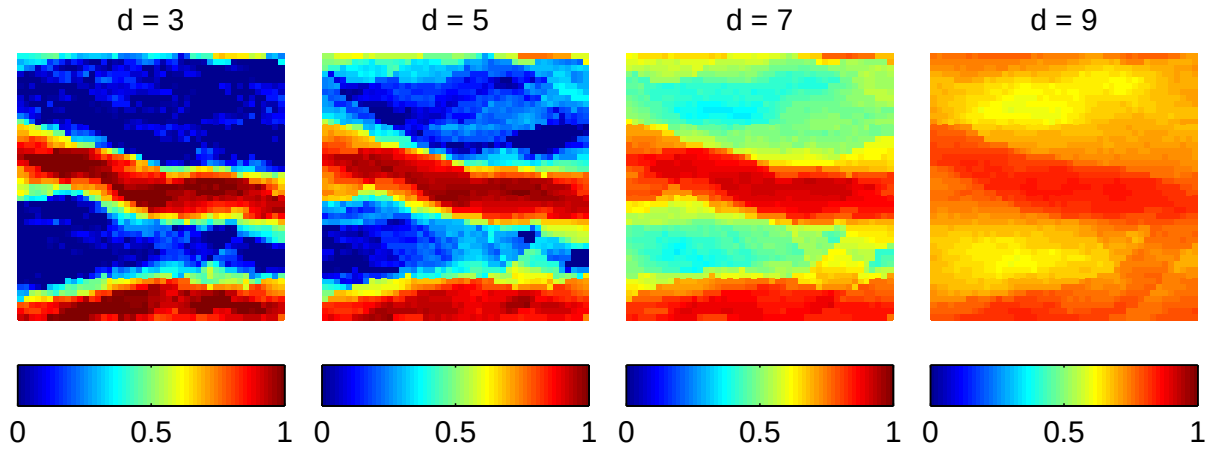


(b) order $d = 5$

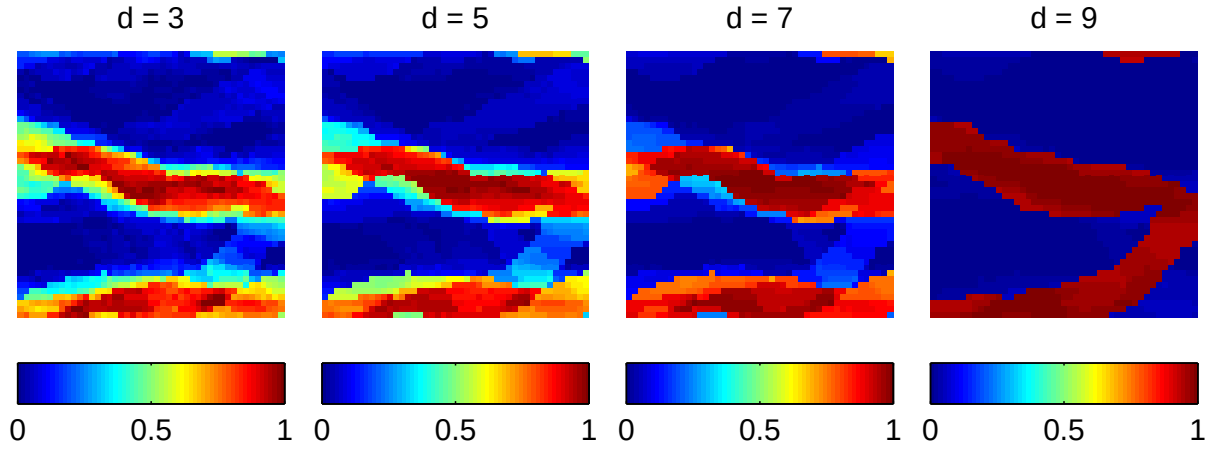
Figure 5.1: Fixed-point iterative scheme convergence



(a) Preliminary preimage problem results, $d = 3$



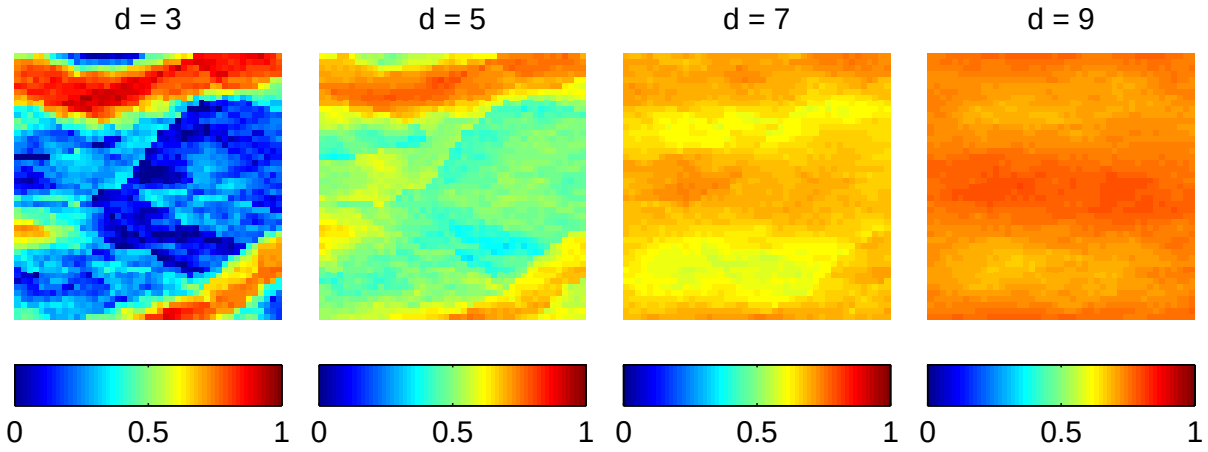
(b) Analytical solution



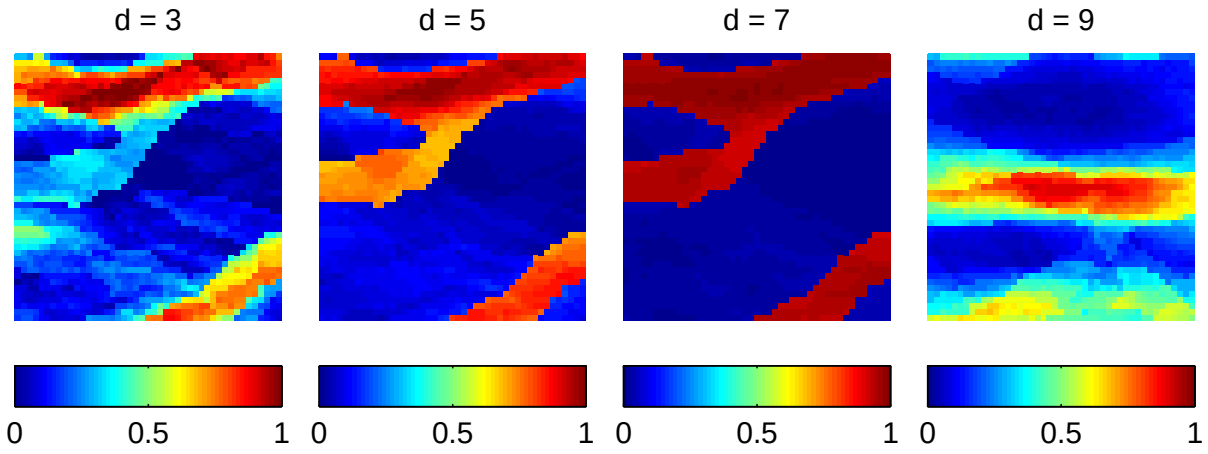
(c) Fixed-point iterative solution

Figure 5.2: Preimage results for sample 501

(a) Truth



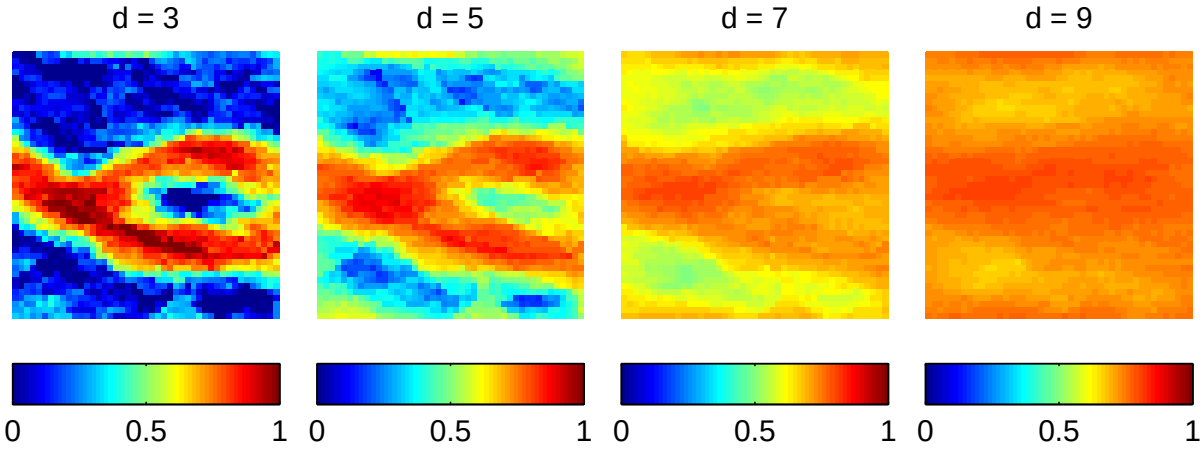
(b) Analytical solution



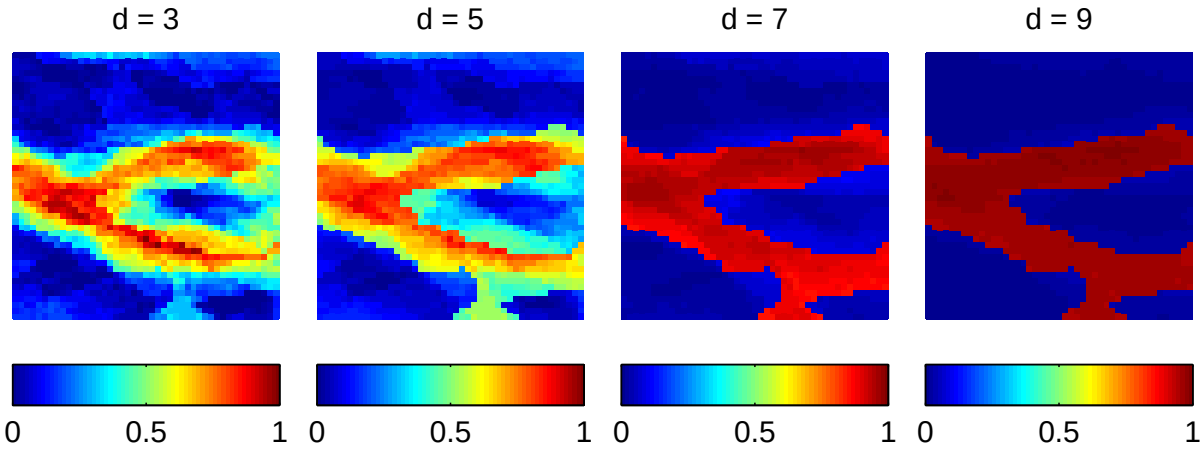
(c) Fixed-point iterative solution

Figure 5.3: Preimage results for sample 502

(a) Truth

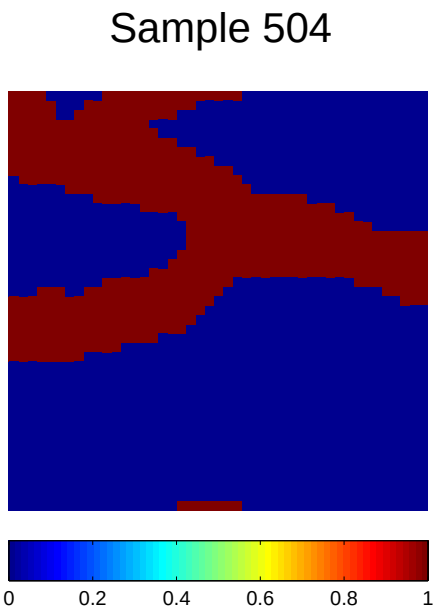


(b) Analytical solution

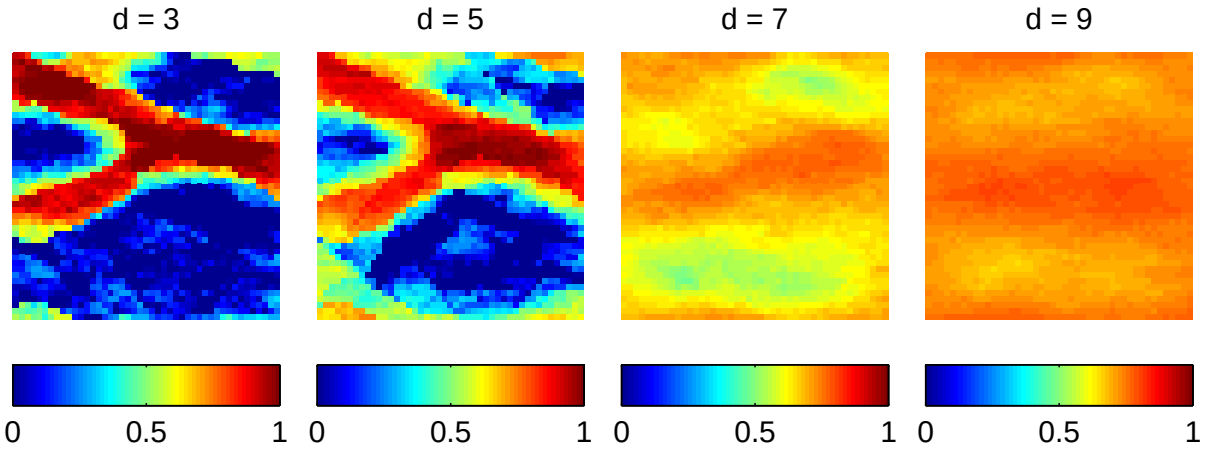


(c) Fixed-point iterative solution

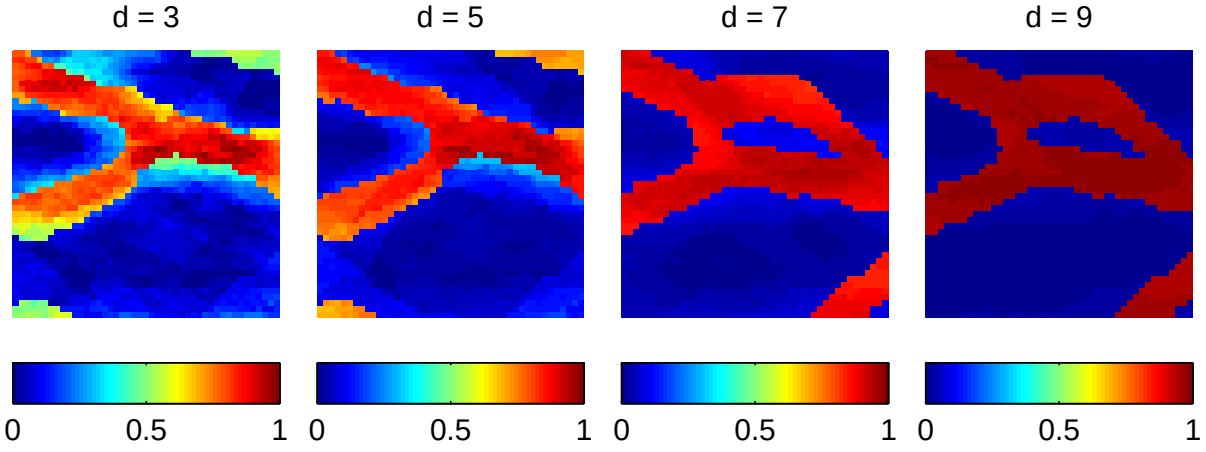
Figure 5.4: Preimage results for sample 503



(a) Truth



(b) Analytical solution



(c) Fixed-point iterative solution

Figure 5.5: Preimage results for sample 504

5.1.2. Sensitivity to the size of the training set

The previous paragraph showed that the preimage results degrade for higher order of the polynomial kernel function. Recall that the dimensionality of the feature space increases with the factorial of d (3.2.17), hence, more training samples are required in order to obtain a converged covariance matrix. We will now put this hypothesis to the test and study the effect of the training set size on preimage results.

The only difference in setup from the previous experiment (table 5.2) is that we change the number of training samples, n_t, from 500, to 1000, 2000 and 4000. As test subjects, we will use $n_e = 2$ new samples – 505, with a Y-shaped channel (figure 5.6a), and 506, with a more complex ribbon shape (figure 5.7a).

As illustrated in figure 5.6, both methods seem to benefit from an increased training set. On the one hand, the results of the analytical method become sharper and the channel structure is reproduced even for higher kernel degrees (see $d = 7$ in figure 5.6f). On the other hand, the iterative scheme gains robustness in avoiding local optima (compare figures 5.6c and 5.6e to figure 5.6g) and, with $n_t = 4000$, it is able to correctly identify the shape of sample 505, even when $d = 9$.

Sample 506 raises more difficulty due to its ribbon channel, which is not well represented in the training set. This is reflected in figures 5.7c, 5.7e and 5.7g, where, regardless of the amount of training samples, the iterative scheme has difficulties in reconstructing the correct shape. In contrast, the analytical method produces a reasonable output for $d = 3$, even with n_t at 500 (figure 5.7b), and, after increasing the size of the training set, we see that the channel features start to appear even when $d = 7$ and 9 (figure 5.7f).

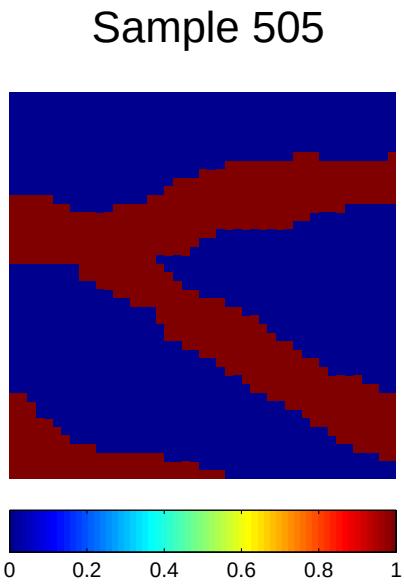
In conclusion, we now have experimental evidence that the poor results obtained with higher order kernels are due to an unconverged covariance matrix in the feature space. Still, the rate at which we have to increase the size of the training set in order to obtain better performance seems to be exponential in d. Also, we observed that an increase in the size of the training set benefits the analytical solution more than the iterative. This can be due the fact that the dimension of the search space increases with n_t, making it more likely to arrive in a local optimum.

5.1.3. Sensitivity to the size of the reservoir grid

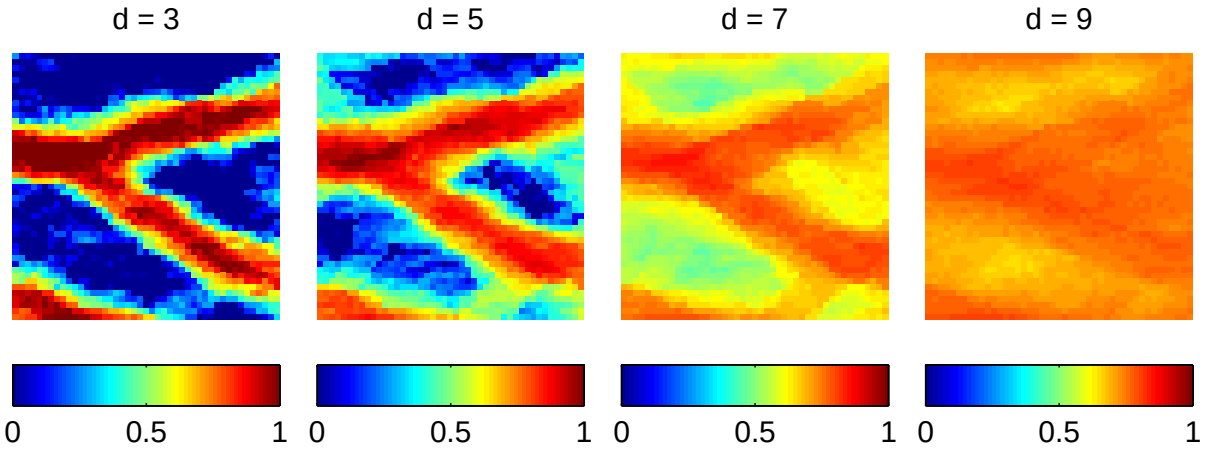
Another aspect to consider is the size of the reservoir grid, which, as per equation (3.2.17), also contributes to the dimensionality of the polynomial feature space. For this purpose, we have used snesim to generate three new sets with $n = 1501$ samples and grid sizes of 65×65 (figure 5.8), 100×100 (figure 5.9) and 200×200 (figure 5.10) cells, respectively. The first $n_t = 1500$ samples from each set were used to train the parameterizations, and the last $n_e = 1$, for testing.

The results are presented in figures 5.8 to 5.10 and we notice that, regardless of grid size, the best results were obtained for lower kernel degrees. Indeed, when $d = 3$ or $d = 5$, the channel boundaries are clearly distinguishable – more so for the analytical method than the iterative scheme (compare, for example, figure 5.8b to 5.8c). As d increases, however, the outputs of both methods tend to become uniform, to the point where the channels can hardly be separated from the background (figures 5.9 and 5.10 for $d = 9$).

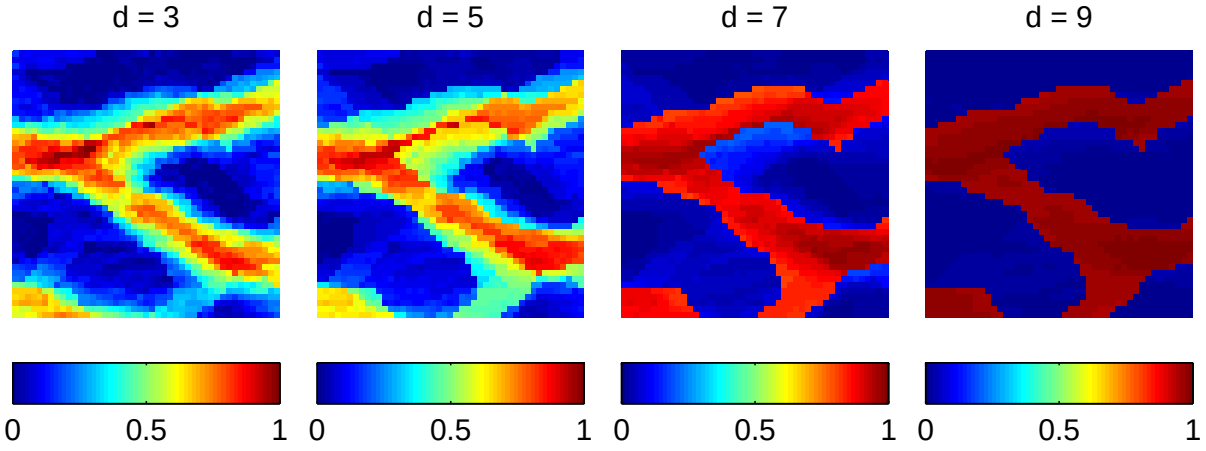
If we compare figures 5.8 and 5.10, we see that, while a higher grid size does seem to reduce the sharpness of the preimages produced by both methods, the impact is not dramatic. This is a promising result in the prospect of applying the parameterization (albeit with a low value of d) to large-scale reservoirs ($\sim 10^6$ grid cells).



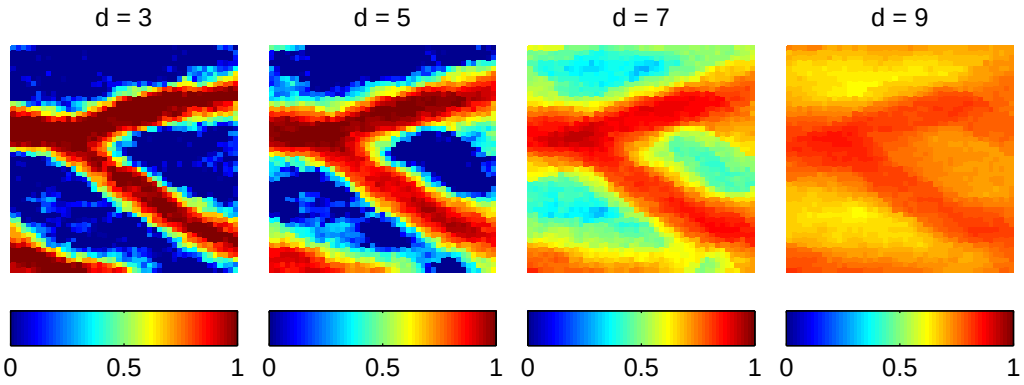
(a) Truth



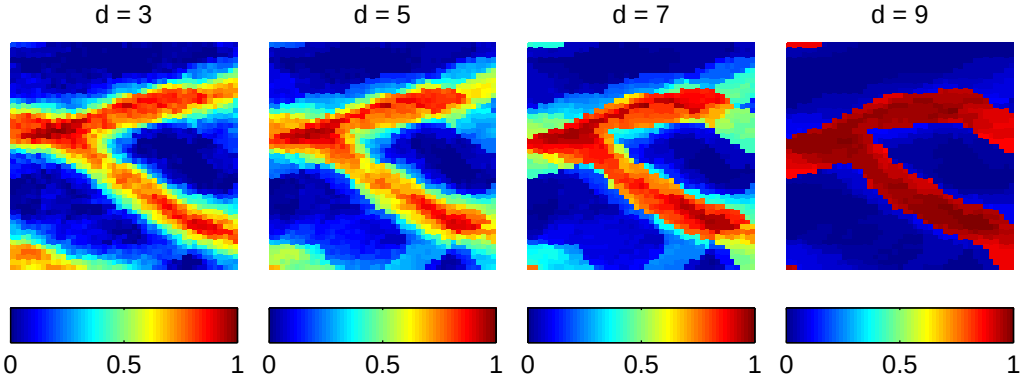
(b) Analytical solution with 500 training samples



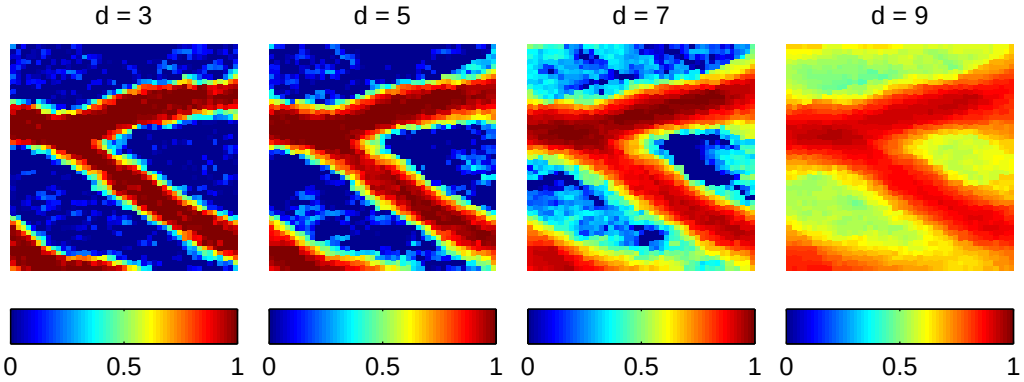
(c) Fixed-point iterative solution with 500 training samples



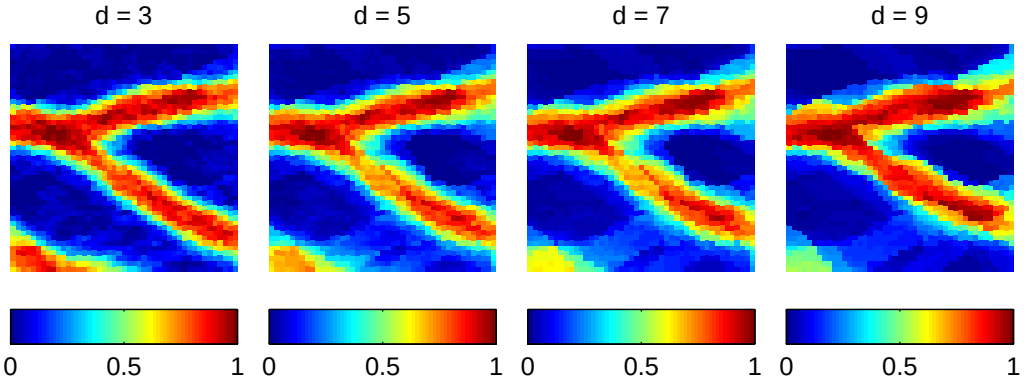
(d) Analytical solution with 1000 training samples



(e) Fixed-point iterative solution with 1000 training samples



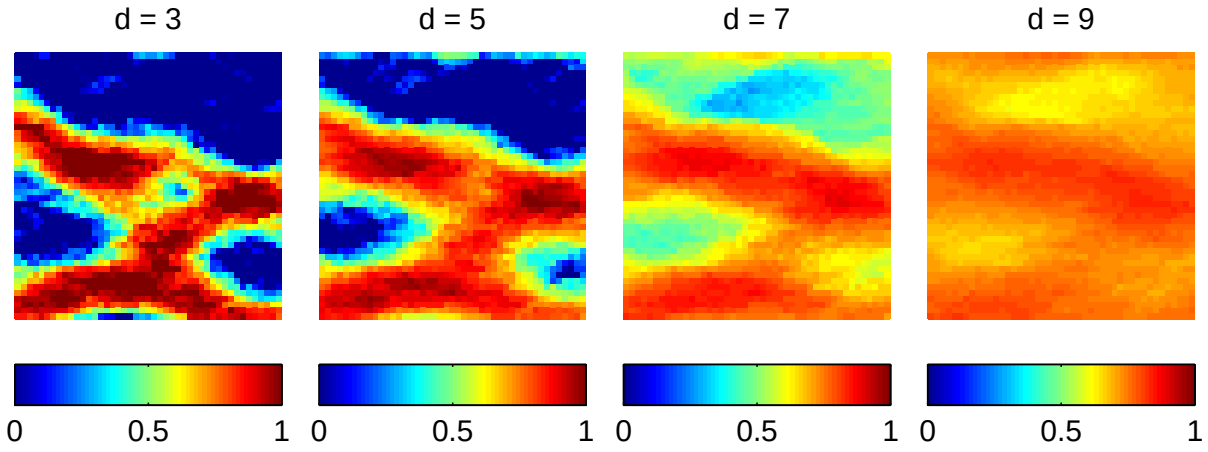
(f) Analytical solution with 4000 training samples



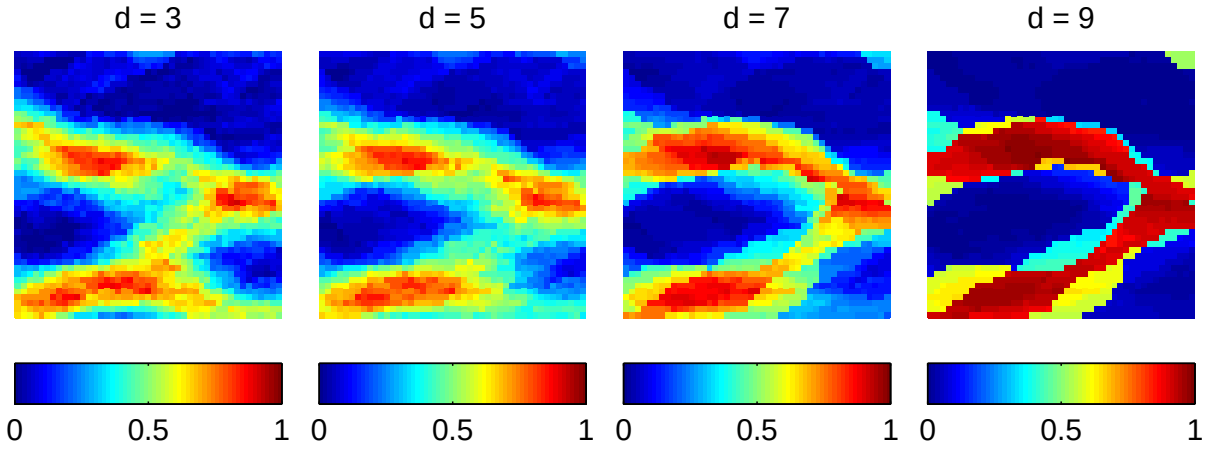
(g) Fixed-point iterative solution with 4000 training samples

Figure 5.6: Preimage results for sample 505 with different training set sizes

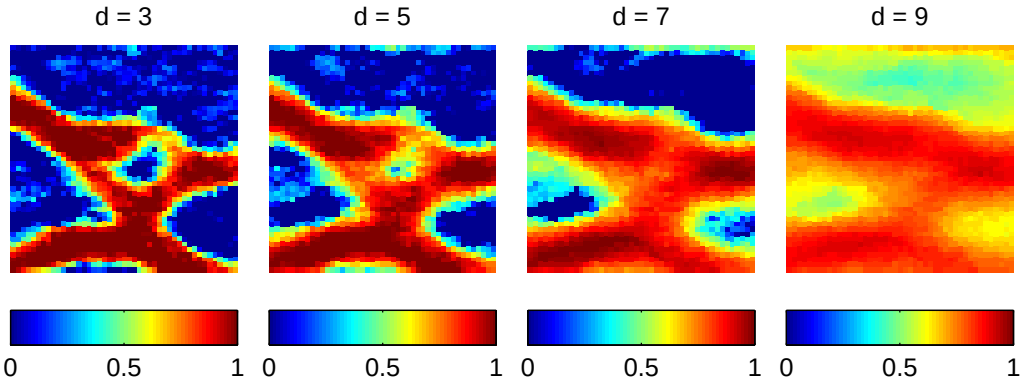
(a) Truth



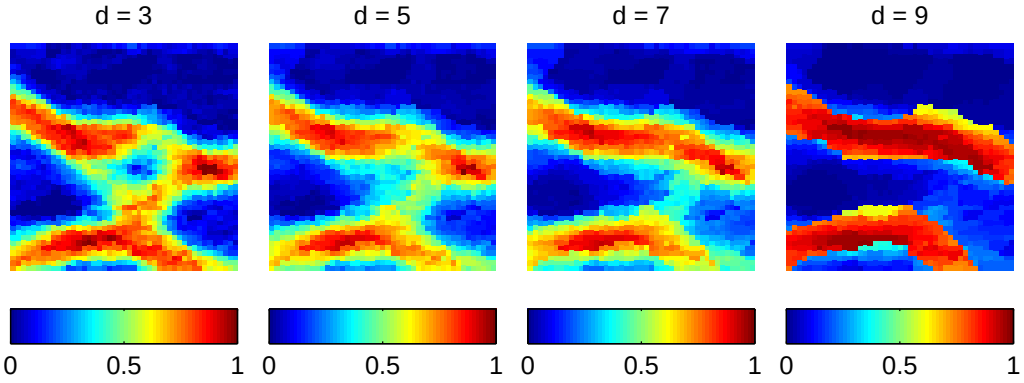
(b) Analytical solution with 500 training samples



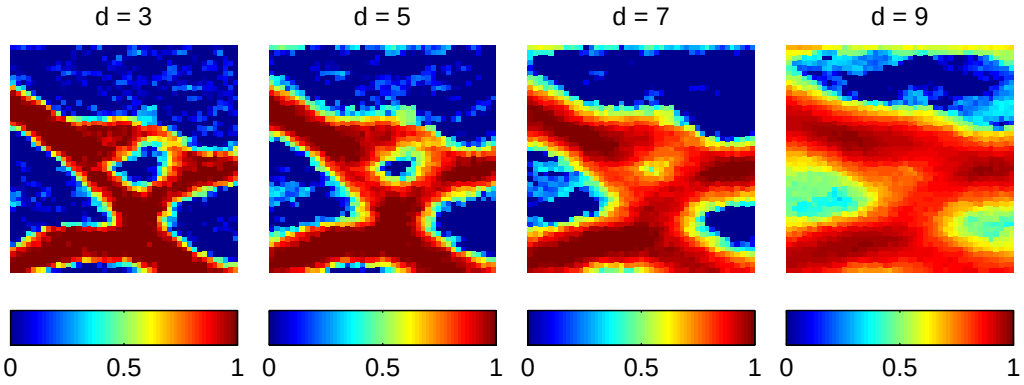
(c) Fixed-point iterative solution with 500 training samples



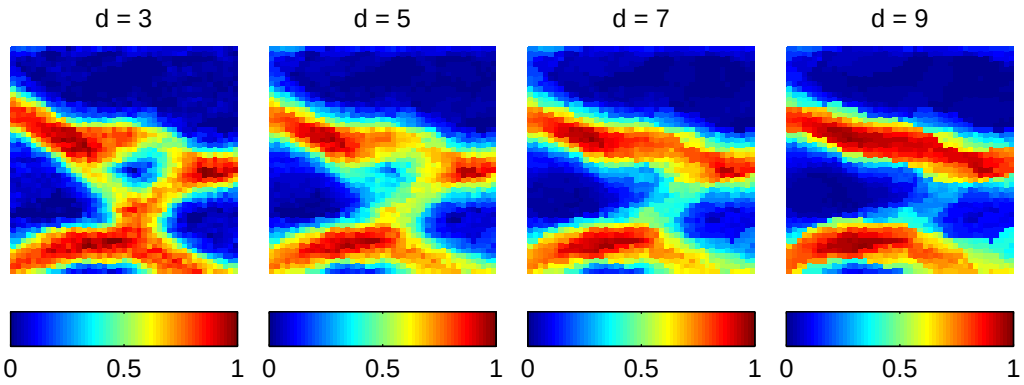
(d) Analytical solution with 2000 training samples



(e) Fixed-point iterative solution with 2000 training samples

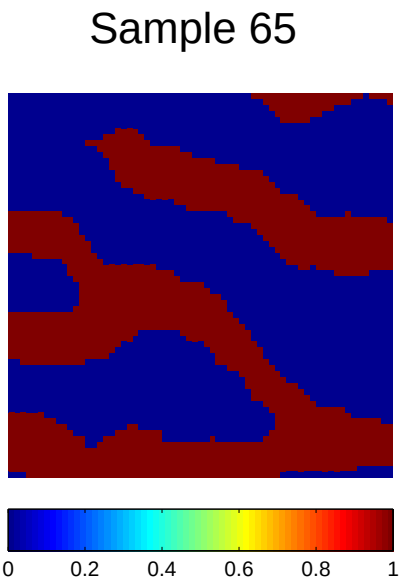


(f) Analytical solution with 4000 training samples

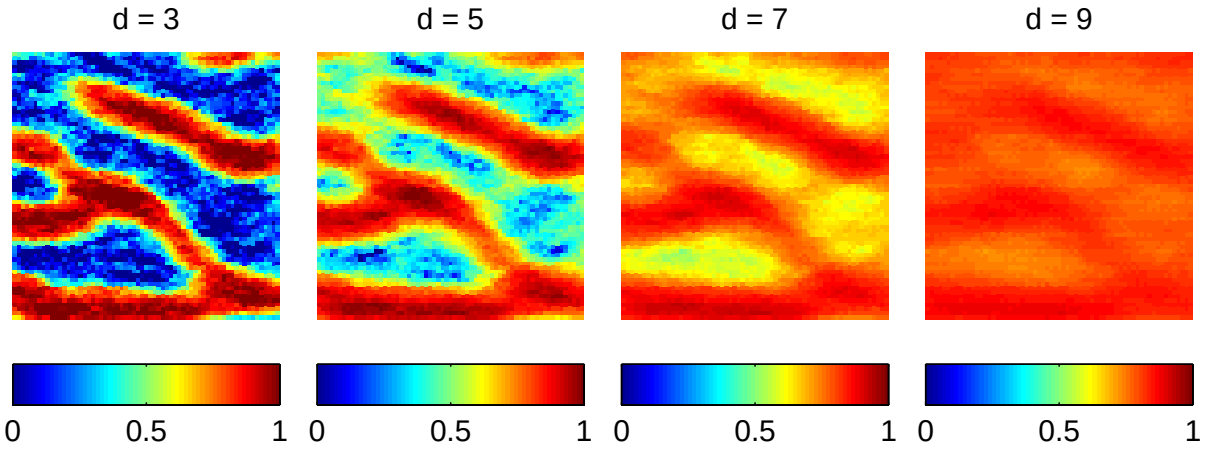


(g) Fixed-point iterative solution with 4000 training samples

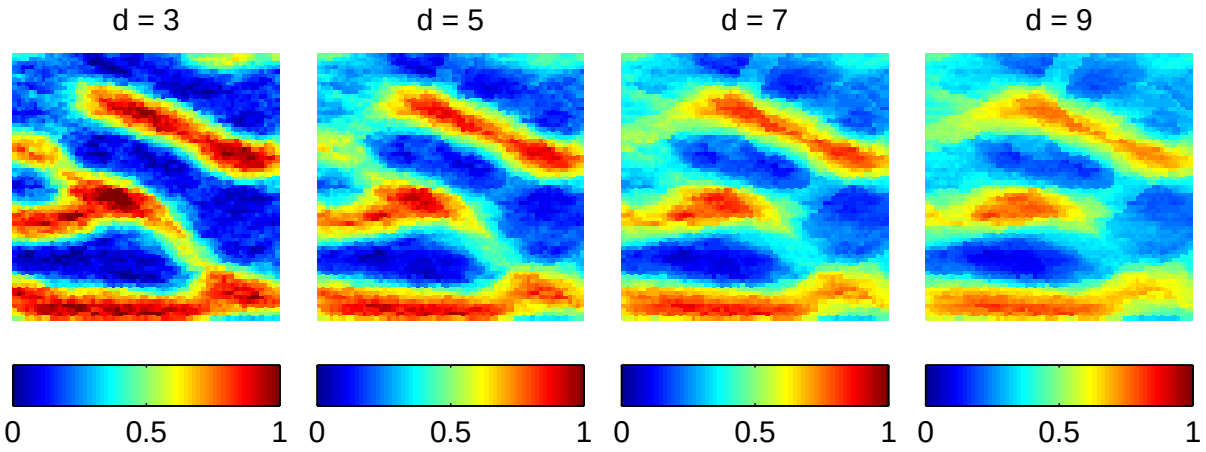
Figure 5.7: Preimage results for sample 506 with different training set sizes



(a) Truth

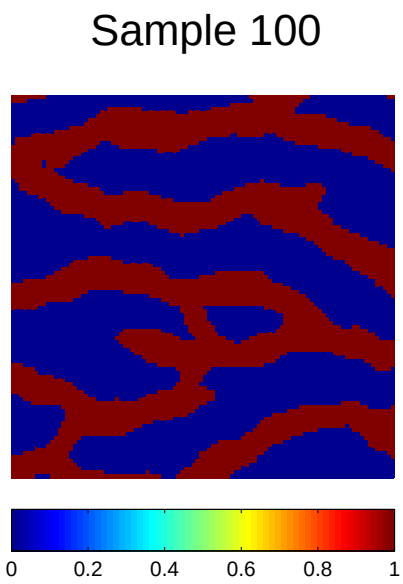


(b) Analytical solution

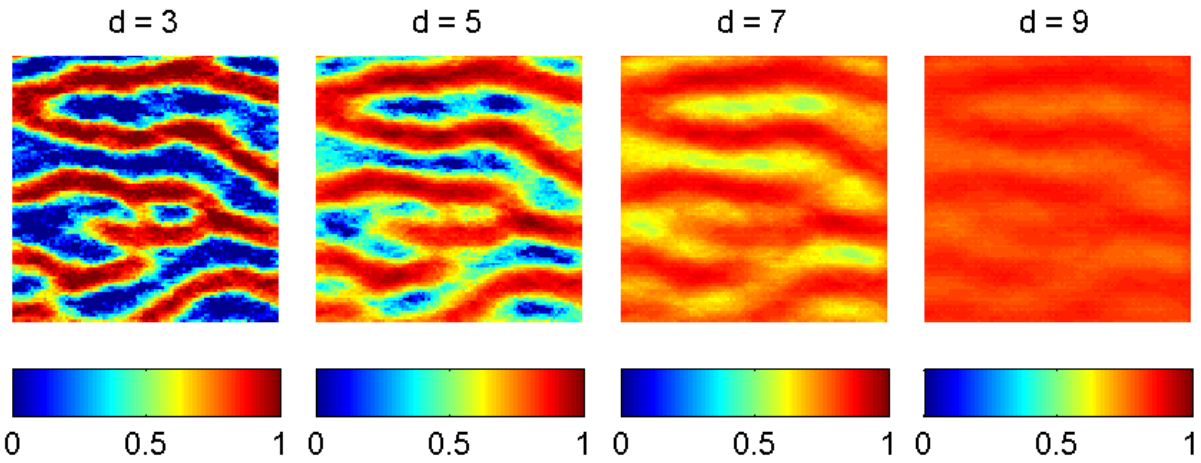


(c) Fixed-point iterative solution

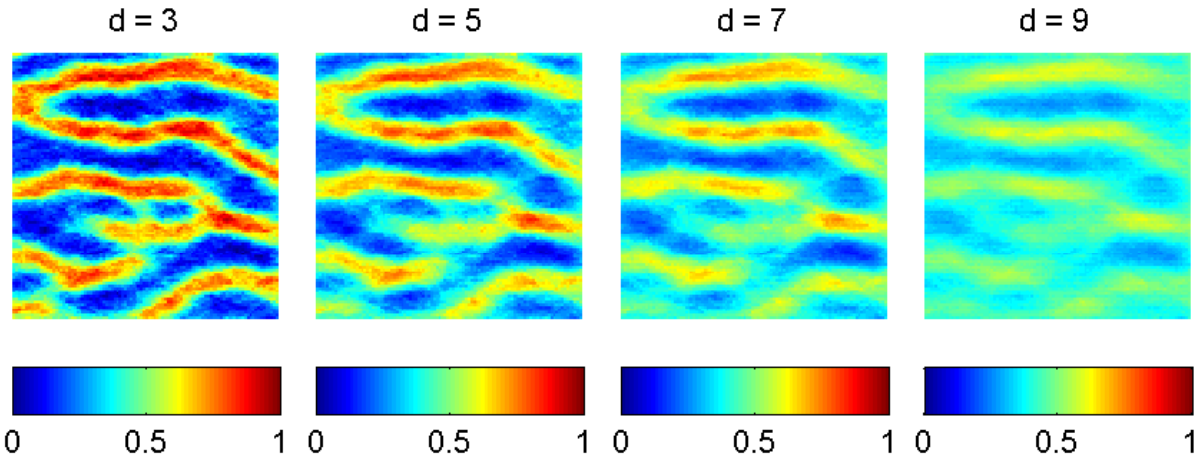
Figure 5.8: Preimage results for a 65×65 reservoir grid



(a) Truth

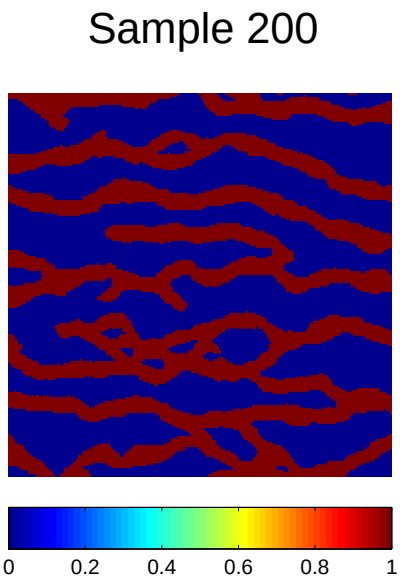


(b) Analytical solution

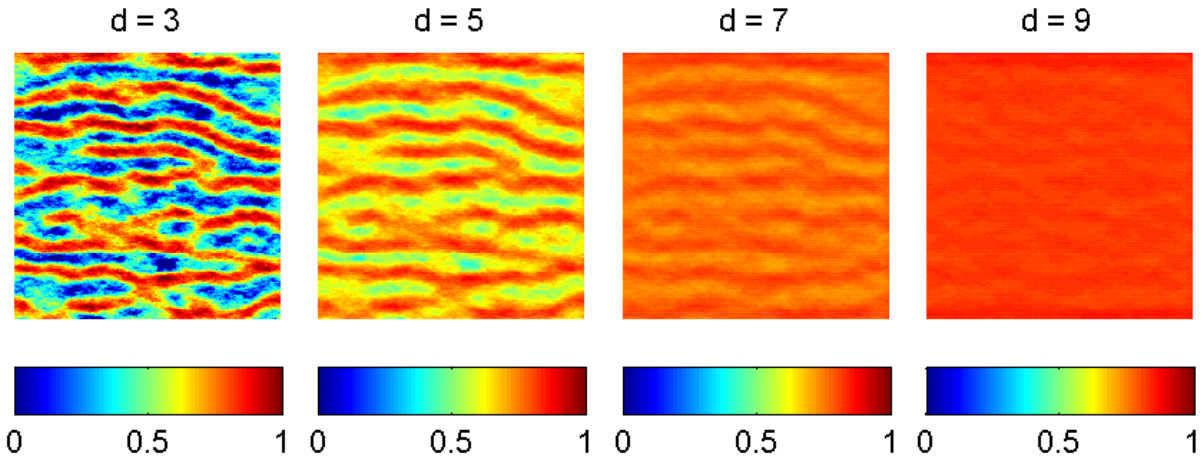


(c) Fixed-point iterative solution

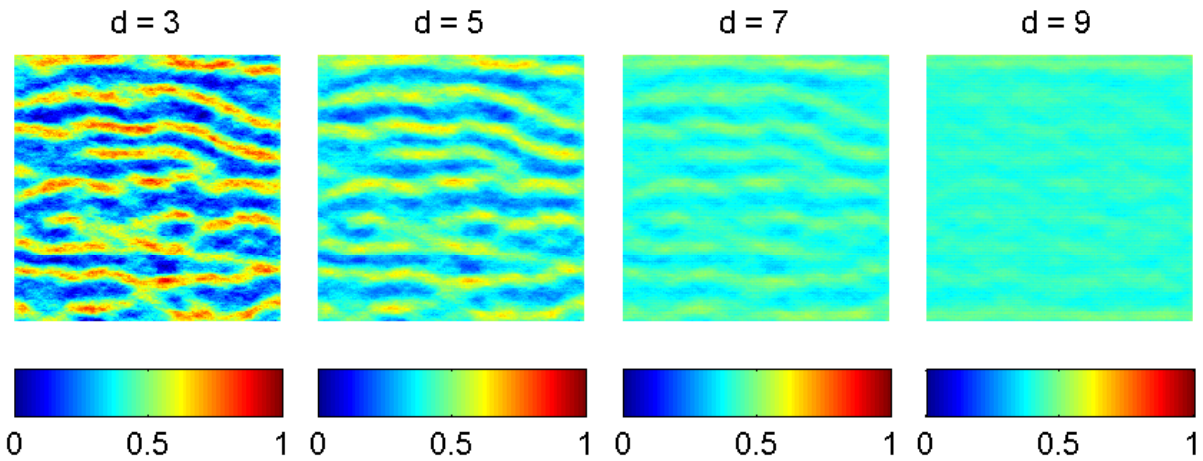
Figure 5.9: Preimage results for a 100×100 reservoir grid



(a) Truth



(b) Analytical solution



(c) Fixed-point iterative solution

Figure 5.10: Preimage results for a 200×200 reservoir grid

5.1.4. Effect of the logistic transform

This paragraph describes the final experiment performed on the parameterization, before incorporating it into the data assimilation cycle. Specifically, we studied the effect of the logistic transform, as an alternative to truncation in constraining the preimage results to the interval $[0, 1]$. This implies evaluating the *logit* function (2.3.46) in each reservoir grid cell, before feeding the samples to the parameterization, and, in turn, applying the inverse (2.3.47) on the outputs.

For comparison, we re-ran some of the previous experiments with their corresponding setups, and the results are illustrated in figure 5.11. We observe, straight away, that the marginal distribution in all results are, now, nearly binary. This seems to be a side effect of the logistic transform, which acts, in essence, similar to a thresholding filter, suppressing the intermediate values from $[0, 1]$ and, thus, making the preimages consistent with the truth.

Figures 5.11a to 5.11d show the outputs for samples 502 and 503 and we notice that the results of the analytical solution seem to "erode" as the order increases, while the iterative scheme attains local optima (that sometimes differ from those obtained with truncation; compare, for example, figures 5.3c and 5.11b). The behaviour is the same in figures 5.11e to 5.11h, where we used increased training set sizes, $n_t = 1000$ for sample 505 and $n_t = 2000$ for sample 506, respectively (see the corresponding plots in figures 5.6 and 5.7).

Finally, in figures 5.11i to 5.11l we give the outputs for higher grid sizes (samples 100 and 200). Notice that, in this case, the "filtering" behaviour eliminates a good number of important features from the channel structure (see the corresponding figures 5.9 and 5.10). Therefore, unlike the truncated version we studied earlier, the logistic polynomial kernel parameterization may prove problematic in applications with large-scale reservoirs. On the bright side, the experiments in paragraph 5.1.2 suggest that a higher number of training samples may help in improving the accuracy of the results. However, as we will see in the following, this comes at the cost of more computational resources.

5.1.5. Computational expense

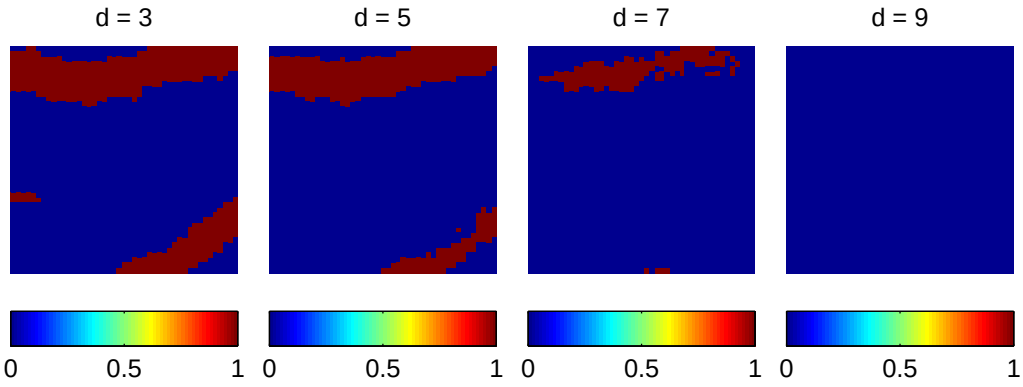
This paragraph shows a benchmark of the computational time necessary to train and apply the polynomial kernel parameterization. The measurements were done by setting the affinity of MATLAB to a single microprocessor core and performing tests on sets of 1000 samples. As before, we vary the size of the training set and reservoir grid.

The values are given in tables 5.3 and 5.4 and plotted in figures 5.12 and 5.13 respectively. Right from the get-go, we notice that the training times are relatively similar for all methods (left-hand plots), while for the run times there is a difference of an order of magnitude, favouring the analytical method over the iterative (right-hand plots).

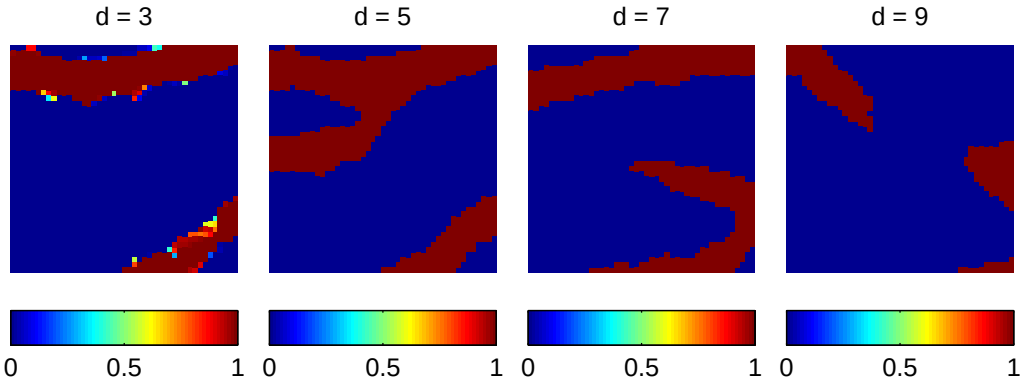
Variations in n_t seem to have a superlinear impact on the training time (table 5.3). This is because n_t gives the size of the kernel matrix and the complexity of the SVD algorithm (see Golub and van Loan 2012). On the othe hand, in terms of run time, n_t only displays linear influence, however we notice a difference in slope between the four methods. This is most likely caused by two factors:

- The first is the nature of the fixed-point scheme, which requires multiple iterations before a converged result is obtained. The analytical method is at advantage here, since, under the same circumstances, it always performs the same amount of computational work, roughly equivalent to one fixed-point iteration.
- The second is the logistic transform, which, besides the flat amount of overhead induced by evaluating the *logit* function and its inverse for each sample, seems to also have a negative impact on the iterative scheme's rate of convergence (table 5.3).

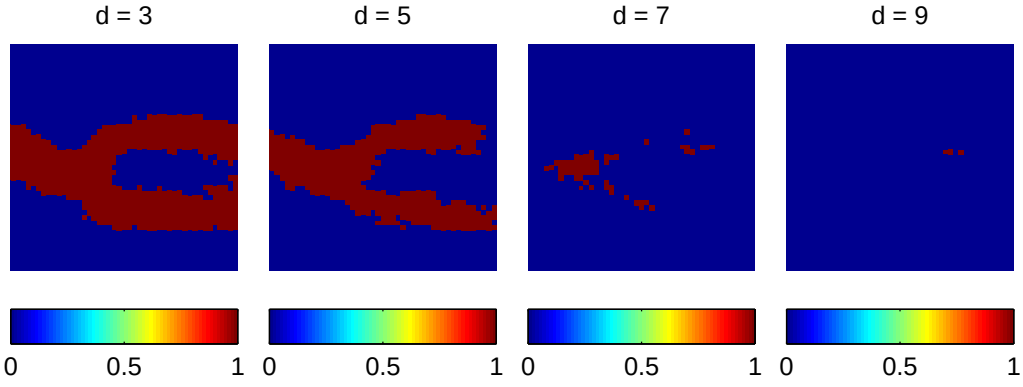
Increasing the grid size has a linear influence on both training and run times (table 5.4). By comparing the right plots of figures 5.12 and 5.13 we also notice an impact on the convergence rate of the logistic iterative scheme, which is even more significant than that of n_t.



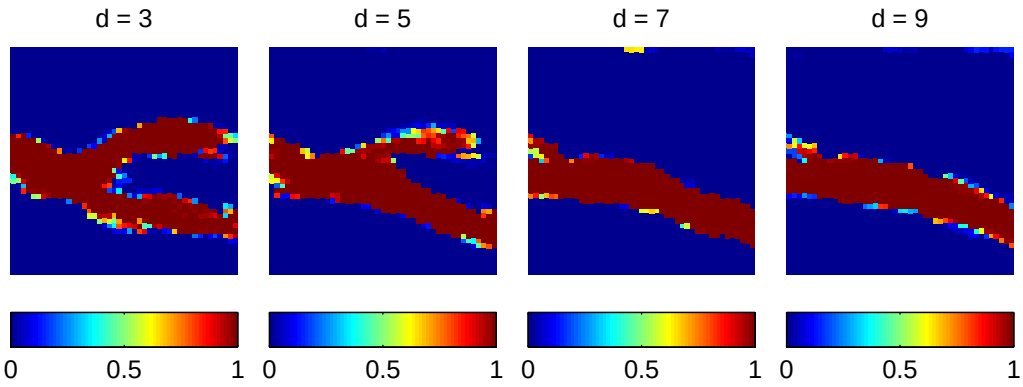
(a) Logistic analytical solution for sample 502



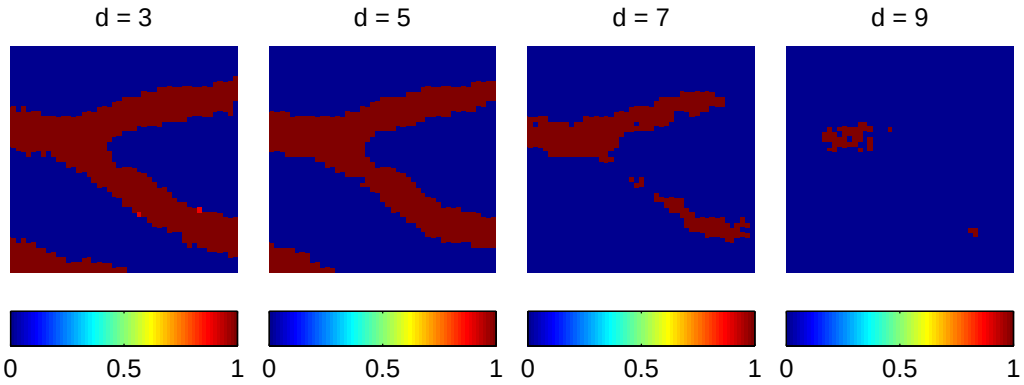
(b) Logistic fixed-point iterative solution for sample 502



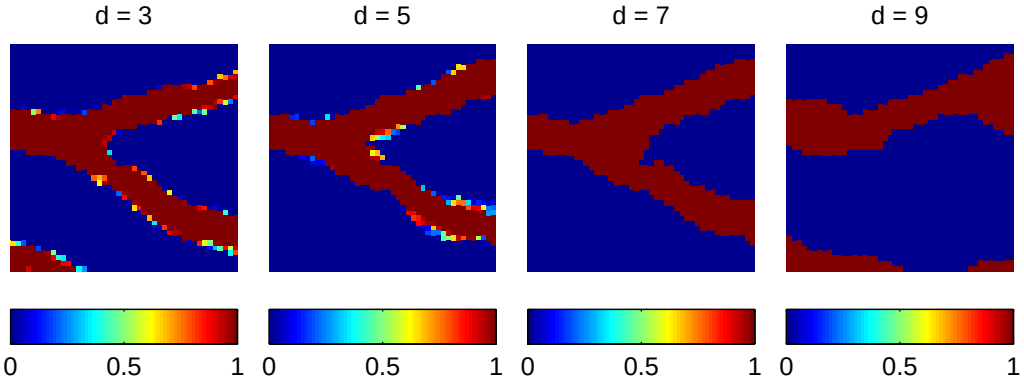
(c) Logistic analytical solution for sample 503



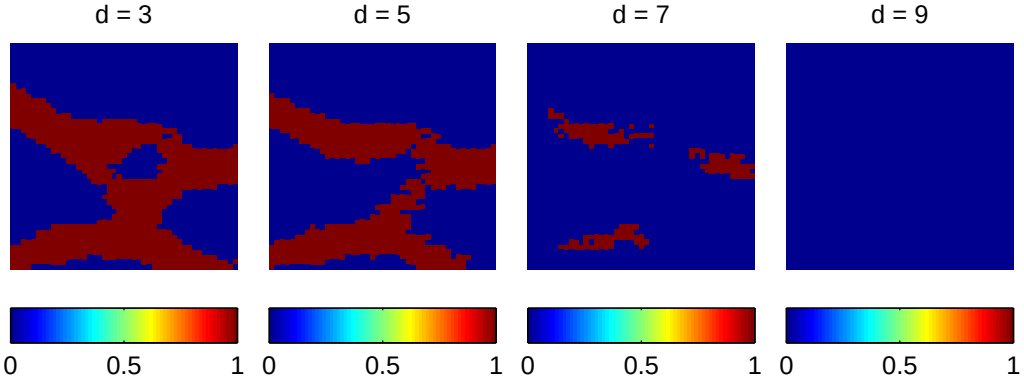
(d) Logistic fixed-point iterative solution for sample 503



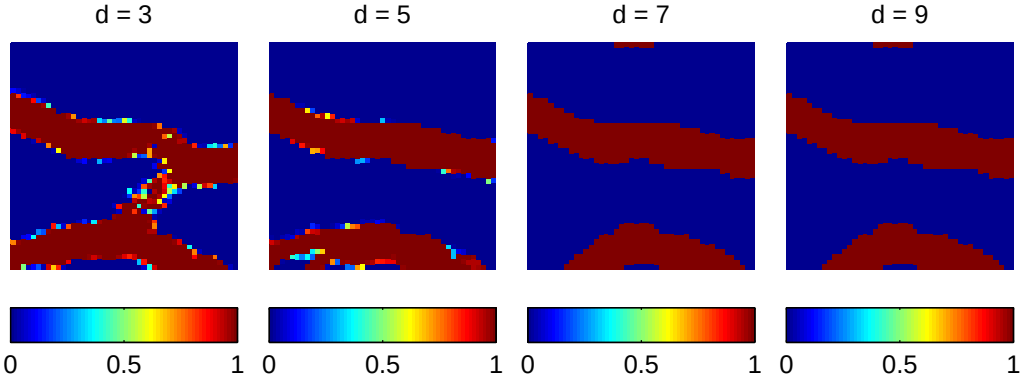
(e) Logistic analytical solution for sample 505 with training set size 1000



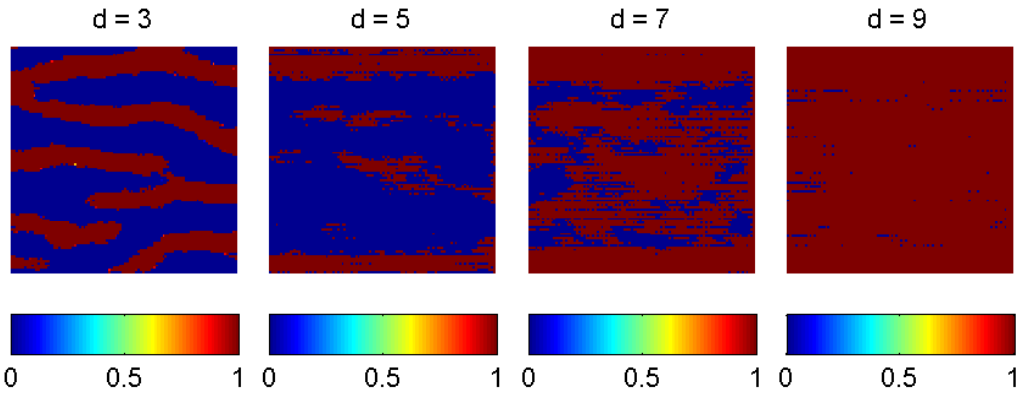
(f) Logistic fixed-point iterative solution for sample 505 with training set size 1000



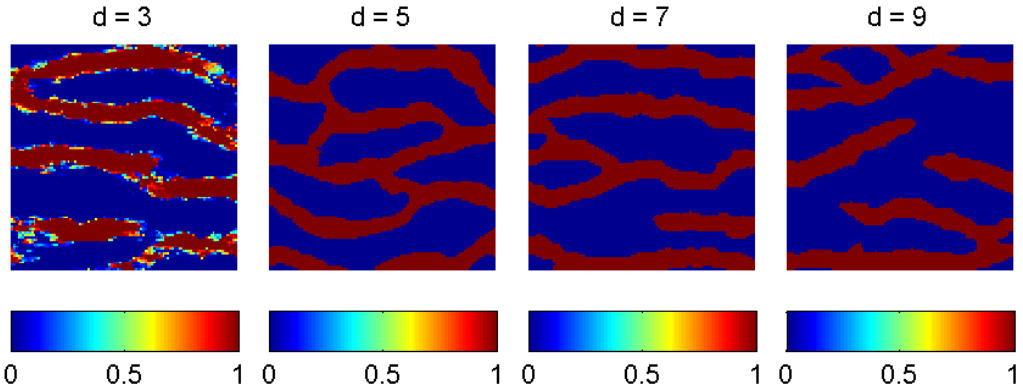
(g) Logistic analytical solution for sample 506 with training set size 2000



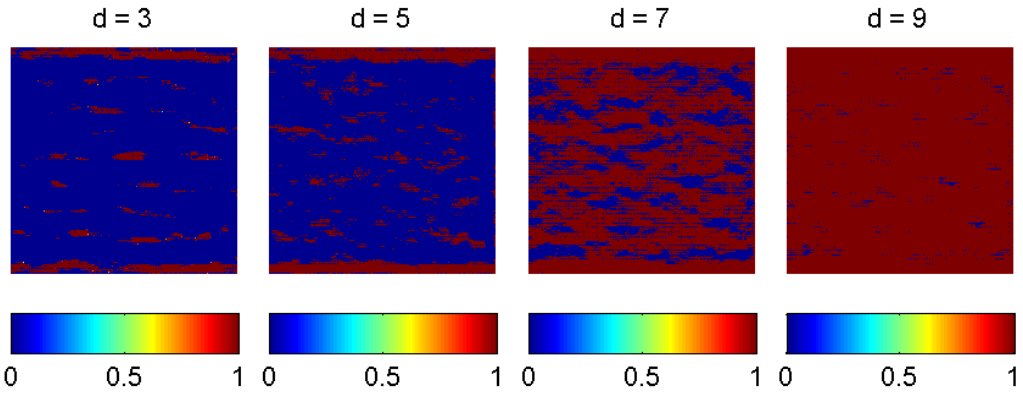
(h) Logistic fixed-point iterative solution for sample 506 with training set size 2000



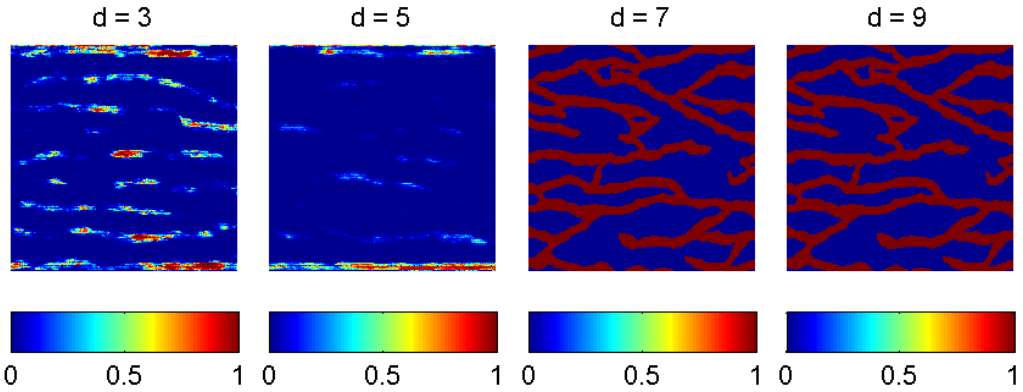
(i) Logistic analytical solution for sample 100



(j) Logistic fixed-point iterative solution for sample 100



(k) Logistic analytical solution for sample 200



(l) Logistic fixed-point iterative solution for sample 200

Figure 5.11: Effect of the logistic transform on the parameterization

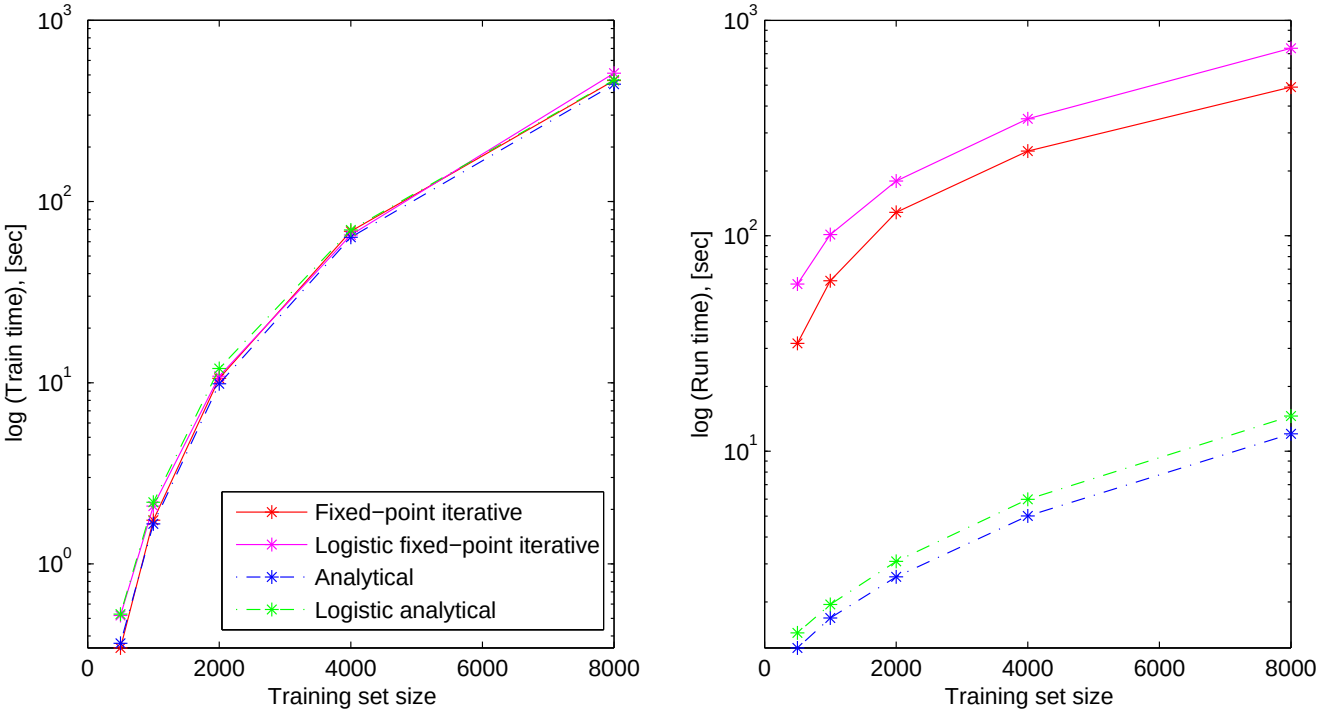


Figure 5.12: Comparison of computational times with different training set sizes; left: training time; right: run time for 1000 samples

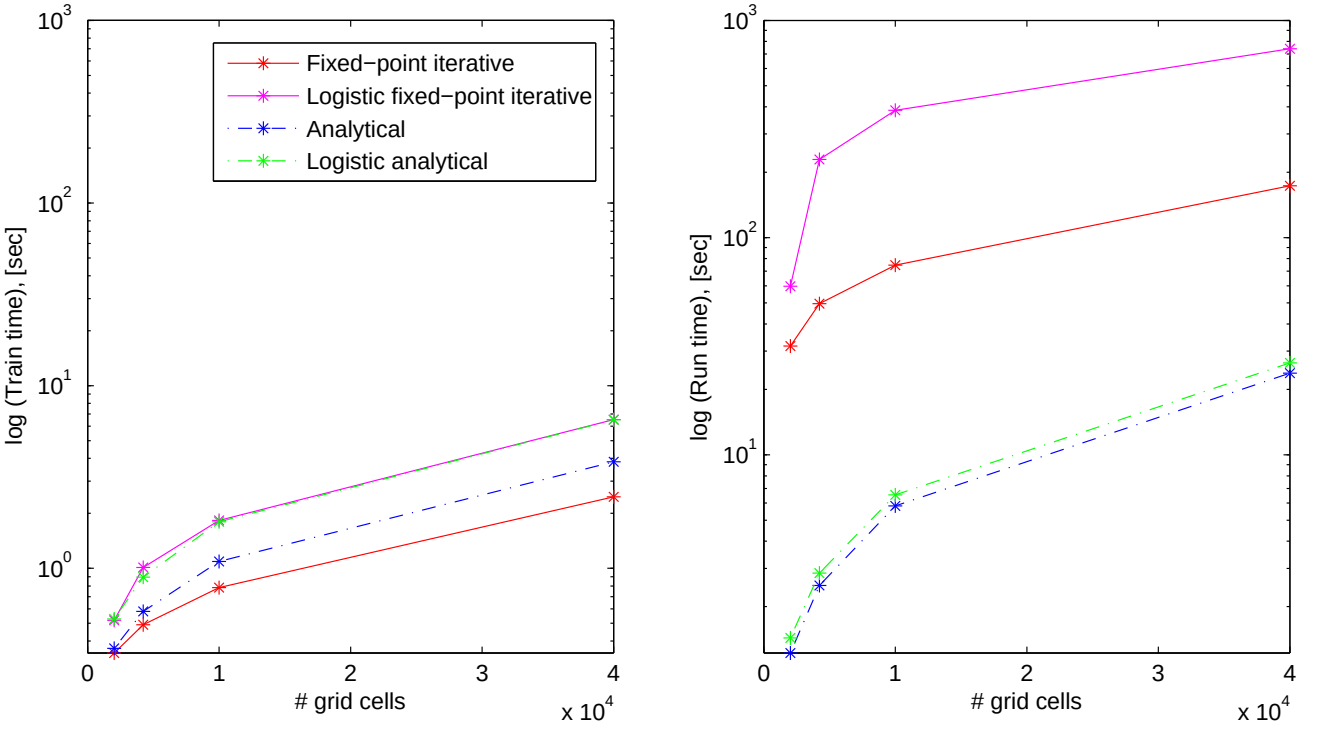


Figure 5.13: Comparison of computational times with different grid sizes; left: training time; right: run time for 1000 samples

Training set size	Training time	Run time
500	0.3442 s	31.7001 s
1000	1.7412 s	61.8026 s
2000	10.5670 s	128.4134 s
4000	68.5852 s	247.3553 s
8000	465.8573 s	489.8302 s

(a) Fixed-point iterative solution

Training set size	Training time	Run time
500	0.5192 s	59.6966 s
1000	2.0877 s	101.3104 s
2000	10.8576 s	179.4641 s
4000	65.4415 s	348.8961 s
8000	511.1044 s	741.5589 s

(b) Logistic fixed-point iterative solution

Training set size	Training time	Run time
500	0.3649 s	1.2238 s
1000	1.6637 s	1.6829 s
2000	9.8691 s	2.6152 s
4000	63.4499 s	5.0124 s
8000	443.6609 s	12.0598 s

(c) Analytical solution

Training set size	Training time	Run time
500	0.5280 s	1.4352 s
1000	2.1894 s	1.9439 s
2000	11.9768 s	3.0832 s
4000	69.7542 s	5.9801 s
8000	468.6185 s	14.5627 s

(d) Logistic analytical solution

Table 5.3: Computational times when using different training set sizes

Grid size	Training time	Run time
45×45	0.3442 s	31.7001 s
65×65	0.4914 s	49.6051 s
100×100	0.7844 s	74.6686 s
200×200	2.4634 s	173.0273 s

(a) Fixed-point iterative solution

Grid size	Training time	Run time
45×45	0.5192 s	59.6966 s
65×65	1.0116 s	229.0619 s
100×100	1.8292 s	385.6604 s
200×200	6.5193 s	740.8032 s

(b) Logistic fixed-point iterative solution

Grid size	Training time	Run time
45×45	0.3649 s	1.2238 s
65×65	0.5816 s	2.5042 s
100×100	1.0908 s	5.8159 s
200×200	3.8302 s	23.7782 s

(c) Analytical solution

Grid size	Training time	Run time
45×45	0.5280 s	1.4352 s
65×65	0.8940 s	2.8523 s
100×100	1.7973 s	6.5384 s
200×200	6.5026 s	26.5016 s

(d) Logistic analytical solution

Table 5.4: Computational times when using different grid sizes

In conclusion, in terms of computational times, the analytical solution, with or without the logistic transform, has a clear edge over the iterative scheme, on all fronts.

The main reason behind conducting the experiments in this section was to assess the impact of the coefficients that differentiate the kernel (3.2.16) used by Sarma, Durlofsky et al. (2008) and the inhomogeneous kernel (3.5.18), proposed as substitute in this report. The results presented above allow us, not only to conclude that their effect is not detrimental, but also that the analytical solution produces sharper results and is more efficient than the iterative scheme.

Therefore, the data assimilation experiments to be presented in the remainder of this report all use analytical preimages in their kernel parameterizations.

5.2. Comparative history matching results

The experiments in this paragraph aim to study the behaviour of the different data assimilation algorithms, discussed in the previous chapters of the report. More specifically, we will match the classic EnKF against the Iterative KPCA-EnS (kernel degree 3) and the Subspace EnKF (kernel degree 1) on two channelized reservoirs.

However, before moving on, we will first provide an overview of the data flow in the history matching framework, depicted in figure 5.14.

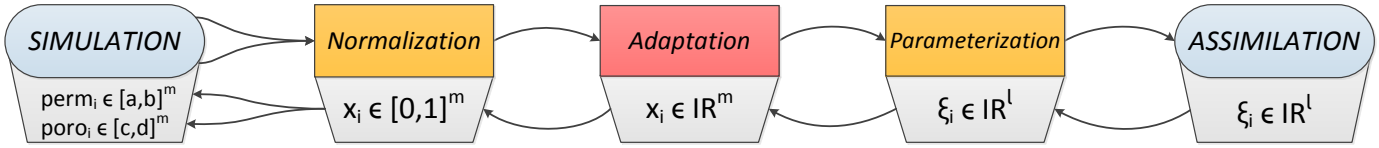


Figure 5.14: Data flow in the history matching framework

Recall from paragraph 2.3.4, that we chose to reduce the size of the state vector by replacing permeability and porosity with a value in $[0, 1]$, which represents the likelihood that the respective grid cell is in a channel and is done via *normalization*. After that we use an *adaptation* step, whose purpose is to constrain the assimilation results to their feasible ranges – after the discussion in paragraph 2.3.4, we decided that the logistic transform is preferable for this purpose, since truncation has a detrimental impact on the ensemble error covariance. Finally, before entering the assimilation cycle, the state vector undergoes *parameterization*, which stands to address the EnKF's limitation in handling higher-order statistical moments (see chapter 3). Also note, from the experiments in paragraph 5.1.4, that we need to use a sufficient number of training samples to counteract the thresholding effect of *logit* on the KPCA preimages.

With these in mind, we are now ready to introduce the first experiment.

5.2.1. The *Curved reservoir* experiment

The field used in this experiment is shown in figure 5.15. Notice that there is one injector and four producer wells, most of which are placed inside the channel. The exception is $Prod_3$, which we expect not to have a significant contribution in the field's production data, since the background shale (table 2.1) is close to impermeable. Even so, its position introduces an additional constraint in the data assimilation cycle and it is interesting to see how this translates into the results.

The challenge, here, is to determine the shape of the lower curved channel, which may prove difficult, given the considerable distance between Inj_1 and $Prod_1$. As we have seen in the plots from paragraph 2.1, a major event in channelized reservoir production is the water breakthrough and, in the case of $Prod_1$, it will occur in the later stages of the data assimilation cycle. This is also the reason for the increase in injector BHP and the longer observation and assimilation intervals in the experimental setup (table 5.5), when compared to that of the previously studied *Y-channel reservoir* (tables 2.2 and 3.1).

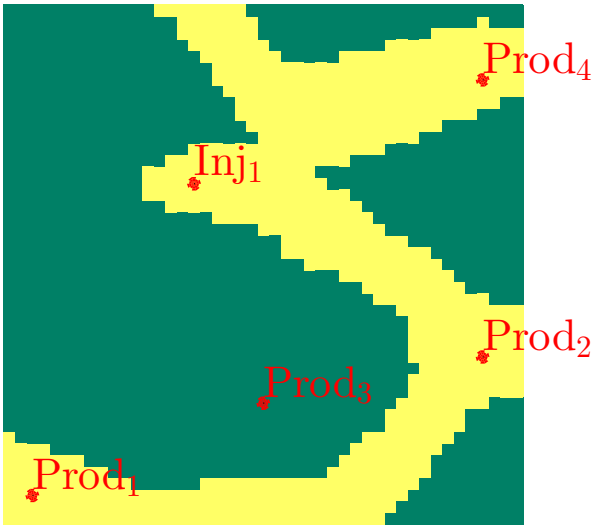


Figure 5.15: The *Curved reservoir* with well locations

As before, we imposed hard data constraints at the well locations and this can be clearly seen in the prior mean and variability (figure 5.16a). We also notice that these constraints, together with the proximity between Inj_1, $Prod_2$ and $Prod_4$, makes the upper part of the channel appear frequently among the initial ensemble members. This is also reflected in the oil rate registered at the producers (figure 5.16b), where we see the lack of spread in production response for $Prod_2$ and $Prod_4$.

As expected, $Prod_3$ has insignificant oil rate values, due to its position in the background shale, and, given the hard data constraints, the situation is unlikely to change during assimilation. However, the spread in its production response will still be affected by the update, allowing us to detect a potential ensemble collapse.

Finally, we see that $Prod_1$ registers more spread than the other wells due to the higher uncertainty regarding the surrounding channel's shape. We would also like to point out that the oil rate registered by running the simulation with the ensemble mean (the blue line) is outside the range of the ensemble (the grey lines). This is most likely due to the discrepancy in permeability between the channel and the background – after averaging, some grid cells, indicated as impermeable by the majority of the ensemble members, might become slightly permeable in the mean, and this can have a dramatic impact on reservoir fluid flow. Verlaan and Heemink (2001) developed this remark into a measure for model nonlinearity.

EnKF

The results of the EnKF on this reservoir are shown in figure 5.17a, where we see that the general shape of the channel is correctly identified, albeit not its curvature. However, there is a good amount of posterior variability, which leaves room for possible adjustments.

The oil production plots (figure 5.17b) show an improvement over the prior, in the sense that the response of the members was "pulled" towards the truth. This is particularly evident for $Prod_1$, for which the truth is now well within the range of the ensemble. The same plot also reveals that, due to high model nonlinearity, the posterior mean might not be the best estimator for the true state.

Item	Description
Grid size	$45 \times 45 \times 1$ cells
Physical size of 1 grid cell	$15m \times 15m \times 2m$
Snesim search ellipsoid	$10 \times 10 \times 1$ grid cells (2D isotropic)
Target marginal distribution for facies	50% shale, 50% sand
Hard data constraints	facies at well locations
Ensemble size	100 members
Initial reservoir pressure	$p_0 = 100$ bars
Initial reservoir saturation	20% water, 80% oil (20% residual oil)
Injector well control	BHP $= 2.5 \cdot p_0$
Producer well control	BHP $= p_0$
Observed variables	water and oil rates in wells
Observation uncertainty	$r = \max(10\% \cdot \text{measured value},\ 0.1)$
Observation interval	6 months
Assimilation interval	12 months
Simulation time	108 months
Training set size	$n_t = 900$ samples (per subspace)
Kernel degree	$d = 3$ for Iterative KPCA-EnS $d = 1$ for Subspace EnKF
Parameterization variance threshold	$q = 90\%$
Adaptation method	*logit*

Table 5.5: Setup for the *Curved reservoir* experiment

Iterative KPCA-EnS

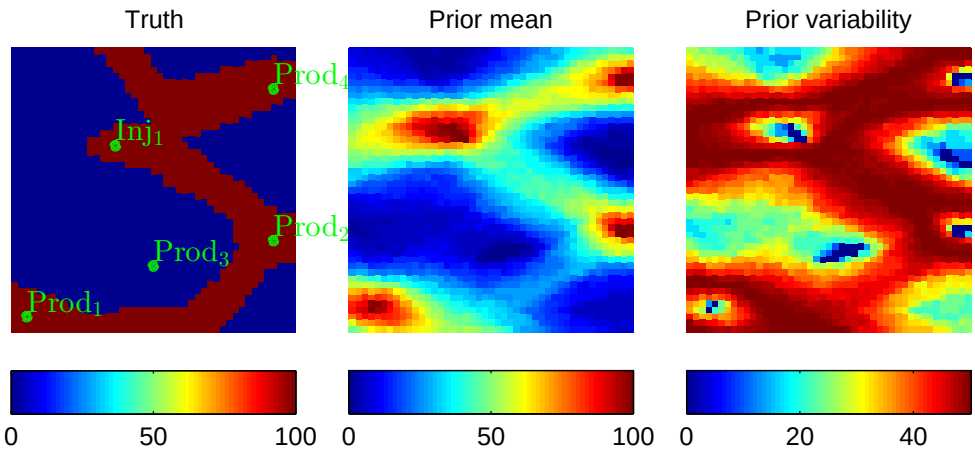
The KPCA Ensemble Smoother with degree 3 finds a similar channel shape (figure 5.18a) as the one obtained with the EnKF. Also, by accounting for higher order moments, the integrity of the channel structure and the sharpness of its boundary is well preserved. However, the curvature is not well represented in the mean and, this time, the ensemble variability in the lower half of the field is not sufficient to permit significant deviations from the suggested shape.

This is confirmed by the significantly reduced spread in the production plot for $Prod_3$ (figure 5.18b). We also notice the development of an offset between the ensemble and the truth in the oil rate of $Prod_1$. This leads us to believe that an additional increase in the number of updates may, eventually, lead to collapse, with the posterior ensemble drifting even further away from the truth.

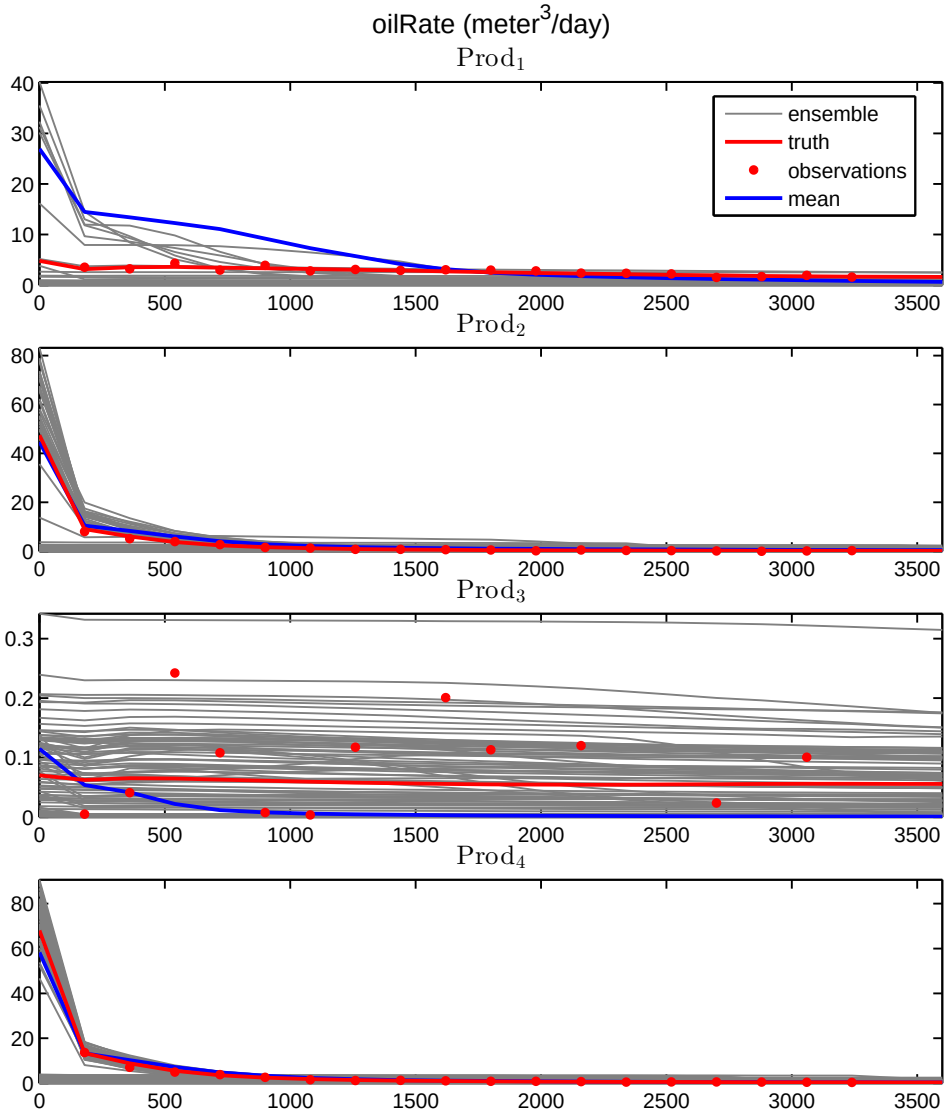
Subspace EnKF

The Subspace EnKF, with 5 subspaces and $d = 1$, seems to combine the advantages of the previous two algorithms. Indeed, from figure 5.19a, we see that its output has a smooth channel structure, while maintaining the sharpness of the boundary between shale and sand. Also, the ensemble variability is sufficient to account for the imperfections in shape(for example, the newly-appeared branch in the lower channel segment) and curvature in the posterior mean.

Moreover, the oil rate plots (figure 5.19b) exhibit a similar behaviour to that obtained with the EnKF (figure 5.17b) – for $Prod_1$, the production response of the ensemble is centered around the truth and sufficient spread is maintained for all wells.

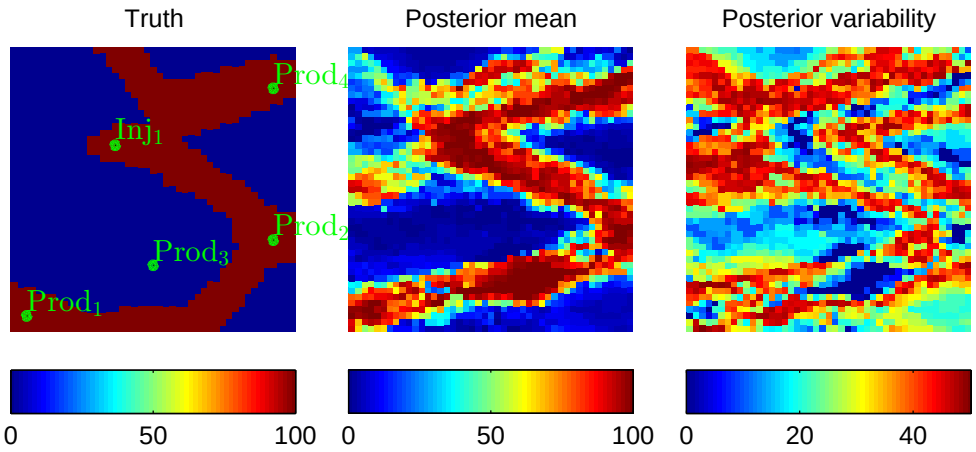


(a) True permeability field, ensemble mean and variability

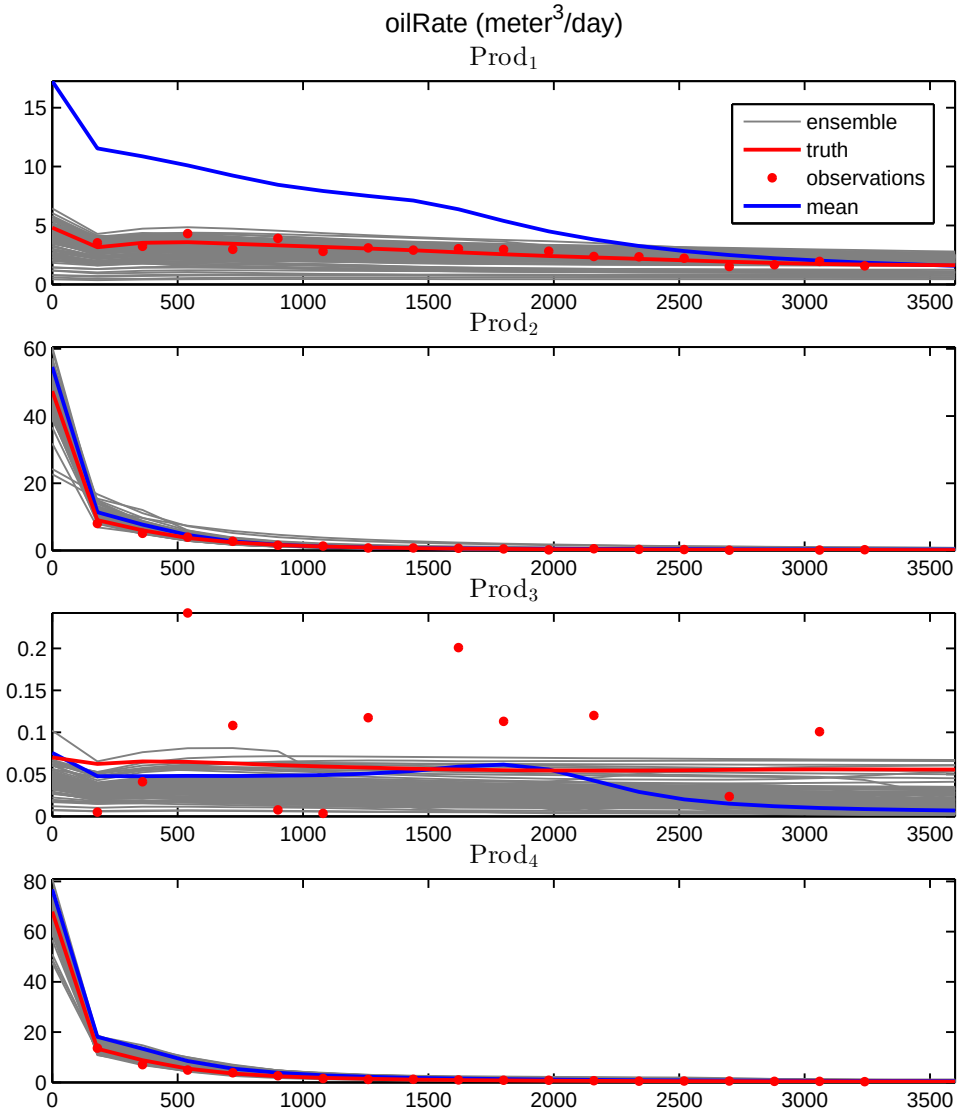


(b) Produced oil rates

Figure 5.16: Prior information for the *Curved reservoir* experiment

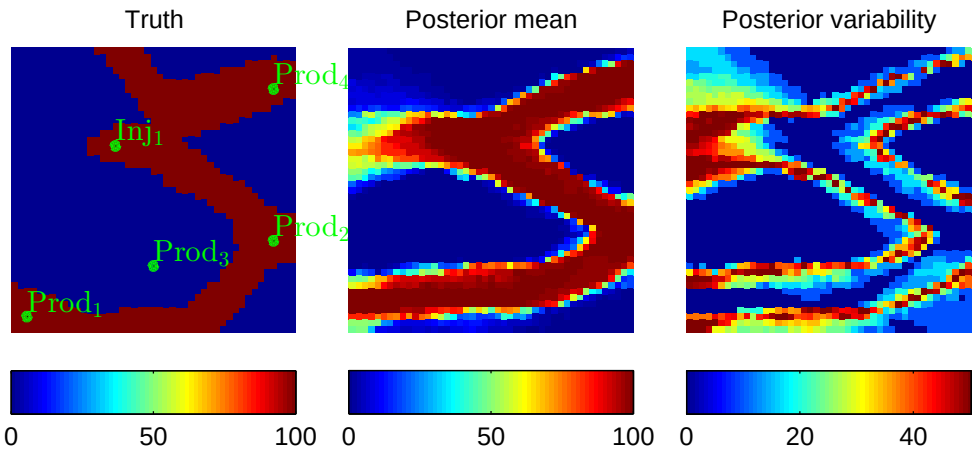


(a) True permeability field, ensemble mean and variability

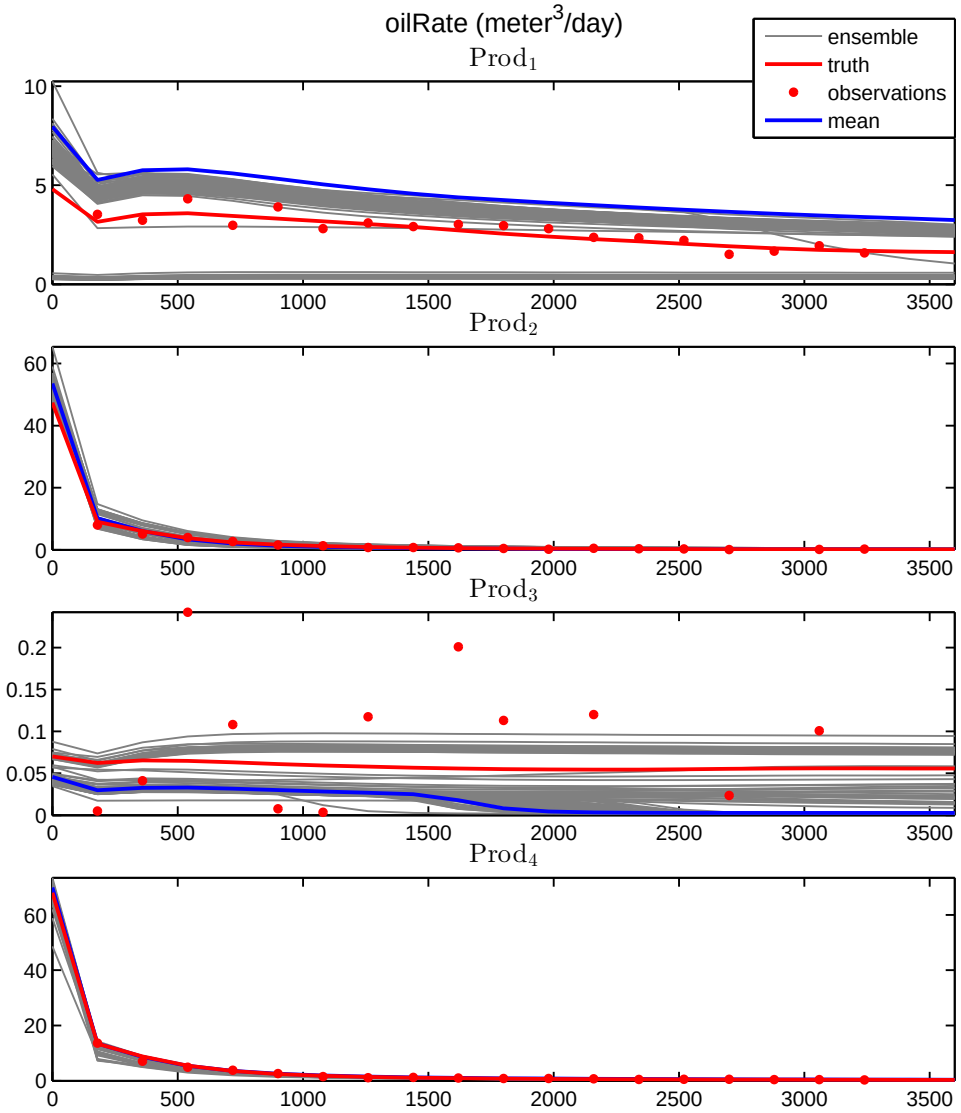


(b) Produced oil rates

Figure 5.17: Result of the EnKF on the *Curved reservoir*

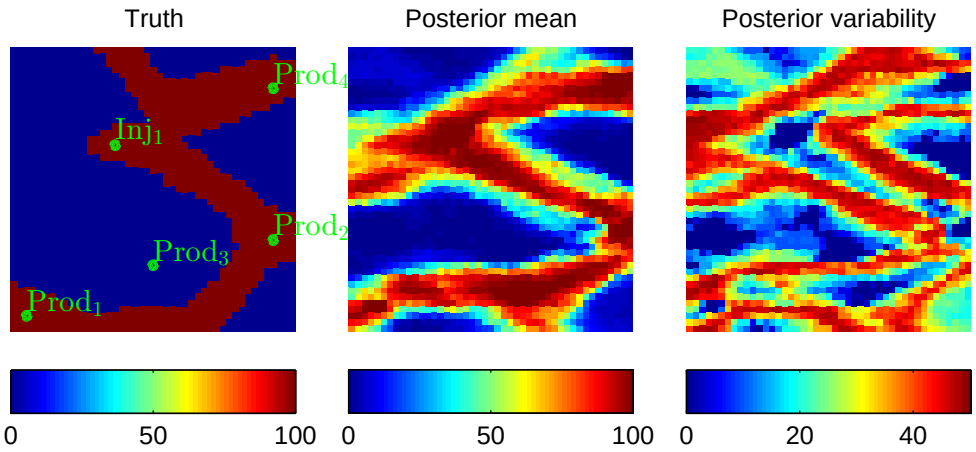


(a) True permeability field, ensemble mean and variability

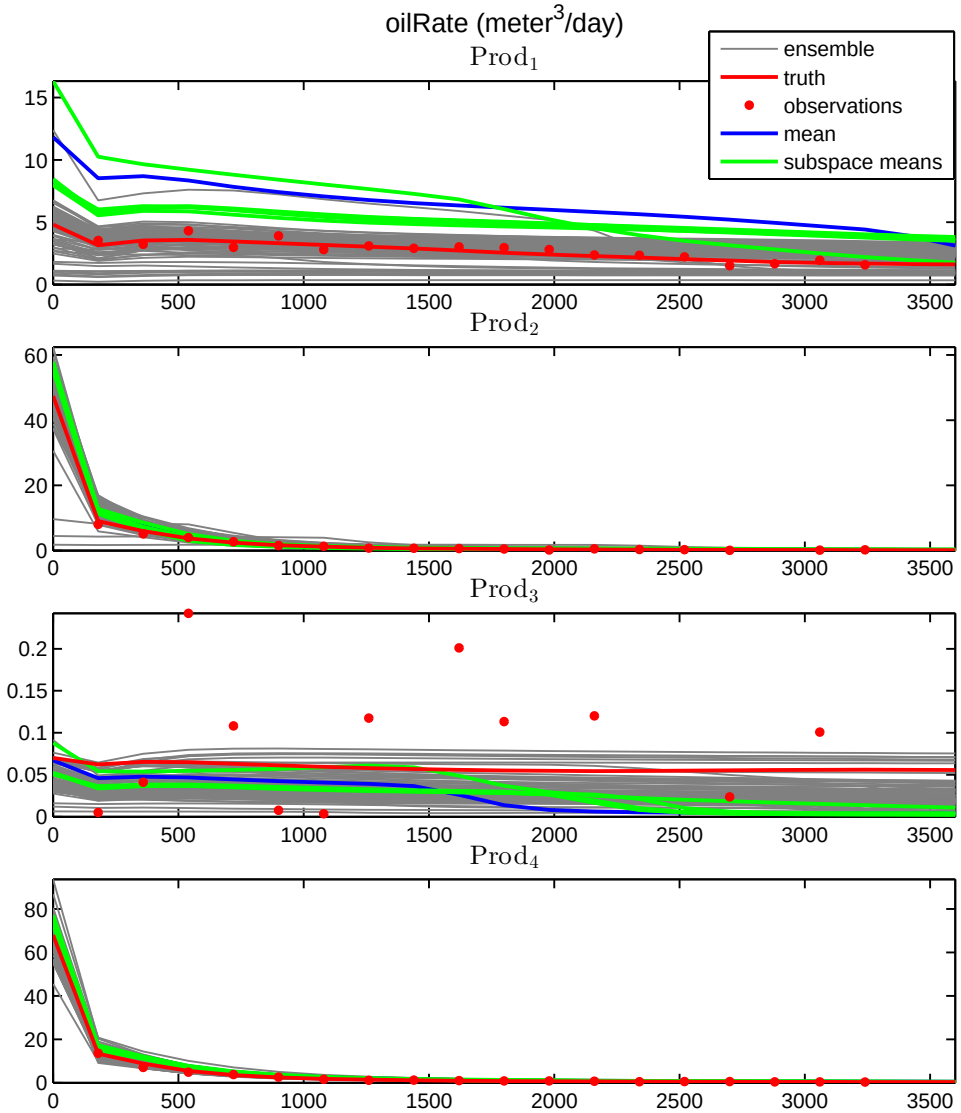


(b) Produced oil rates

Figure 5.18: Result of the Iterative KPCA-EnS on the *Curved reservoir*



(a) True permeability field, ensemble mean and variability



(b) Produced oil rates

Figure 5.19: Result of the Subspace EnKF with 5 subspaces on the *Curved reservoir*

The conclusion after this experiment is that the Subspace EnKF is a viable alternative for history matching, despite the strong assumptions in its theoretical background and apparent inability to operate directly with higher-degree kernels. Also, the classic EnKF performs remarkably well and we were surprised by the variability in its posterior ensemble, which was not affected significantly, in this particular case, by the model nonlinearity. Finally, from a theoretical perspective, the Iterative KPCA-EnS is attractive for its handling of high order statistical moments, however, even with the smoother formulation, its results seem to be plagued by ensemble collapse.

5.2.2. The *Ribbon reservoir* experiment

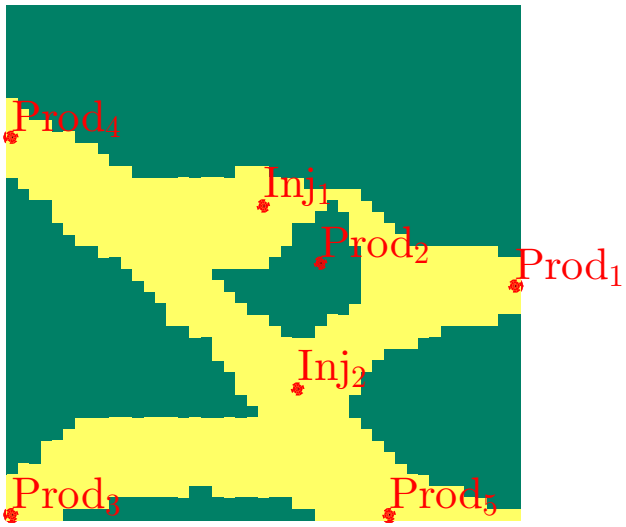


Figure 5.20: The *Ribbon reservoir* with well locations

We now turn towards the *Ribbon reservoir*, which was derived from sample 506 (figure 5.7a), and populated with two injectors and five producers, one of which ($Prod_2$) is placed in the central shale enclave (see figure 5.20).

As mentioned before, this is an atypical channel shape and, with it, we aim to design a stress-test for the three data assimilation algorithms under review. The experiment setup is detailed in table 5.6, and the most important remark is that we have dropped the hard data constraints, both for the ensemble members and the parameterization training samples.

Consequently, the prior ensemble mean is nearly uniform and there is high variability in all grid blocks (figure 5.21a). Therefore, it may seem surprising that the production plots (figure 5.21b) show less spread than the ones obtained using conditioned samples (figure 5.16b). However, in this type of reservoirs, the probability that two cells are connected by a channel is lower than the converse and, in order to operate, a producer well needs a connection to at least one injector. The plots also reveal that none of the members has a connection between $Prod_2$ and either injector. This was not deliberate, but, fortunately, it is in line with the truth.

EnKF

The posterior mean produced by the EnKF (figure 5.22a) does have some resemblance to the truth, however, instead of channels, its structure is composed mainly of disconnected permeable "blobs". Also, the ensemble variability, which was very high in the prior, is mostly lost after the assimilation, indicating collapse.

Item	Description
Grid size	$45 \times 45 \times 1$ cells
Physical size of 1 grid cell	$15m \times 15m \times 2m$
Snesim search ellipsoid	$10 \times 10 \times 1$ grid cells (2D isotropic)
Target marginal distribution for facies	50% shale, 50% sand
Hard data constraints	none
Ensemble size	100 members
Initial reservoir pressure	$p_0 = 100$ bars
Initial reservoir saturation	20% water, 80% oil (20% residual oil)
Injector well control	BHP $= 1.5 \cdot p_0$
Producer well control	BHP $= p_0$
Observed variables	water and oil rates in wells
Observation uncertainty	$r = \max(10\% \cdot \text{measured value},\ 0.1)$
Observation interval	1 month
Assimilation interval	4 months
Simulation time	34 months
Training set size	$n_t = 1500$ samples (per subspace)
Kernel degree	$d = 3$ for Iterative KPCA-EnS $d = 1$ for Subspace EnKF
Parameterization variance threshold	$q = 90\%$
Adaptation method	*logit*

Table 5.6: Setup for the *Ribbon reservoir* experiment

If we examine the production plots (figure 5.22b), we see that, apart from $Prod_3$, which is the well farthest away from the injectors, the spread has been greatly reduced. As a result, the water breakthrough is not well captured by the ensemble (especially for $Prod_1$ and $Prod_4$).

Iterative KPCA-EnS

The *Ribbon reservoir* proves to be a challenge also for the KPCA Ensemble Smoother with $d = 3$. The posterior ensemble is collapsed, this time on a local optimum (figure 5.23a), in which the reservoir is split into two parallel channels – the upper one connects Inj_1 to $Prod_1$ and $Prod2$, while the lower one connects Inj_2 to $Prod_3$ and nearly misses $Prod_5$.

The production plots (figure 5.23b) show almost no spread and the truth is outside the ensemble range for three out of the five producer wells.

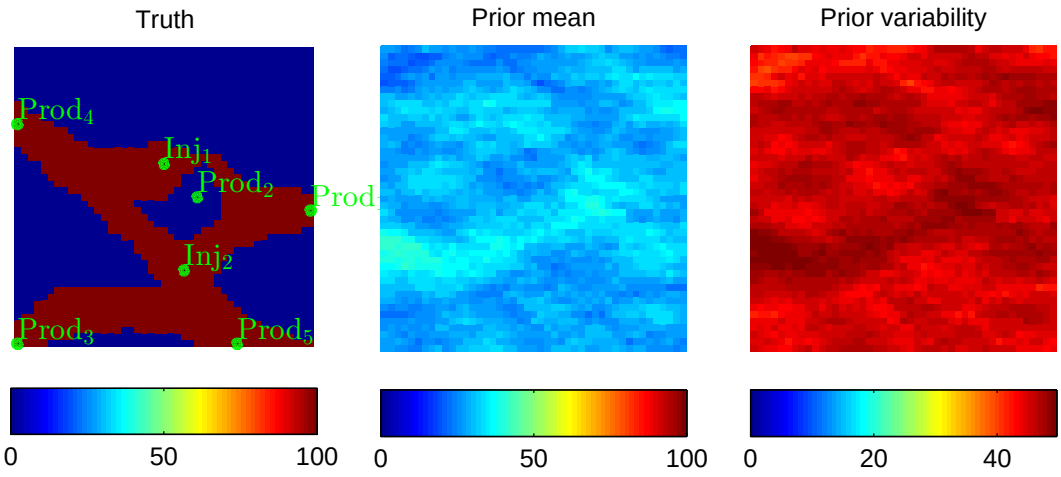
Subspace EnKF

The posterior mean obtained using the Subspace EnKF with 5 subspaces and $d = 1$ (figure 5.24a) seems to be a smoothed-out version of the EnKF result (figure 5.22a). As such, it does bear similarity to the truth and we also see that the variability is better preserved and concentrated in the central area. This is somewhat unfortunate; for example, the algorithm is certain about the existence of the upper channel, which is, in fact, not present in the truth.

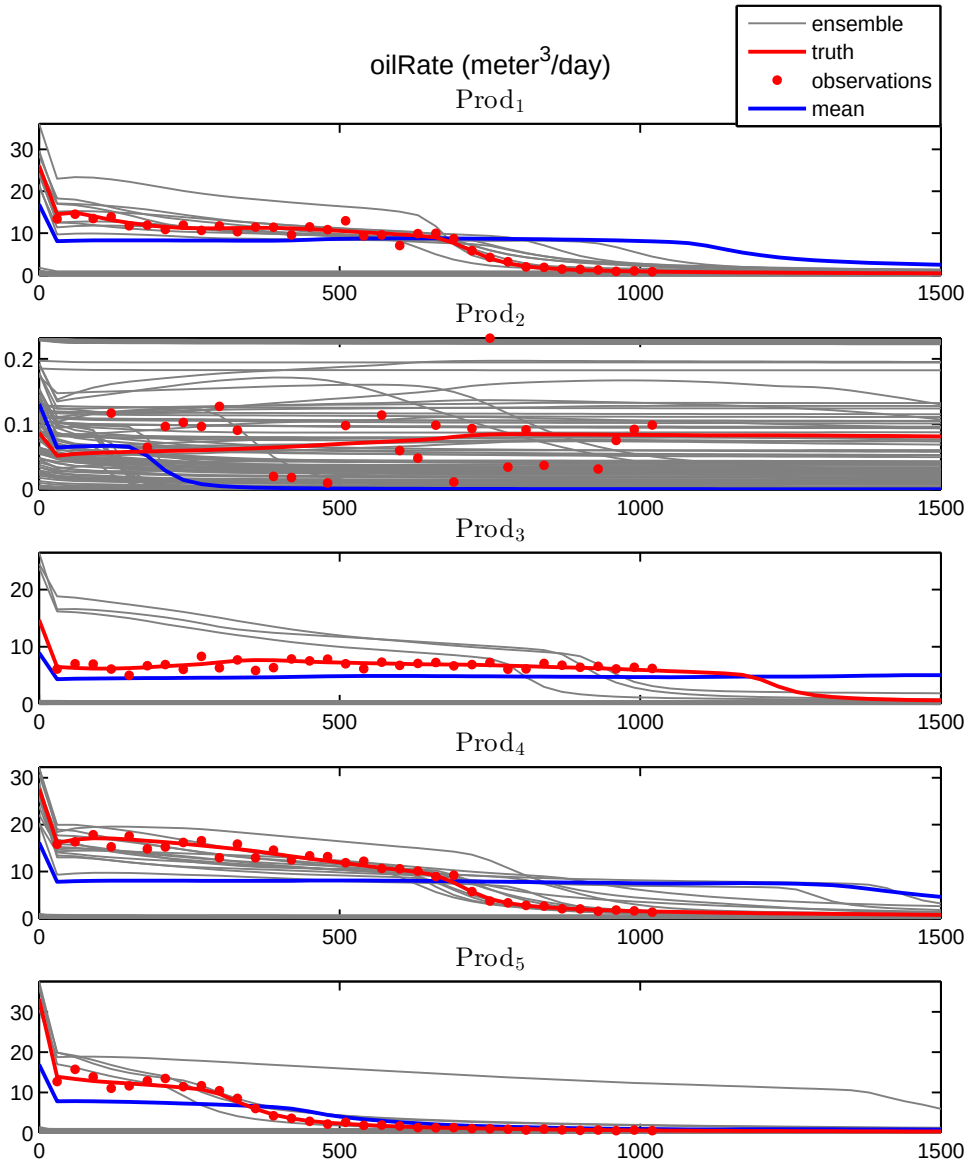
In order to assess the impact this has on production, we turn to figure 5.24b, which shows an obvious increase in spread, when compared to the corresponding plot from the EnKF (figure 5.22b).

The truth is captured in the ensemble range for almost all producer wells. The only exception is $Prod_4$, for which none of the members register water breakthrough, in contrast to the truth, for which this does occur at roughly 750 days. In order to provide an explanation, we turn back to the true permeability field (figure 5.24a), where we see that $Prod_4$ (upper-left) is connected to both injectors, with Inj_1 being the closest one. However, in the posterior mean, the channel between the two is severed and the production from $Prod_4$ is, now, mainly due to the more distant Inj_2.

In conclusion, the *Ribbon reservoir* poses difficulties for all three data assimilation algorithms. The Iterative KPCA-EnS performs the worst due to obvious ensemble collapse. Between the remaining two, we prefer the Subspace EnKF, which produces results with channel-like features, while retaining some of the initial variability.

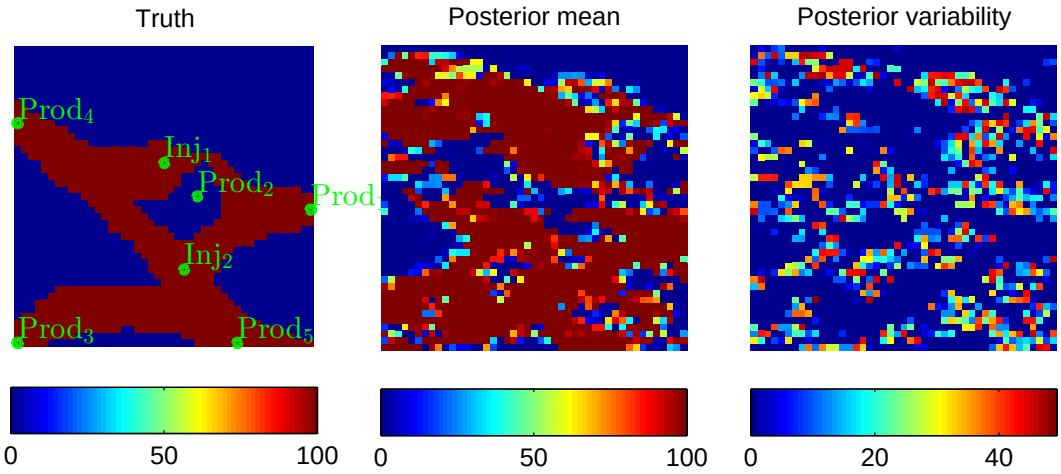


(a) True permeability field, ensemble mean and variability

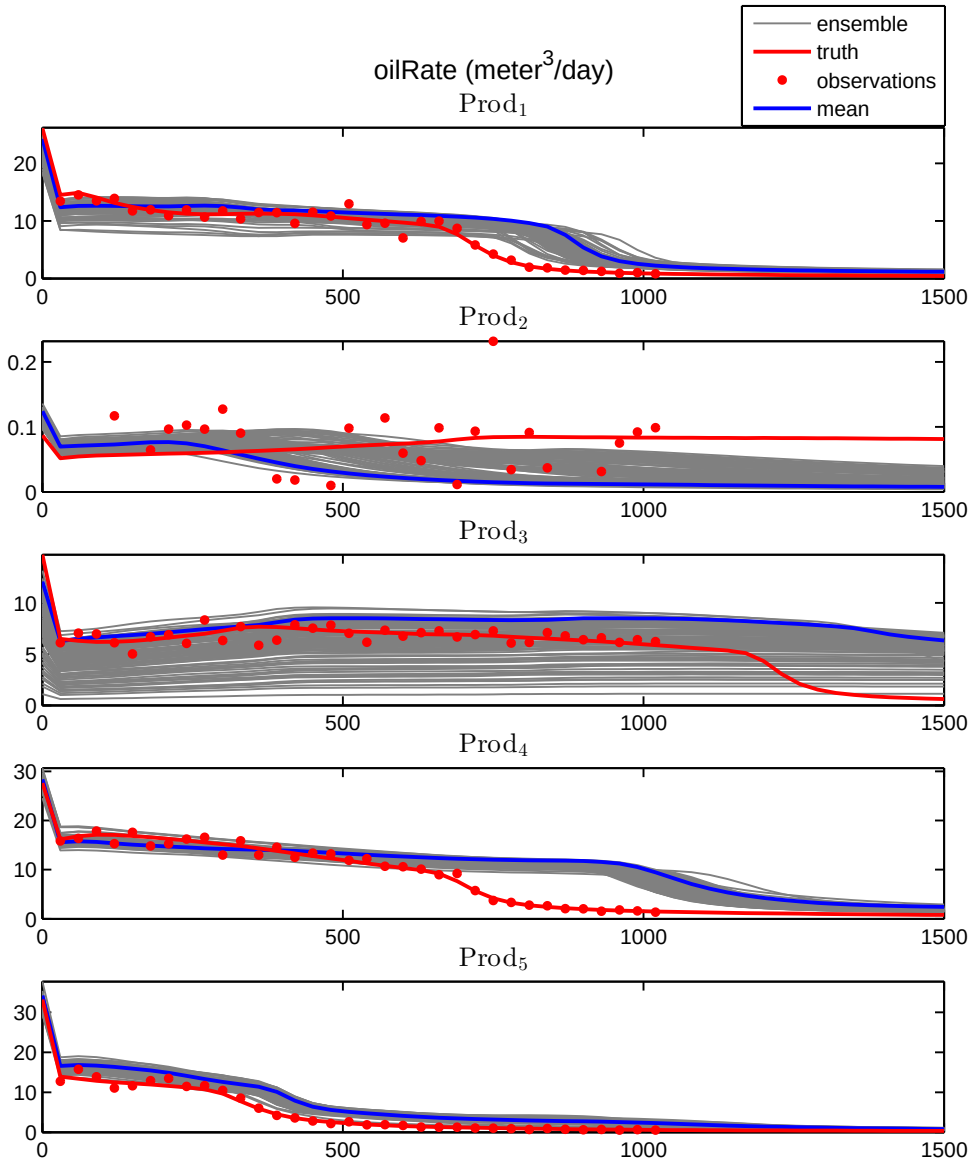


(b) Produced oil rates

Figure 5.21: Prior information for the *Ribbon reservoir* experiment

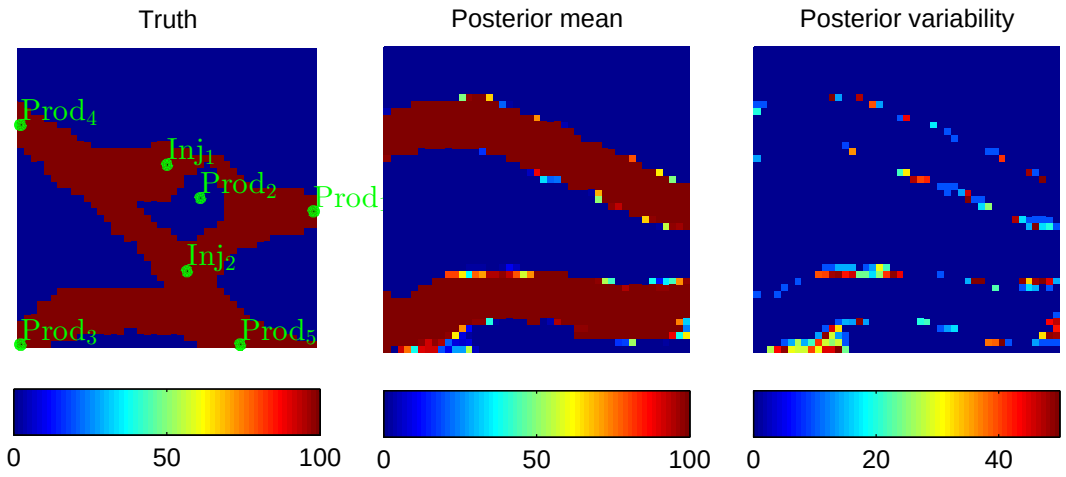


(a) True permeability field, ensemble mean and variability

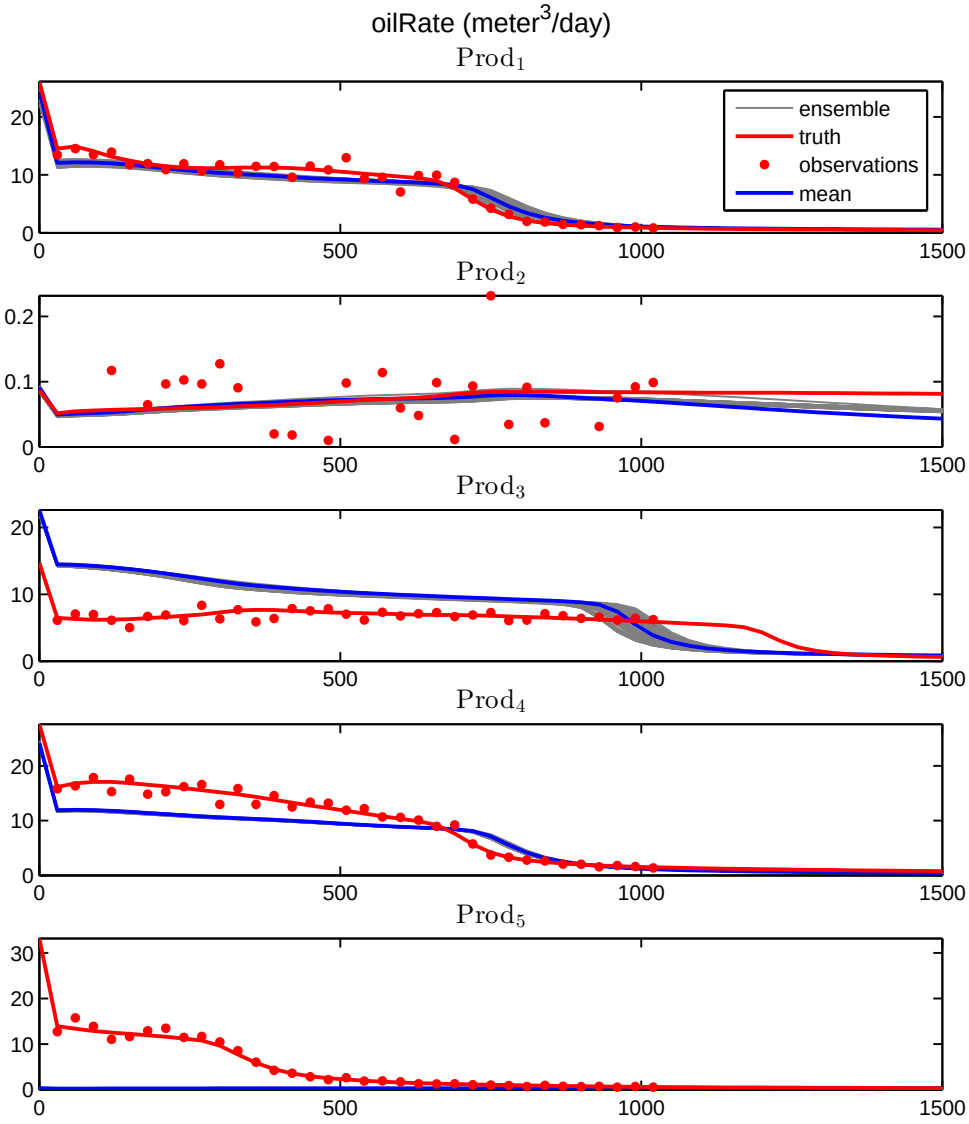


(b) Produced oil rates

Figure 5.22: Result of the EnKF on the *Ribbon reservoir*

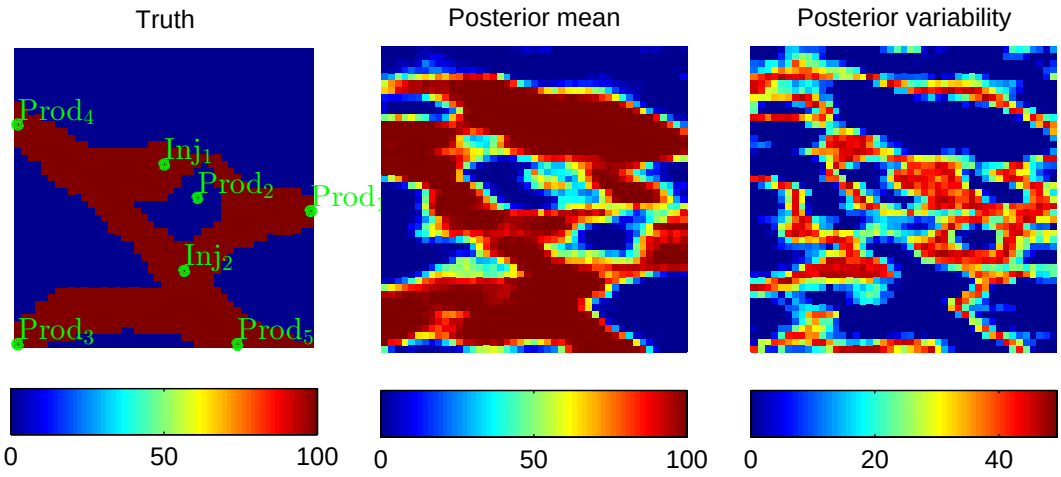


(a) True permeability field, ensemble mean and variability

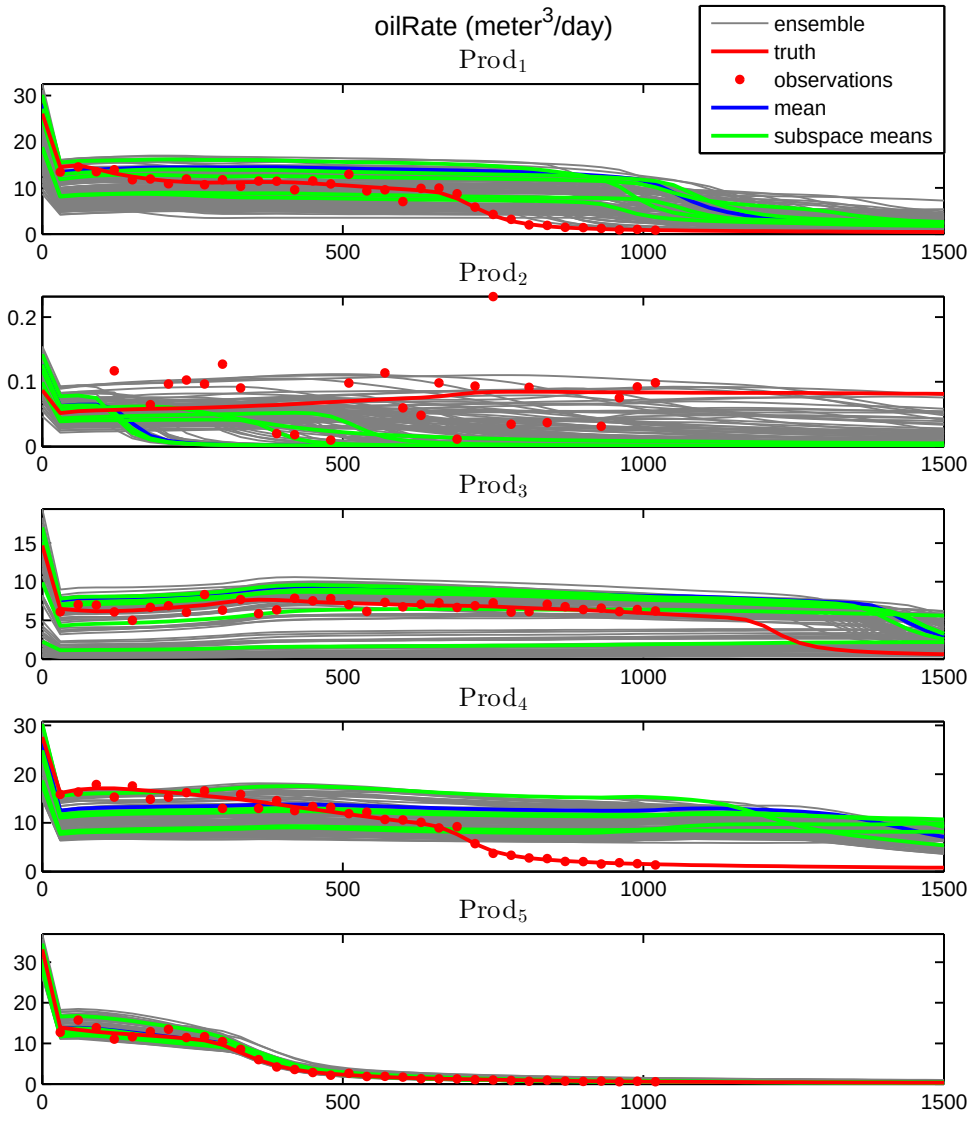


(b) Produced oil rates

Figure 5.23: Result of the Iterative KPCA-EnS on the *Ribbon reservoir*



(a) True permeability field, ensemble mean and variability



(b) Produced oil rates

Figure 5.24: Result of the Subspace EnKF with 5 subspaces on the *Ribbon reservoir*

5.3. Study on the Subspace EnKF

Encouraged by its performance in the previous data assimilation experiments, we dedicate the remainder of the chapter to the Subspace EnKF, in order to provide answers to the last two research topics formulated in paragraph 1.2.

5.3.1. Sensitivity to the number of subspaces

During the *Ribbon reservoir* experiment, we used a set of 7500 unconstrained channelized reservoir samples to train the parameterizations of the Subspace EnKF (1500 samples for each of the 5 subspaces). Since we want to assess the impact of the number of subspaces on assimilation results, we will divide the same set, evenly, for the training of 2, 10, 25 and, finally, 50 subspaces. In all other aspects, the setup is the same as described in table 5.6, including the 100 ensemble members, which, just like the training samples, will be equally split across the subspaces.

2-Subspace EnKF

Since we decided to partition the whole set of samples evenly, the parameterizations of 2 subspaces will be slightly over-trained (3750 samples for each), in comparison to the usual setup. Nevertheless, the training set, together with the ensemble, contributes to the volume of prior information available to the (parameterized) data assimilation scheme and, for the validity of our argument, we need to keep it intact.

The results are given in figure 5.25a and we recognize, yet again, that the structure of the posterior mean bears some resemblance to the truth. However, the channels are distorted and the ensemble variability seems to be diminished slightly, in comparison to figure 5.24a.

After analyzing the production plots (figure 5.25b), we see that, this time, the water breakthrough in both $Prod_1$ and $Prod_4$ is incorrectly represented by the ensemble. We also notice an overall reduction in spread (see, for example, $Prod_4$ and its corresponding plot in figure 5.25b).

In conclusion, a smaller number of subspaces seems to result in a reduction of ensemble variability, in terms of both geological structure and production.

10-Subspace EnKF

With 10 subspaces, the result (figure 5.26a) seems to deviate slightly from the general shape of the *Ribbon reservoir*. Most notably, the channel connecting $Prod_4$ to Inj_2 is severed in all members (see variability) and this is reflected in its production plot (figure 5.26b), where despite the high spread, the water breakthrough in the ensemble is delayed considerably (∼500 days) from that of the truth.

On the positive side, the increased number of subspaces seems to enhance the overall spread in ensemble oil production, when compared to the previous cases (figures 5.25b and 5.24b). In terms of geological structure, the amount of variability seems to be comparable to that obtained with 5 subspaces (figure 5.24a).

25-Subspace EnKF

25 subspaces takes us to the opposite side of the spectrum, where the parameterizations are under-trained (300 samples for each). Nevertheless, the posterior mean is a fair representation of the truth, especially when paired with the variability (figure 5.27a), significantly increased when compared to the previous results.

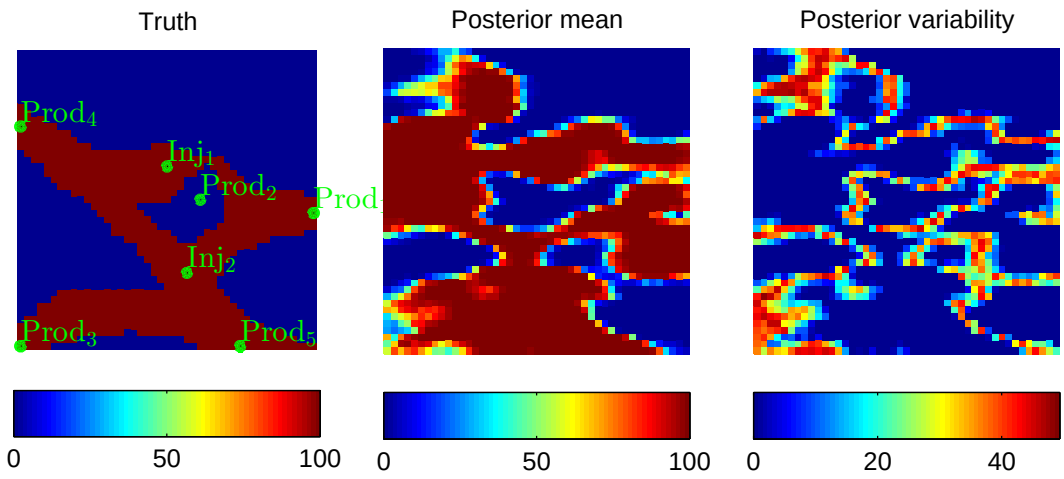
The improvement is also visible in the production plots (figure 5.27b), where we notice that, this time, the ensemble has less delay in representing the water breakthrough in $Prod_1$ and $Prod_4$. In terms of production variability, it is surprising to see that the present result has, overall, surpassed the prior ensemble (figure 5.21b). An explanation can be that, after the Kalman updates, the permeability may attain intermediate values between that of the two facies, thus increasing the spectrum of possible production trends.

50-Subspace EnKF

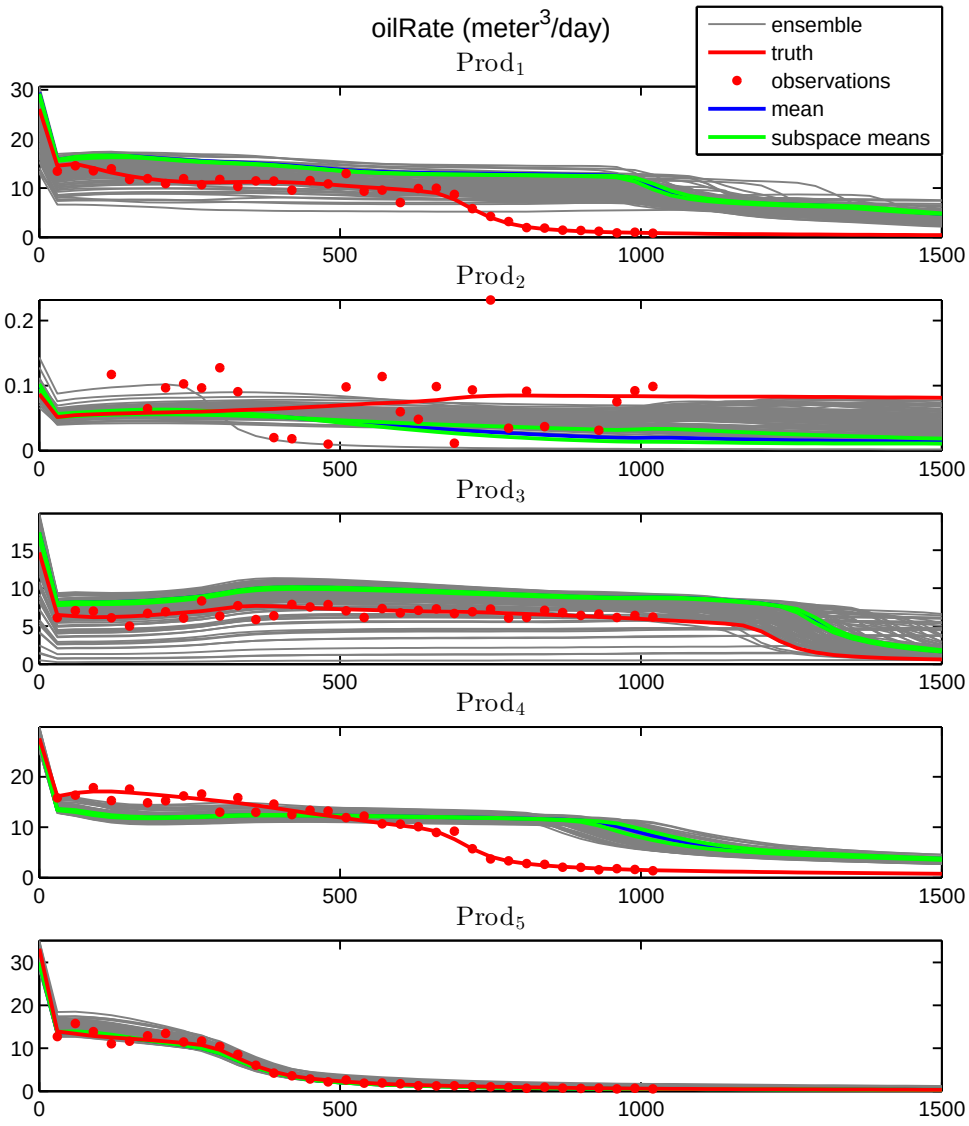
Note that, with 50 subspaces, each parameterization only has 150 training samples at its disposal. We see that this has an impact on the posterior mean (figure 5.28a), in the sense that the features lose curvature and, instead, tend to become parallel to each other (due to the reduced number of kernel principal components). Despite this, the production responses register a further increase in spread (figure 5.28b) and the ensemble captures the truth appropriately in all wells.

In conclusion, the experiments show that the number of subspaces has a positive influence on the ensemble spread, both in terms of rock properties and production response. For future applications, it might even be beneficial to have a separate subspace for each ensemble.

Another aspect to consider is the number of training samples. Although our results are promising even in degenerate cases (see the previous experiment, with 50 subspaces), we expect that the structure of the estimated geological features will suffer if the parameterizations are insufficiently trained.

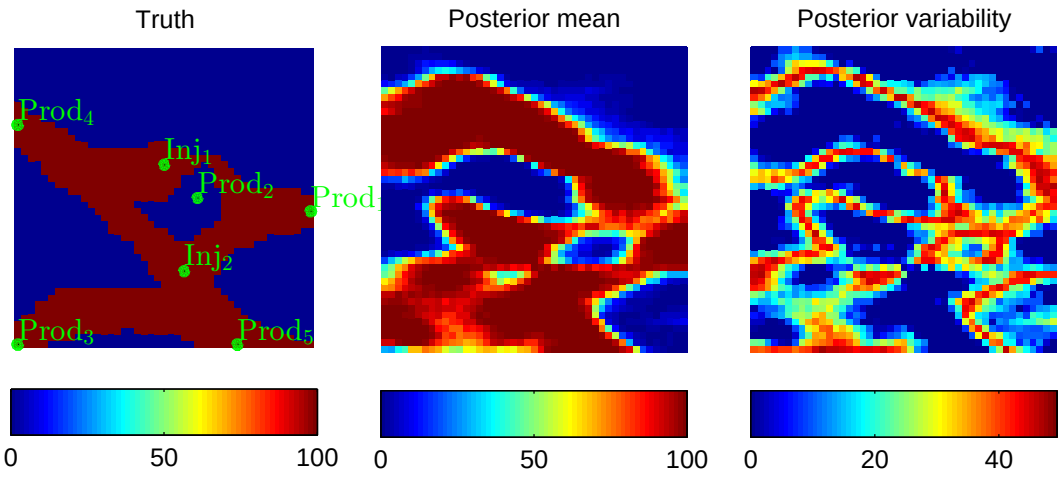


(a) True permeability field, ensemble mean and variability

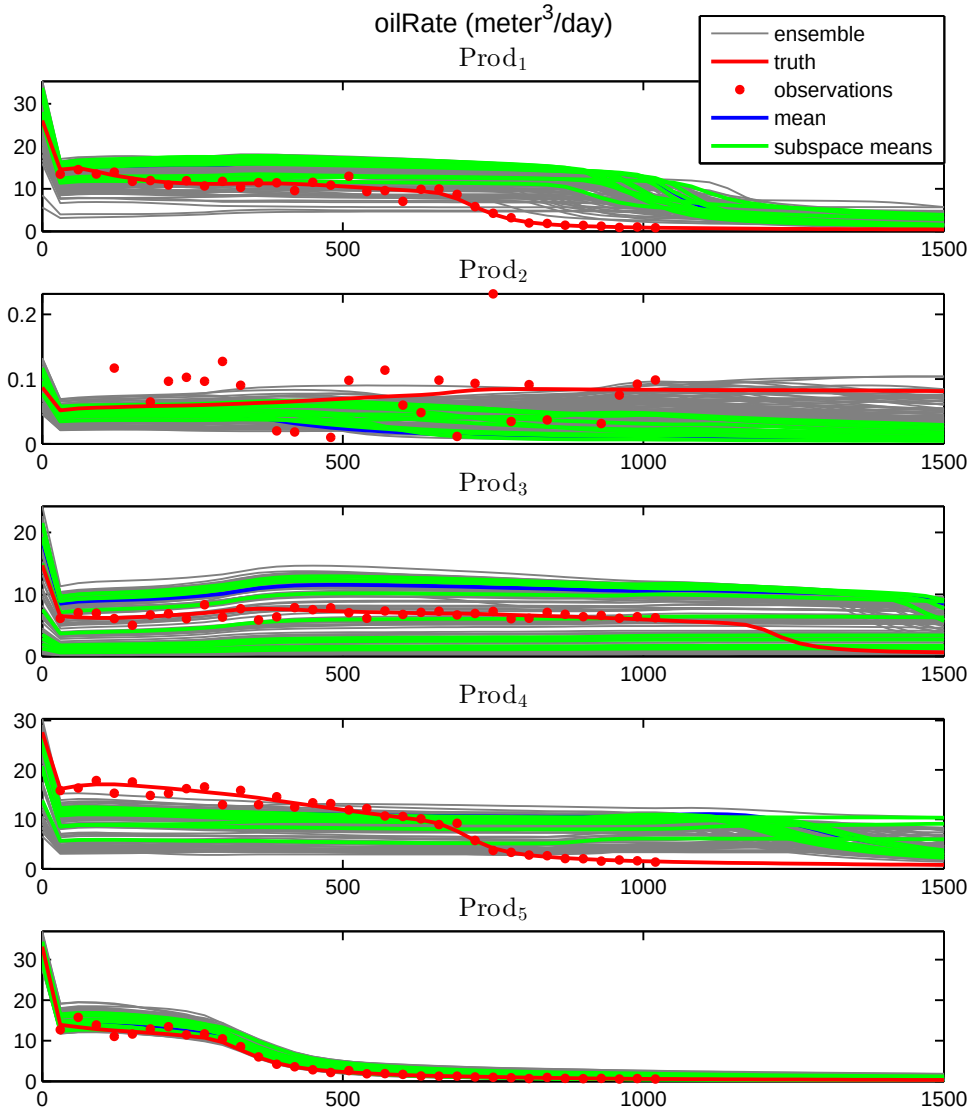


(b) Produced oil rates

Figure 5.25: Result of the Subspace EnKF with 2 subspaces on the *Ribbon reservoir*

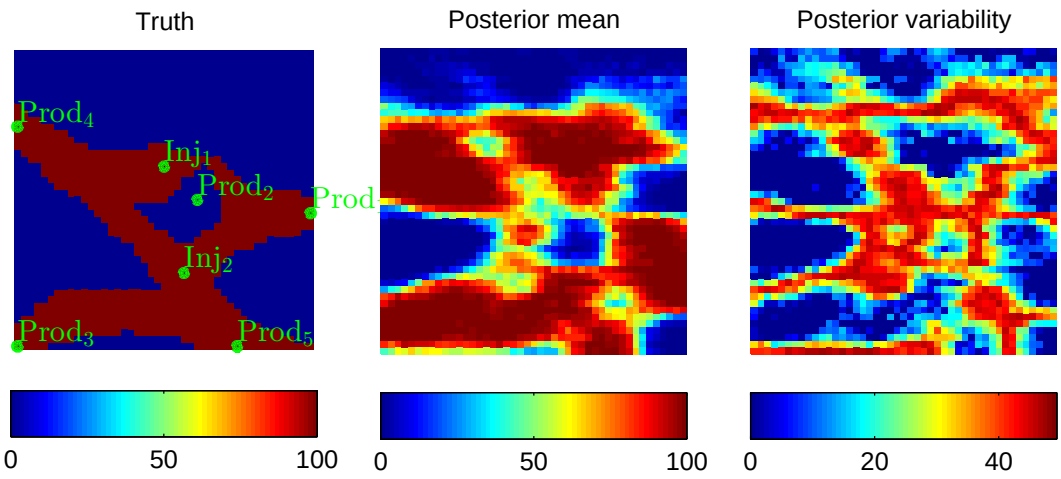


(a) True permeability field, ensemble mean and variability

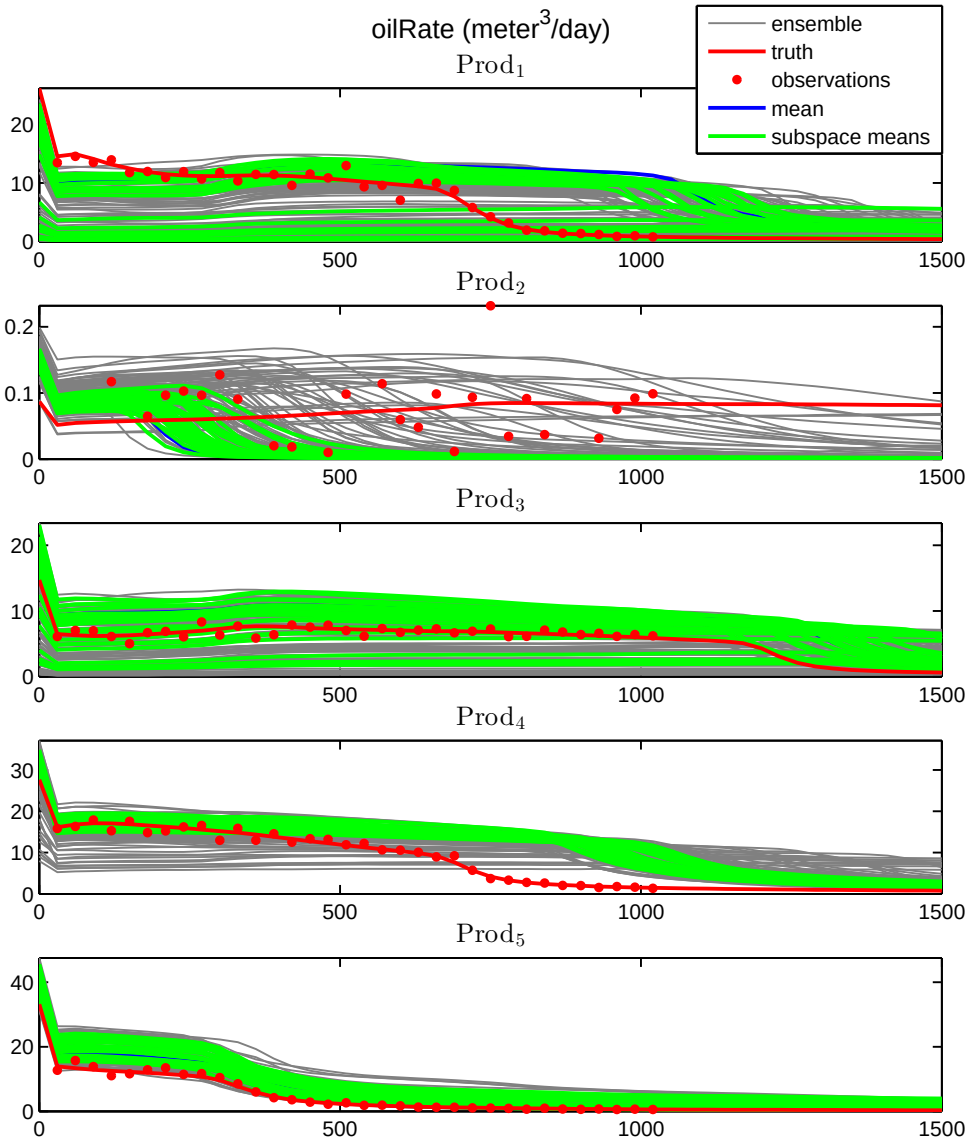


(b) Produced oil rates

Figure 5.26: Result of the Subspace EnKF with 10 subspaces on the *Ribbon reservoir*

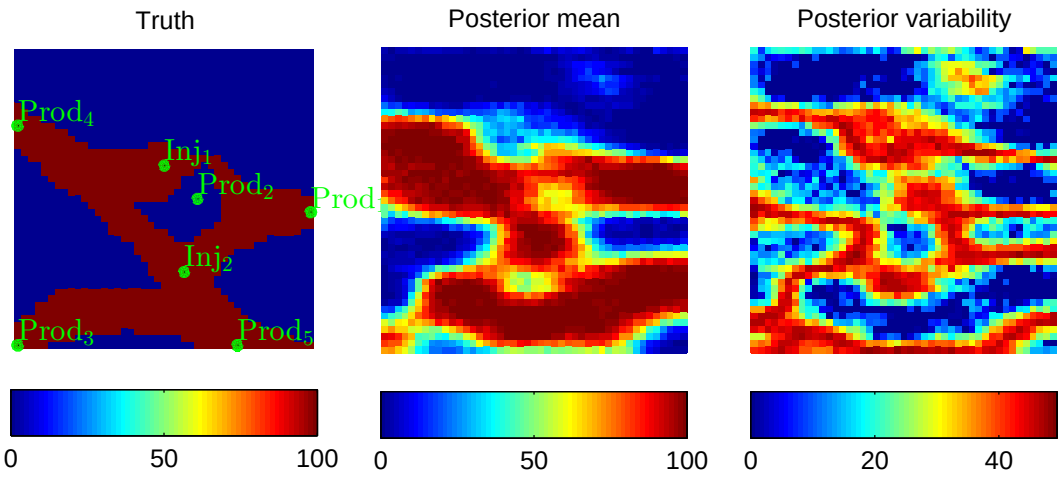


(a) True permeability field, ensemble mean and variability

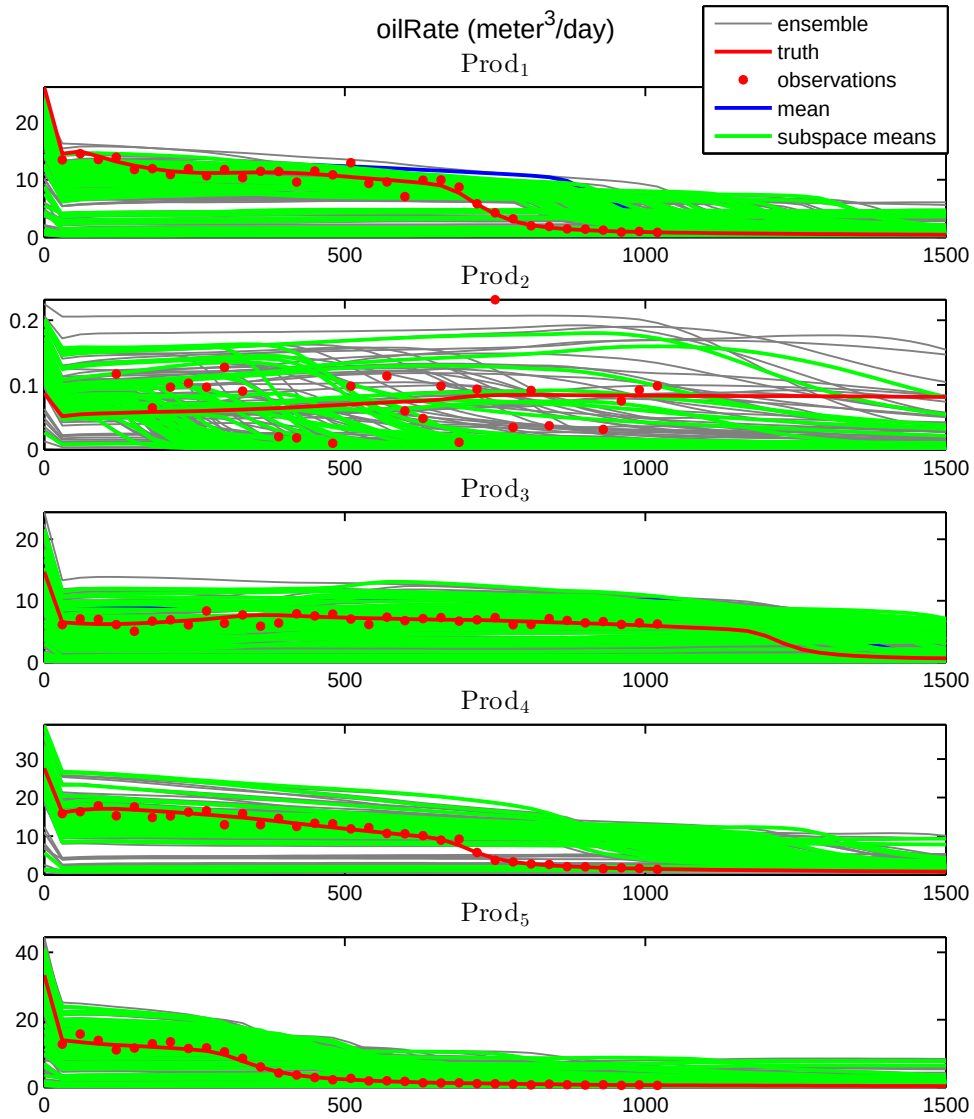


(b) Produced oil rates

Figure 5.27: Result of the Subspace EnKF with 25 subspaces on the *Ribbon reservoir*



(a) True permeability field, ensemble mean and variability



(b) Produced oil rates

Figure 5.28: Result of the Subspace EnKF with 50 subspaces on the *Ribbon reservoir*

5.3.2. Sensitivity to training set clustering

One of the main goals of this study was to develop a way to build the parameterizations which takes into account the prior knowledge about the geological structure of our reservoir. To this end, we propose to use clustering when determining the training sets for the subspaces, in the hope that the more specialized parameterizations, thus obtained, will amount to a greater variability in the posterior ensemble. A clear advantage is that this approach is not restricted to channelized reservoirs and can be generally applied when history matching with the Subspace EnKF.

With this in mind, the steps of the experiments in this paragraph are oultined in the following

Algorithm 5.2: Training set clustering experiment

Input: A set of $n = n_{clust} + n_t$ reservoir samples, $\mathbf{t}_1, \ldots, \mathbf{t}_n \in \mathbb{R}^m$
k, the number of desired subspaces.

Output: The partition of the n_t samples into k training sets for the subspace parameterizations.

① Use the trailing n_{clust} samples to train a separate kernel PCA parameterization, g_{clust} (for example, with degree $d = 3$).

② Apply this parameterization to the leading n_t samples, $\mathbf{t}_i$, in order to obtain the projection vectors, $\boldsymbol{\xi}_i = g_{clust}(\mathbf{x}_i)$.

③ Use the K-means clustering algorithm (see Forgy 1965; and Lloyd 1982) to partition the $\boldsymbol{\xi}_i$ into k groups.

④ Use the partition indices to split the original samples, $\mathbf{t}_i$, $i = 1, \ldots, n_t$, into k training sets.

where we chose $n_t = 7500$, just as before, $n_{clust} = 1400$ and, for efficiency reasons, we used the Matlab implementation by M. Chen (2012) instead of the built-in *kmeans* function.

In other respects, the experiments are identical to those performed in the previous paragraph.

2-Subspace EnKF

In this case, the K-means clustering algorithm partitioned the training set in $\{2058, 5442\}$ samples. By comparing the results, given in figure 5.29a, with those obtained without clustering (figure 5.25a) we notice a slightly greater variability. At the same time, the production plots (figure 5.29b) exhibit a significant increase in spread over all wells (figure 5.25b).

10-Subspace EnKF

For 10 subspaces, K-means split the training samples roughly as $\{2500, 9 \times 550\}$. The results (figure 5.30a) reveal a surprising increase in ensemble variability over the former experiment (figure 5.26a) and, also, the posterior mean is a better reflection of the truth. However, on the production plot side (figure 5.30b) the differences in spread are not that dramatic, in this case.

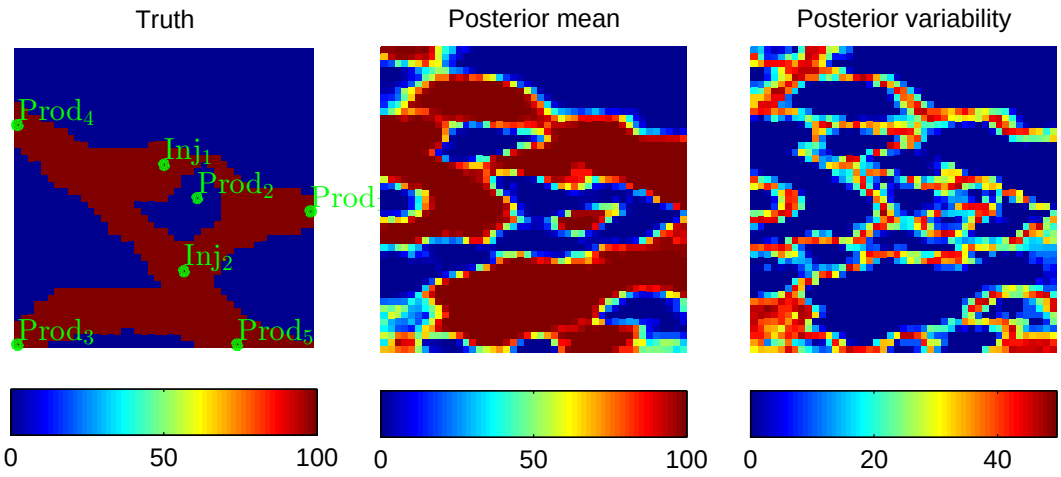
25-Subspace EnKF

The trend carries on over to 25 subspaces. As before, one of the subspaces was assigned more training samples (~ 1200) than its peers (~ 260). The posterior variability (figure 5.31a) is much greater than in figure 5.27a and, also, more uniformly distributed over the reservoir. Finally, the production plots (figure 5.31b) all capture the truth in the spread of the ensemble.

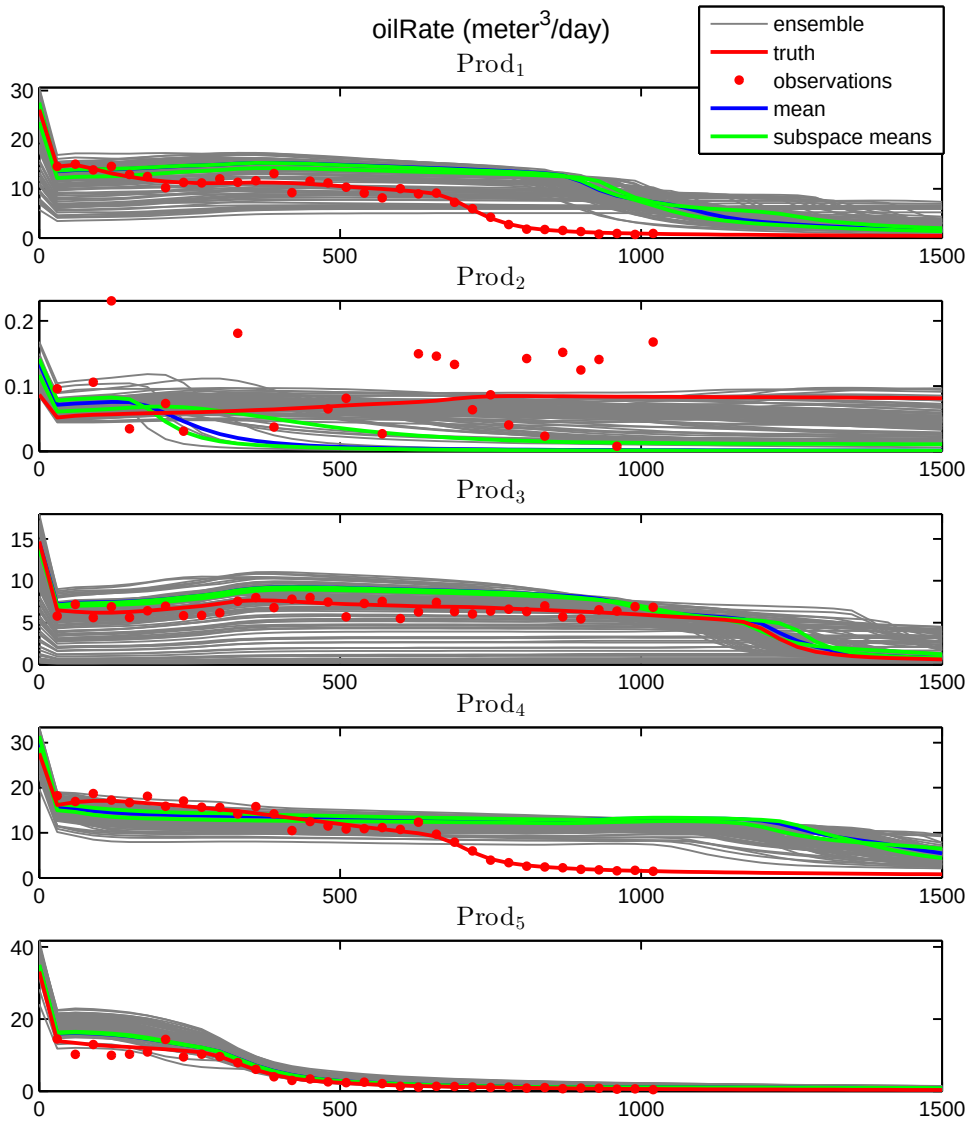
50-Subspace EnKF

The most notable output was obtained with 50 subspaces. We see (figure 5.32a) that the posterior mean is surprisingly similar to the truth and is accompanied by a good amount of variability. The production plots (figure 5.32b) exhibit, roughly, the same amount of spread as in the results without clustering (figure 5.28b), with the exception of $Prod_4$, for which the truth breaks outside of the ensemble range for the first part of the simulation (0-600 days).

This experiment shows that the Subspace EnKF generally performs better in conjunction with training set clustering. Some surprising results were obtained with 50 subspaces, where a large portion of the prior variability was maintained, while, at the same time, providing an informative posterior mean, with features similar to the true geological structure.

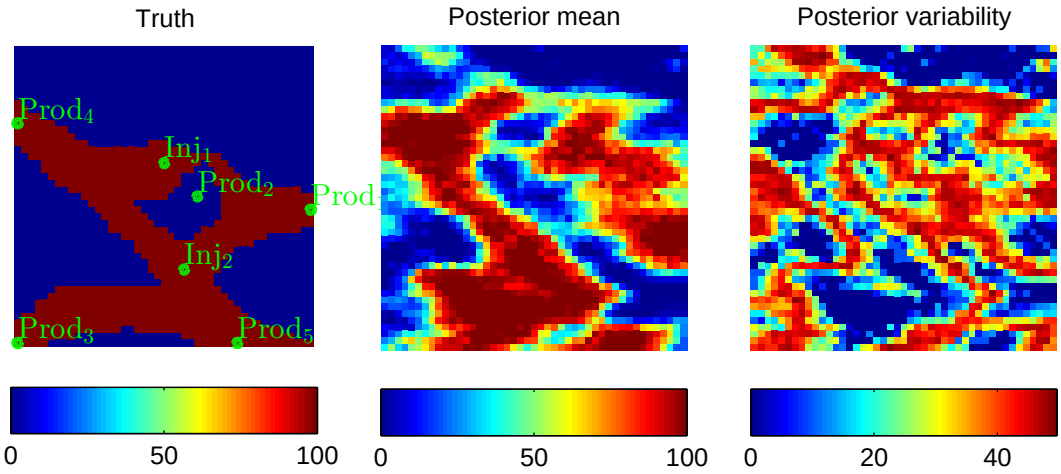


(a) True permeability field, ensemble mean and variability

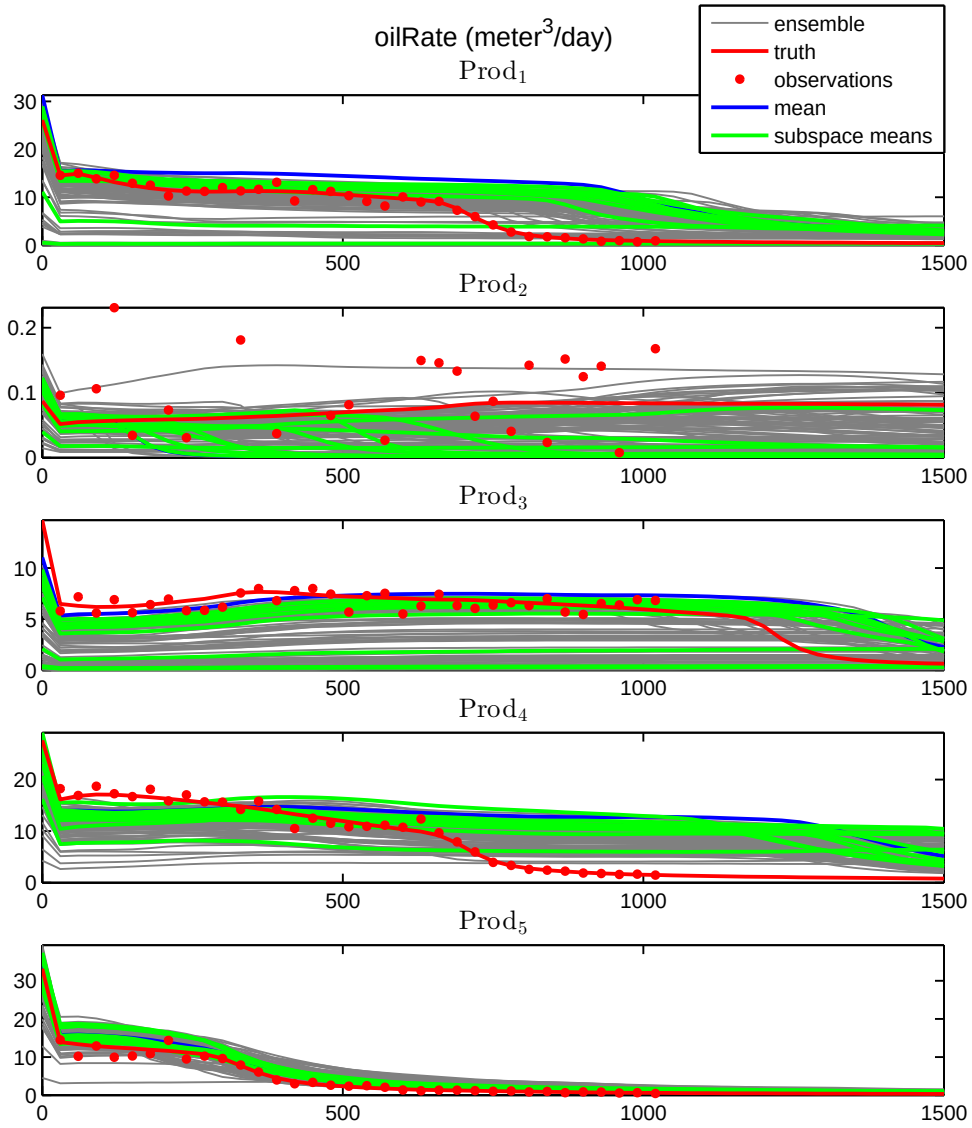


(b) Produced oil rates

Figure 5.29: Result of the Subspace EnKF with K-means clustering over 2 subspaces on the *Ribbon reservoir*

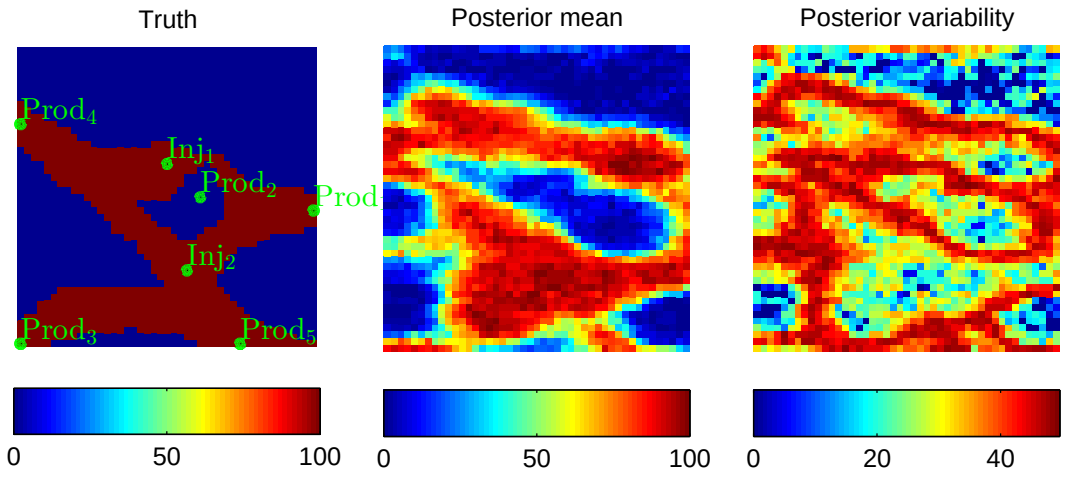


(a) True permeability field, ensemble mean and variability

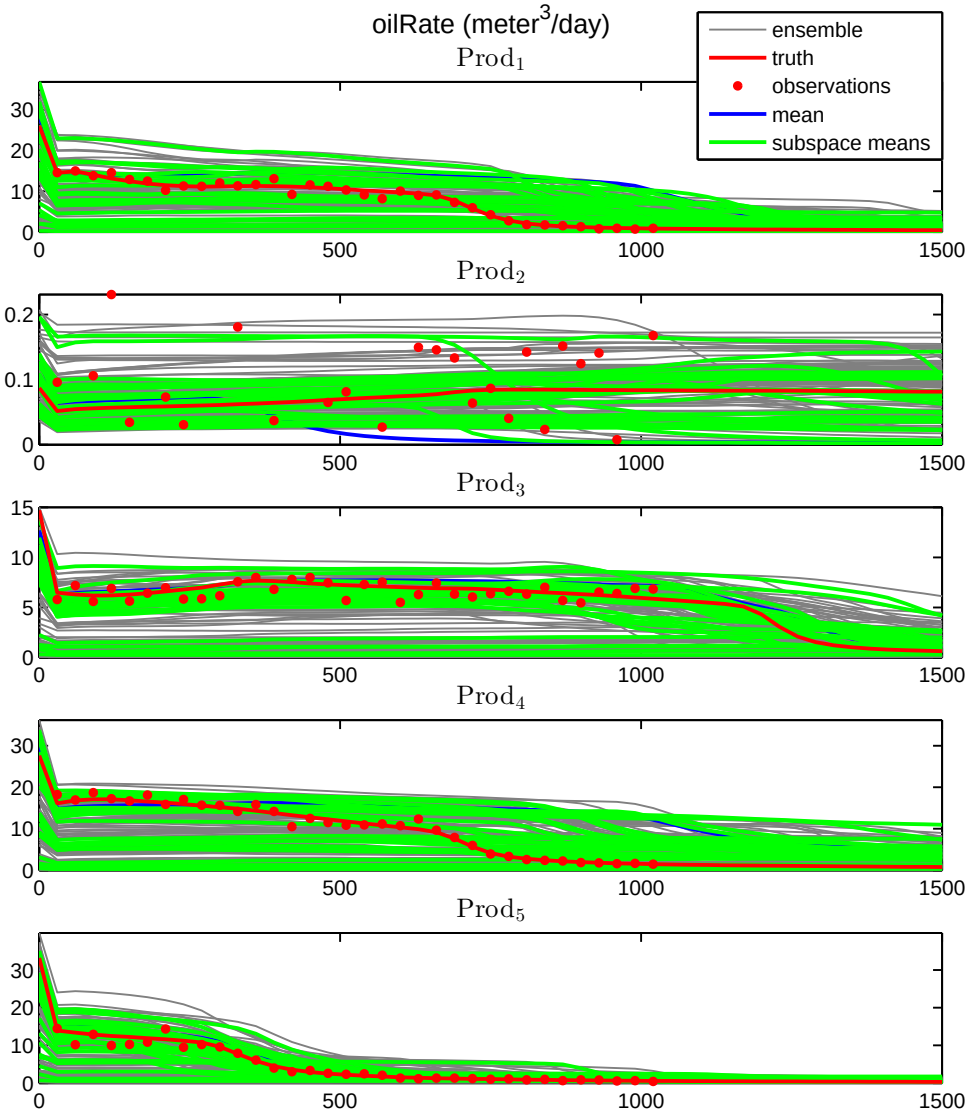


(b) Produced oil rates

Figure 5.30: Result of the Subspace EnKF with K-means clustering over 10 subspaces on the *Ribbon reservoir*

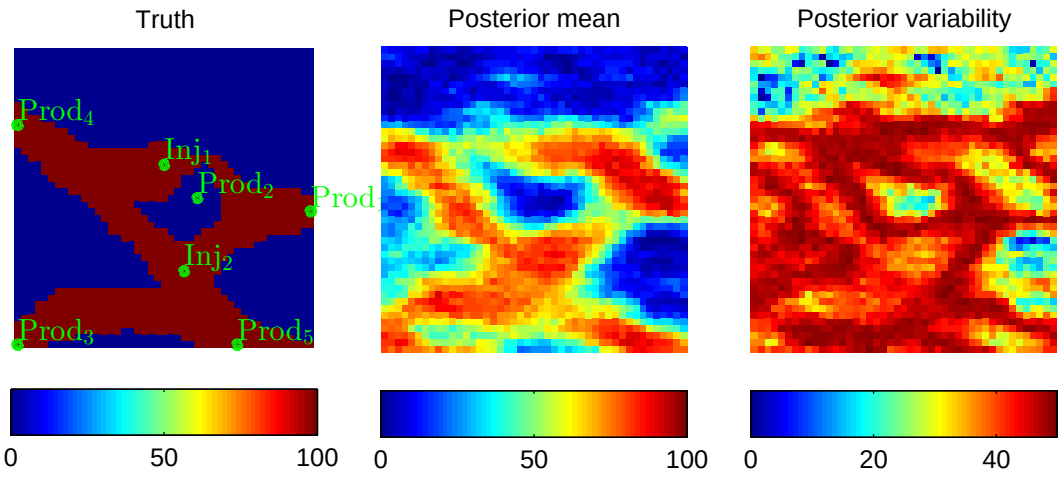


(a) True permeability field, ensemble mean and variability

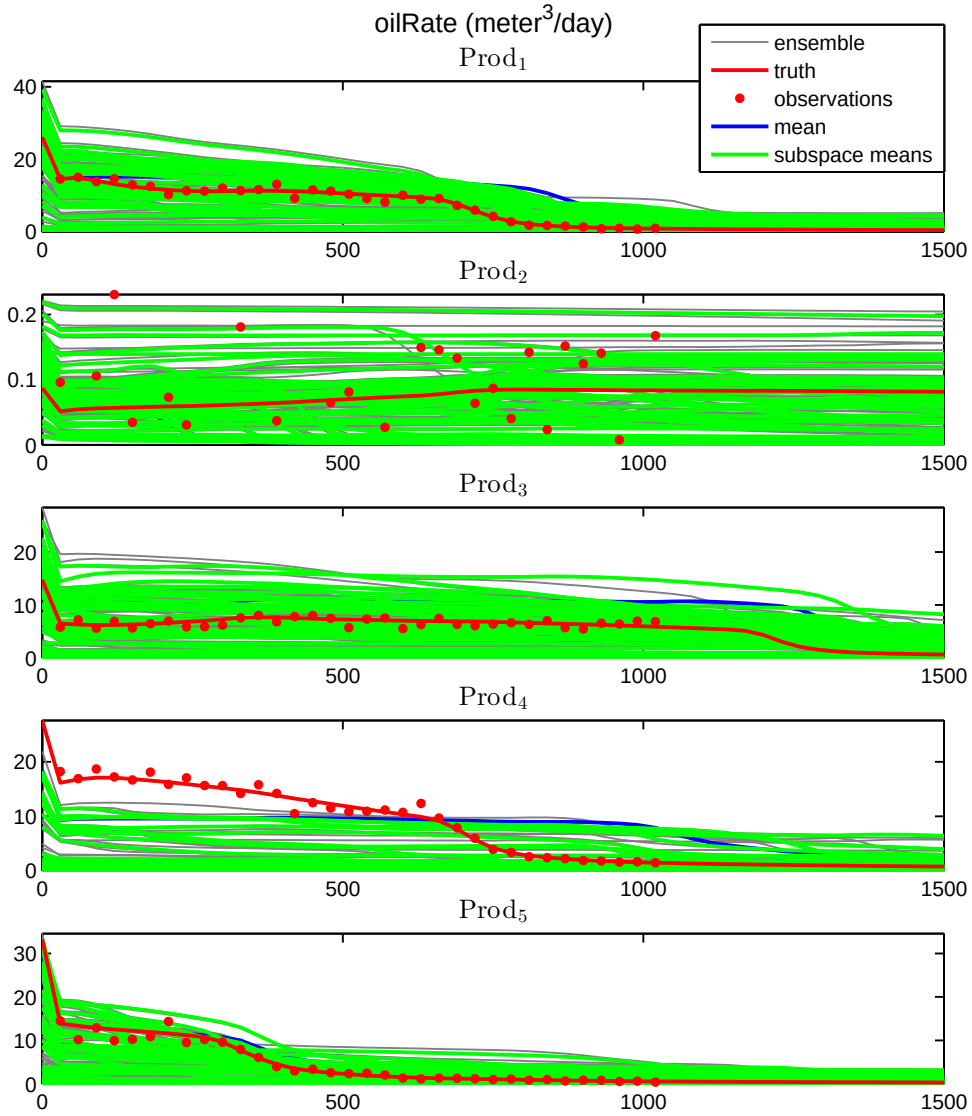


(b) Produced oil rates

Figure 5.31: Result of the Subspace EnKF with K-means clustering over 25 subspaces on the *Ribbon reservoir*



(a) True permeability field, ensemble mean and variability



(b) Produced oil rates

Figure 5.32: Result of the Subspace EnKF with K-means clustering over 50 subspaces on the *Ribbon reservoir*

Chapter 6

Conclusions

This chapter provides a summary of the developments in the present study, along with conclusions on the experimental results. We also make recommendations, in the hope that they will be useful when using the discussed algorithms in real-world application, and, finally, we end our presentation by sketching possible directions for further research.

Without further ado,

6.1. Summary of contributions

① We proposed a new approach for using data assimilation to estimate the geological structure of reservoirs composed of two facies. The permeability and porosity in each grid cell were replaced by a single value in $[0, 1]$, describing the likelihood of having one rock type, in detriment of the other (paragraphs 2.3.1 and 2.3.4). This also implies a reduction in state vector size, which is beneficial for the convergence of the scheme.

② We studied the problem of updating state variables with bounded feasible ranges. The first impulse was to use truncation, however the twin-experiment from paragraph 2.3.4 revealed its detrimental impact on the ensemble covariance, which led to a significant reduction in posterior variability. As an alternative, we proposed the logistic transform, for which, the resulting mean had a much closer resemblance to the truth and was accompanied by a better propagation of uncertainty.

③ We developed an analytical solution to the polynomial kernel PCA preimage problem. The sensitivity studies from paragraph 5.1 show that the new method has a similar, and, in some cases, better performance than the fixed-point iterative scheme suggested by the literature. More specifically, it is not susceptible to local optima and the channel features are better preserved when varying the training set size or the dimension of the reservoir grid. These benefits also come at a dramatically lower expense in computational resources.

④ We paired the analytical kernel PCA parameterization with the Iterative Ensemble Smoother, proposed by Emerick and Reynolds (2012). The results from paragraph 4.1 show an increase in ensemble variability, in comparison to the performance of the KPCA-EnKF (Sarma and W. Chen 2009).

⑤ We also applied said parameterization to the Subspace EnKF (Sarma and W. Chen 2013) and, despite obtaining usable results only for kernel degree 1, the stress-test from paragraph 5.3.1 revealed that the new scheme has a good ability in tackling ensemble collapse. Moreover, the spread in the ensemble's production response seems to increase with the number of subspaces.

⑥ We proposed training set clustering as a general approach to adapt the subspace parameterizations to the prior geological information. The experiments in paragraph 5.3.2 show that this results in further spread in posterior production response, when compared to the case without clustering. In particular, we obtained very promising results with the 50-Subspace

EnKF (figure 5.32), in which, the posterior mean shows a similar channel structure as the true field, while at the same time, a large part of the initial variability is preserved.

⑦ We added new functionality to the MRST data assimilation module developed at TNO (Leeuwenburgh 2012), accommodating all the algorithms mentioned in this report and allowing for parallel execution, in order to benefit from multi-core hardware.

⑧ Finally, we used this software framework to perform a comparison between the classic EnKF, the KPCA-EnS and the Subspace EnKF on two channelized reservoir examples (see paragraph 5.2). The results show that, in terms of avoiding ensemble collapse, the Subspace EnKF has an edge over EnKF, while the KPCA-EnS is plagued by the danger of getting stuck in local optima.

6.2. Recommendations

We would like to formulate the following of recommendations, with the hope that they will be useful for future research endeavours or attempts to use the studied algorithms in real-world applications:

- For polynomial kernel PCA parameterizations with odd degree, it is generally preferable to use the proposed analytical preimage solution (paragraph 3.5.2) over approximate schemes, such as fixed-point iterations.
- Our experiments showed that the KPCA parameterization scales well with the increase in reservoir grid size (paragraph 5.1.3). Still, after a certain point, more training samples might become necessary, in order to obtain usable preimage results. This is particularly evident when KPCA is used in conjunction with the logistic transform (paragraph 5.1.4). It is, then, important to note that the computational expense of the training phase grows exponentially with the number of samples (paragraph 5.1.5).
- When assimilating observations with different ranges, we recommend incorporating the rescaling procedure described in (Emerick and Reynolds 2012) into the Kalman update equation (see paragraph 2.3.3).
- If the state variables have prescribed bounds, we suggest normalizing them to $[0, 1]$ and then applying the logistic transform, before entering the assimilation cycle, as described in paragraph 2.3.4. This ensures that the results will be in the feasible range, with little impact on the performance of the EnKF and its derivatives.
- When using the Subspace EnKF, it is beneficial to choose the number of subspaces as high as possible. However, one also needs to consider the amount of training samples available for each of the subspace parameterizations. Theoretically, it is recommended for this to be sufficiently high, such that the information loss during the image-preimage cycle is at an acceptable level. That being said, though, we did achieve good results even in degenerate cases (paragraph 5.3.1).
- The Subspace EnKF seems to benefit from training set clustering (paragraph 5.3.2). This provides a way to accommodate the parameterizations to the prior geological information and can be generally applied, without significant computational overhead.
- Even though the KPCA-EnS frequently results in ensemble collapse, it should not be discarded from the start, since its ability to handle high-order moments is not shared by the Subspace EnKF (due to its limitation to kernel degree 1). We recommend, if feasible, to use both methods, alongside the classic EnKF, and discuss the results with the expert.

6.3. Proposals for future research

The subject of geological parameterizations seems to be an area of active research with many open threads to follow. During the various stages of the present study, we identified limitations and drawbacks of the various data assimilation algorithms, or directions that we simply lacked the logistical means to follow. We list them below, with the hope that they might stir interest in future research endeavours.

- What is the effect of the polynomial KPCA parameterization, when used in conjunction with data assimilation algorithms, on cases with continuous state variables? Consult the work by C. Mariş (TU Delft) on this topic.
- Is it possible to extend the formulation of the likelihood state vector, presented in paragraph 2.3.1 and 2.3.4, to cases with more than 2 facies? See the related work by Sebacher et al. (2013).
- Do the assimilation results benefit from using polynomial chaos expansions to approximate the distribution of the feature space images induced by KPCA? For more information, see (Ma and Zabaras 2011), where the authors generalize the results obtained by (Sarma, Durlofsky et al. 2008).
- What is the difference in the effect of the Kalman update equation and the steepest descent with step 1 on the state vector, i.e. to what extent does conjecture 4.1 hold? See (Sarma and W. Chen 2013), where this was used to develop the Subspace EnKF, and (Sayed and Kailath 1994), for a possibly related result.
- Is there a way to use higher-order KPCA directly in the Subspace EnKF framework? The results form Sarma and W. Chen (2013) seem to suggest so, however, the experiments in the present report show surprisingly slow convergence. Can this be an advantage in situations with a large volume of observations (such as seismic data)?
- How do the data assimilation algorithms studied in this report perform (in terms of both performance and computational expense), when faced with realistic 3D reservoirs?

Bibliography

A. Aizerman, E. M. Braverman and L. I. Rozoner (1964). 'Theoretical foundations of the potential function method in pattern recognition learning'. *Automation and remote control* 25, pp. 821–837.

G. Burgers, P. J. van Leeuwen and G. Evensen (1998). 'Analysis Scheme in the Ensemble Kalman Filter'. *Monthly weather review* 126.6, pp. 1719–1724.

A. Cauchy (1847). 'Méthode générale pour la résolution des systemes d'équations simultanées'. *Comp. Rend. Sci. Paris* 25.1847, pp. 536–538.

Y. Chang, C. Hsieh, K. Chang, M. Ringgaard and C. Lin (2010). 'Training and testing low-degree polynomial data mappings via linear SVM'. *The Journal of Machine Learning Research* 99, pp. 1471–1490.

M. Chen (2012). *LiteKmeans, a fully vectorized kmeans algorithm*. URL: http://www.mathworks.nl/matlabcentral/fileexchange/24616-kmeans-clustering.

C. Cortes and V. Vapnik (1995). 'Support-vector networks'. *Machine learning* 20.3, pp. 273–297.

H. B. Curry (1944). 'The method of steepest descent for nonlinear minimization problems'. *Quart. Appl. Math* 2.3, pp. 250–261.

A. P. Dempster, N. M. Laird and D. B. Rubin (1977). 'Maximum likelihood from incomplete data via the EM algorithm'. *Journal of the Royal Statistical Society, Series B (Methodological)*, pp. 1–38.

D. Elizondo (2006). 'The linear separability problem: Some testing methods'. *IEEE Transactions on Neural Networks* 17.2, pp. 330–344.

A. A. Emerick and A. C. Reynolds (2012). 'Ensemble smoother with multiple data assimilation'. *Computers & Geosciences*.

G. Evensen (1994). 'Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics'. *Journal of Geophysical Research: Oceans* 99.C5, pp. 10143–10162.

G. Evensen (2003). 'The ensemble Kalman filter: Theoretical formulation and practical implementation'. *Ocean Dynamics* 53.4, pp. 343–367.

G. Evensen (2009). *Data Assimilation: the Ensemble Kalman Filter, 2nd Edition*. Springer.

I. K. Fodor (2002). *A survey of dimension reduction techniques*. Tech. rep. UCRL-ID-148494. Lawrence Livermore National Laboratory.

E. W. Forgy (1965). 'Cluster analysis of multivariate data: efficiency versus interpretability of classifications'. *Biometrics* 21, pp. 768–769.

E. R. Fried, C. L. Schultze, G. L. Perry, G. Basevi, T. Watanabe, W. Tims, J. Williamson, J. A. Yager and S. E.B (1975). *Higher oil prices and the world economy: the adjustment problem*. Tech. rep. Brookings Institution, 1775 Massachusetts Ave. NW, Washington, DC 20036.

G. H. Golub and C. F. van Loan (2012). *Matrix computations*. Vol. 3. JHU Press.

F. Halliday (2005). *The Middle East in international relations: power, politics and ideology*. Vol. 4. Cambridge University Press.

T. M. Hansen (2011). *SGeMS, Stanford Geostatistical Modeling Software*. URL: http://mgstat.sourceforge.net/.

P. Honeine and C. Richard (2011a). 'A Closed-form Solution for the Pre-image Problem in Kernel-based Machines'. *Journal of Signal Processing Systems* 65.3, pp. 289–299.

P. Honeine and C. Richard (2011b). 'Preimage problem in kernel-based machine learning'. *Signal Processing Magazine, IEEE* 28.2, pp. 77–88.

H. Hotelling (1933). 'Analysis of a complex of statistical variables into principal components'. *Journal of educational psychology* 24.6, p. 417.

J. D. Jansen, D. R. Brouwer, G. Naevdal and C. P. J. W. van Kruijsdijk (2005). 'Closed-loop reservoir management'. *First Break* 23.1.

R. A. Johnson and D. W. Wichern (2002). *Applied multivariate statistical analysis*. Vol. 5. Prentice Hall.

I. Jolliffe (2005). *Principal component analysis*. Wiley Online Library.

R. E. Kalman (1960). 'A new approach to linear filtering and prediction problems'. *Journal of Basic Engineering* 82.1, pp. 35–45.

E. Kreyszig (2007). *Introductory functional analysis with applications*. Wiley.

J. T. Kwok and I. W. Tsang (2004). 'The pre-image problem in kernel methods'. *IEEE Transactions on Neural Networks* 15.6, pp. 1517–1525.

O. Leeuwenburgh (2012). *Ensemble Kalman Filter code for the MATLAB reservoir simulator MRST*. URL: http://enkf.iris.no/Internet/enkf.nsf/wvDocId/A9BBDC9E51A75622C1257A6B002D786E/$file/Olwijn_Leeuwenburgh_2012.pdf.

S. Lloyd (1982). 'Least squares quantization in PCM'. *IEEE Transactions on Information Theory* 28.2, pp. 129–137.

X. Ma and N. Zabaras (2011). 'Kernel principal component analysis for stochastic input model generation'. *Journal of Computational Physics* 230.19, pp. 7311–7331.

J. Mandel (2006). *Efficient Implementation of the Ensemble Kalman Filter*. Tech. rep. 231. University of Colorado at Denver and Health Sciences Center – Center for Computational Mathematics.

O. L. Mangasarian (1965). 'Linear and nonlinear separation of patterns by linear programming'. *Operations research* 13.3, pp. 444–452.

I. Myrseth and H. Omre (2009). 'Reliable Uncertainty Assessment in History Matching Production and Time-lapse Seismic Data'. In: *71st EAGE Conference & Exhibition*.

D. S. Oliver and Y. Chen (2011). 'Recent progress on reservoir history matching: a review'. *Computational Geosciences* 15.1, pp. 185–221.

K. Pearson (1901). 'On lines and planes of closest fit to systems of points in space'. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 2.11, pp. 559–572.

P. Sarma and W. Chen (2009). 'Generalization of the Ensemble Kalman Filter using kernels for nongaussian Random Fields'. In: *SPE Reservoir Simulation Symposium*.

P. Sarma and W. Chen (2013). 'Preventing Ensemble Collapse and Honouring Multipoint Geostatistics with the Subspace EnKF/EnS and Kernel PCA Parametrization'. In: *SPE Reservoir Simulation Symposium*.

P. Sarma, L. J. Durlofsky and K. Aziz (2008). 'Kernel principal component analysis for efficient, differentiable parametrization of multipoint geostatistics'. *Mathematical Geosciences* 40.1, pp. 3–32.

A. H. Sayed and T. Kailath (1994). 'A state-space approach to adaptive RLS filtering'. *IEEE Signal Processing Magazine* 11.3, pp. 18–60.

B. Schölkopf, S. Mika, C. J. Burges, P. Knirsch, K.-R. Müller, G. Rätsch and A. J. Smola (1999). 'Input space versus feature space in kernel-based methods'. *IEEE Transactions on Neural Networks* 10.5, pp. 1000–1017.

B. Schölkopf, S. Mika, A. J. Smola, G. Rätsch and K. R. Müller (1998). 'Kernel PCA pattern reconstruction via approximate pre-images'. In: *The 8th International Conference on Artificial Neural Networks*.

B. Schölkopf and A. J. Smola (2002). *Learning with kernels: Support Vector Machines, regularization, optimization and beyond*. MIT Press.

B. Schölkopf and A. J. Smola (2003). 'A short introduction to learning with kernels'. In: *Advanced lectures on machine learning*. Springer, pp. 41–64.

B. Schölkopf, A. J. Smola and K. Müller (1998). 'Nonlinear component analysis as a kernel eigenvalue problem'. *Neural computation* 10.5, pp. 1299–1319.

SCRF (2011). *SGeMS, Stanford Geostatistical Modeling Software*. URL: http://sgems.sourceforge.net/.

B. Sebacher, R. G. Hanea and A. W. Heemink (2013). 'A probabilistic parametrization for geological uncertainty estimation using the Ensemble Kalman filter (EnKF)'. *Computational Geosciences*, pp. 1–20.

SINTEF (2013). *MRST - MATLAB Reservoir Simulation Toolbox*. URL: https://www.sintef.no/Projectweb/MRST/.

SINTEF et al. (2013). *MRST Modules webpage*. URL: http://www.sintef.no/Projectweb/MRST/Modules/.

S. Strebelle (2002). 'Conditional simulation of complex geological structures using multiple-point statistics'. *Mathematical Geology* 34.1, pp. 1–21.

P. Taylor (26th June 2013). 'Big data in the spotlight as never before'. *Financial Times*.

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

M. Verlaan and A. W. Heemink (2001). 'Nonlinearity in data assimilation applications: A practical method for analysis'. *Monthly Weather Review* 129.6, pp. 1578–1589.

Appendix A

Background information concerning datasets

A dataset is a collection of equal-length vectors, $\mathbf{x}_1, \ldots \mathbf{x}_n \in \mathbb{R}^m$, which satisfy a set of common properties. Statistically, they can be seen as samples from the joint distribution of m random variables. Finally, from a geometric point of view, they describe the Cartesian coordinates of a cloud of points in an m-dimensional space. But, regardless of their abstract interpretation, the components of these vectors are quantities with physical meaning and we are interested to identify and study their characteristics.

The following paragraphs give a brief revision of the basic mathematical concepts related to datasets. And, since formulas are sometimes more compact when expressed in matrix form, we define the $m \times n$ matrix X as

$$X[:, i] = \mathbf{x}_i \qquad \forall i = 1, \ldots, n \tag{A.0.1}$$

A.1. Statistical properties

Important aspects of a dataset's structure can be described through its first and second order moments, *mean* and *covariance*, defined below.

Definition A.1: Sample mean

(Johnson and Wichern 2002) Let $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^m$ be a dataset with n members. Then the *sample mean* is the vector

$$\overline{\mathbf{x}} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i = X\mathbf{1_n} \tag{A.1.1}$$

Before we move on, we need to investigate a closely related concept. *Mean-centering* is a transformation of a dataset, such that its mean coincides with the origin. The result is referred to as the *mean-centered* dataset and is obtained by subtracting the mean from each of the original datapoints,

$$\widetilde{\mathbf{x}}_i = \mathbf{x}_i - \overline{\mathbf{x}} \qquad \forall i = 1, \ldots, n \tag{A.1.2}$$

or, in matrix notation,

$$\widetilde{X} = X(I_n - 1_n) \tag{A.1.3}$$

where

$$\widetilde{X}[:, i] = \widetilde{\mathbf{x}}_i \qquad \forall i = 1, \ldots, n \tag{A.1.4}$$

Definition A.2: Sample covariance

(Johnson and Wichern 2002) Let $\widetilde{\mathbf{x}}_1, \ldots, \widetilde{\mathbf{x}}_n$ be a mean-centered dataset with n members. Then the *sample covariance* is the matrix

$$S = \frac{1}{n-1} \sum_{i=1}^{n} \widetilde{\mathbf{x}}_i \widetilde{\mathbf{x}}_i^{\,T} = \frac{1}{n-1} \widetilde{X} \widetilde{X}^T \tag{A.1.5}$$

The diagonal elements of S are called *variances* and are defined as

$$S[j,j] = {\sigma_j}^2 = \frac{1}{n-1} \sum_{i=1}^{n} (\mathbf{x}_i[j] - \overline{\mathbf{x}}[j])^2 \qquad \forall j = 1, \ldots, n \tag{A.1.6}$$

where σ_j is the squared root of the variance and is referred to as the corresponding *standard deviation*. The off-diagonal elements of S are simply denoted as *covariances* and they provide measure of the *linear dependence* between the corresponding pair of variables. As a special case, if two variables are *independent*, then their covariance is zero. The converse, however, only implies that the pair is *uncorrelated*, which is a weaker property than independence (see Johnson and Wichern 2002, for a complete explanation).

We can also give a geometrical interpretation: the mean is the center of the cloud of points and the covariance matrix captures its shape, i.e. the variances are proportional to the spread of the points along each axis, while the covariances describe the cloud's projections onto the mutually perpendicular planes around the origin (one plane for each pair of axes).

Pearson's product-moment *correlation* coefficient is, alongside the covariance, also a measure of the linear dependence between variables. The two are closely related through the following

Definition A.3: Sample correlation

(Johnson and Wichern 2002) Given a dataset, $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^m$, and the associated covariance matrix, S, the *sample correlation* is the matrix with elements given by

$$\rho_{i,j} = \frac{S[i,j]}{\sigma_i \cdot \sigma_j} \qquad \forall i,j = 1, \ldots, m \tag{A.1.7}$$

where σ_i and σ_j are the standard deviations.

According to the Cauchy-Schwarz inequality, the correlation coefficient takes values in $[-1, 1]$. Essentially, it is a *dimensionless normalized* version of the covariance.

Throughout the report and in the following chapters we refer to the quantities defined above, simply, as mean vector, correlation and covariance matrix. They are not to be confused, however, with the *population* mean, correlation and covariance, which are the corresponding properties of the distribution from which the dataset was originally sampled (Johnson and Wichern 2002) and, thus, unknown in our application.

We used the notion of statistical moments to describe the mean and covariance. However, there are also higher-order moments and, since they play a crucial role in the present study, we give the following

Definition A.4: Statistical moment

(Johnson and Wichern 2002) Given the m-component random vector $\mathbf{X}$, its mixed moment of order n is defined as

$$E\left(\prod_{i=1}^{m}\mathbf{X}[i]^{k_i}\right) \qquad \begin{array}{l} k \in \mathbb{N}^* \\ \forall k_1,\ldots,k_m \in \mathbb{N}^* \ s.t. \ \sum_{i=1}^{m} k_i = k \end{array} \tag{A.1.8}$$

Notice that the statistical moments are obtained by evaluating monomials of the variables that generate datasets. As such, they are a measure of the *group dependencies* between these variables.

A.2. Linear separability

Consider a dataset composed of m-component samples from two categories, $\mathbf{X}_1$ and $\mathbf{X}_2$. For $m = 2$, the dataset is called *linearly separable* if we can draw a line that has all the samples from one category (say from $\mathbf{X}_1$) on one side, and all others (i.e. from $\mathbf{X}_2$) on its opposite side. This is illustrated in Figure A.1.

In $m \geq 3$ dimensions, the line is replaced by a (hyper)plane, while for $m = 1$ we have a point. Formally,

Definition A.5: Linear separability

(Mangasarian 1965) Let $\mathbf{X}_1$ and $\mathbf{X}_2$ be two classes of points in an m-dimensional space. Then $\mathbf{X}_1$ and $\mathbf{X}_2$ are *linearly separable* if

$$\begin{array}{l} \exists \mathbf{y} \in \mathbb{R}^m, k \in \mathbb{R} \text{ such that} \\ \forall \mathbf{x} \in \mathbf{X}_1, \quad \mathbf{x}^T\mathbf{y} \geq k \text{ and} \\ \forall \mathbf{x} \in \mathbf{X}_2, \quad \mathbf{x}^T\mathbf{y} < k \end{array} \tag{A.2.1}$$

Linear separability is a valuable property in image processing, automatic classification and artificial intelligence applications (Elizondo 2006).

A.3. Dimensionality reduction

Data mining has become the focus of many companies and researchers in recent years (Taylor 2013). To name a few, wireless field sensors, market indices and online social networks are virtually inexhaustible continuous streams of data. Crunching these huge high-dimensional datasets into usable information may prove prohibitively expensive even on the most powerful contemporary computers.

One method to bring a dataset down to a manageable size is to remove some of the data points, retaining only a representative subset. Clustering algorithms (see, for example, Forgy 1965; and Lloyd 1982; or Dempster et al. 1977) help identify groups of similar samples, which can be effectively replaced by their mean. However, in cases when the datapoints are spread away from each other (high variance), there is a lower bound to the number of clusters that can be formed. Crossing this bound can lead to a significant loss in information about the structure of the dataset, with a potentially detrimental impact on analysis results.

The rich availability of data sources might introduce redundancy into the dataset; some variables might measure the same quantity or be strongly correlated. Identifying and eliminating these redundancies can dramatically reduce the volume of data with little to no impact on its informativeness. This procedure is called *dimensionality reduction* and can be formally defined as

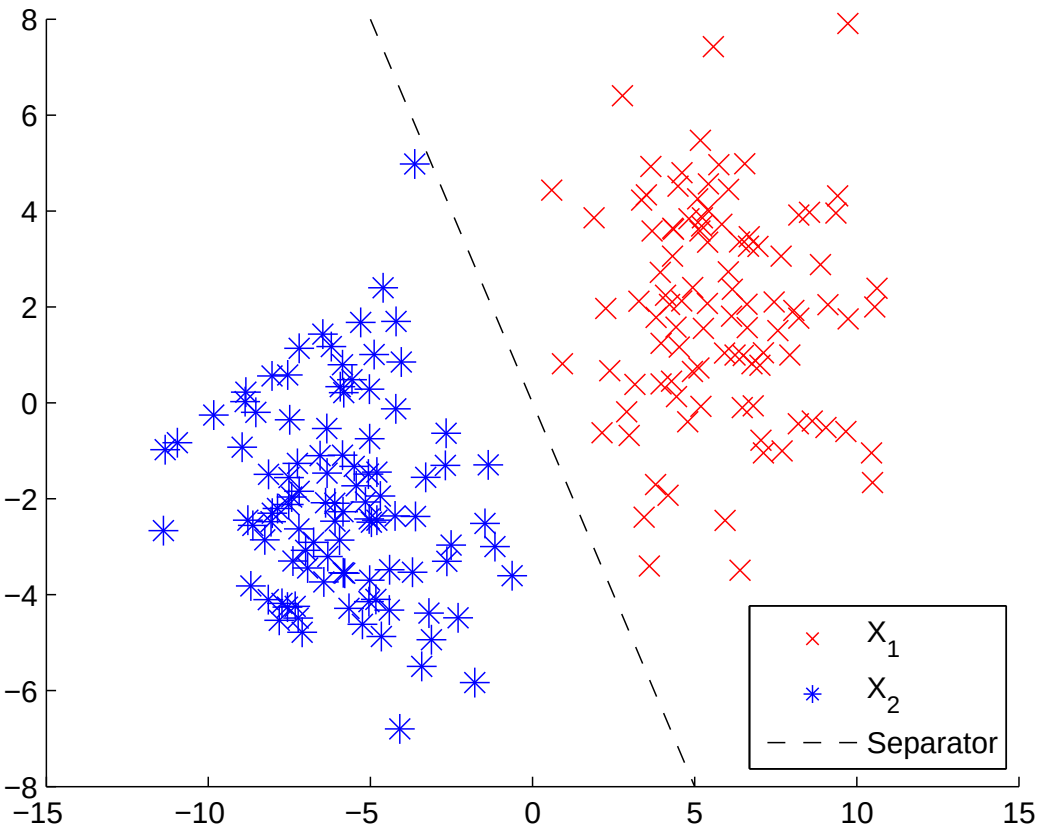


Figure A.1: Linearly separable dataset

Definition A.6: Dimensionality reduction

(Fodor 2002) Given the random vector $\mathbf{X} \in \mathbb{R}^m$, find a representation in a lower-dimensional (sub)space, $\mathbf{Y} = Pr(\mathbf{X})$, where $Pr : \mathbb{R}^m \to \mathbb{R}^l$ and $l \leq m$, such that its structure is best preserved according to a specified criterion c:

$$\min_{\mathbf{Y} \in \mathbb{R}^l} |c(\mathbf{X}, \mathbf{Y})| \tag{A.3.1}$$