TU Delft

Delft University of Technology

Faculty of Electrical Engineering, Mathematics and Computer Science

Delft Institute of Applied Mathematics

Risk and Environmental Modelling

Master of Science Thesis

Accelerated Iterative Adaptive Gaussian Mixture Smoother (IAGS) with Optimal Bandwidth h

by

Fei Cong

(student number: 4180232)

July 18, 2013

Members of the Committee

Responsible Professor:	Prof. Dr. Ir. A.W. Heemink
Daily Supervisor:	Dr. A.S. Stordal
Committee Members:	Prof. Dr. Ir. A.W. Heemink
	Dr. A.S. Stordal
	Dr. Ir. A.G. Chitu
	Dr. Ir. M. Verlaan

Acknowledgement

I would like to start repaying my appreciation by expressing the gratitude to Dr. D. Kurowicka for giving me the opportunity to participate in the Risk and Environmental Modeling group of D.I.A.M and awarding me a scholarship for my study.

I am grateful to Dr. A.S. Stordal who has guided me in this research and offered me valuable advice when I met difficulties.

Many thanks to Prof. A.W. Heemink who has turned on the green light for me although I had passed the deadline for submitting the graduation application form.

My sincere thanks also goes to Dr A.G. Chitu for giving me the chance to make presentation in TNO, where I got a lot of useful feedback and suggestions.

I must acknowledge Dr R.G. Hanea even thought he is far away in Norway. It is he who led me into the field of data assimilation and introduced me to this interesting thesis topic.

I would also like to express my thankfulness to my girlfriend Xi. I still remember my initial embarrassment when leaving my home to Netherlands. Her support and encouragement helps me go through that.

Finally, I offer my special thanks to my parents for their unconditional love and dedication in my life.

Abstract

As an integral part of the reservoir modeling process, history matching technique plays an important role in obtaining the reasonable estimations of the geological parameters. Despite the existence of many history matching methods, their application in the highly nonlinear models with the limit of computational cost still remains a challenge. The iterative adaptive Gaussian mixture filter (IAGM) is mainly designed for this problem, by a combination of the scaled linear update and the iterative importance sampling. Based on the idea of IAGM, there comes the iterative adaptive Gaussian mixture smoother (IAGS). It is already proved that, compared to other iterative smoother methods, the IAGS method is asymptotically optimal for nonlinear models.

However, an important aspect influencing the performance of IAGM and IAGS is the choice of bandwidth parameter h. Although using small bandwidth h helps to reduce the bias introduced in the iterations, that also leads to more iterations and more computational cost. Studies show that the optimal bandwidth h is highly related to the model nonlinearity, so I investigate in this report about how to generate the adaptive bandwidth h based on the measure of model nonlinearity.

By modifying the available way to measure model nonlinearity, I created a more general measure. Applying this measure, I developed a way to calculate the optimal bandwidth h adaptively. The test done with both the Toy Model and the reservoir model suggests that the accelerated IAGS with the adaptive choice of h behaves better than the original IAGS with fixed h. The former results in faster assimilation speed and better final result than the latter.

Key Words: IAGS, bandwidth parameter, model nonlinearity.

Contents

List of Figures

List of Tables

Introduction

Reservoir engineering is a branch of petroleum engineering that applies scientific principles to the drainage problems arising during the development and production of oil and gas reservoirs so as to obtain a high economic recovery. The working tools of the reservoir engineer are subsurface geology, applied mathematics, and the basic laws of physics and chemistry governing the behavior of liquid and vapor phases of crude oil, natural gas, and water in reservoir rock.

One goal of reservoir engineering is to obtain an accurate prediction of the reservoir performance under different operation conditions. To this end, we need both a well developed reservoir model and some history matching techniques, from which we can generate good approximation of the reservoir properties.

Reservoir Model

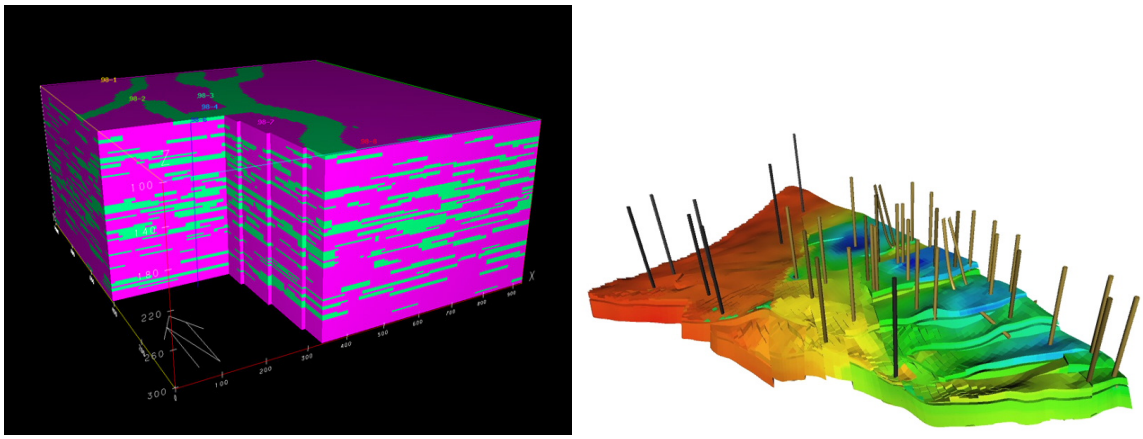
In the oil and gas industry, reservoir modeling involves the construction of a computer model of a petroleum reservoir, for the purposes of improving estimation of reserves and making decisions regarding the development of the field. [29].

A reservoir model represents the physical space of the reservoir by an array of discrete cells, delineated by a grid which may be regular or irregular. The array of cells is usually three dimensional, see Figure 1a, although 1D and 2D models, see Figure 1b, are sometimes used. Values for attributes such as porosity, permeability, pressure and water saturation are associated with each cell. The value of each attribute is implicitly deemed to apply uniformly throughout the volume of the reservoir represented by the cell.

The Reservoir models typically fall into two categories:

- Geological models are created by geologists and geophysicists and aim to provide a static description of the reservoir, prior to production.

- Reservoir simulation models are created by reservoir engineers and used to simulate the flow of fluids within the reservoir over its production lifetime by some numerical methods, which include finite difference method, finite element method and finite volume method.



(a) 3D Model **(b)** 2D Model

Figure 1: Two kinds of reservoir model.

Sometimes a single "shared earth model" is used for both purposes. More commonly, a geological model is constructed at a relatively high resolution. A coarser grid for the reservoir simulation model is constructed, with perhaps two orders of magnitude fewer cells. The prior effective values of attributes for the simulation model are then derived from the geological model by a process of "upscaling". Alternatively, if no geological model exists, the prior attribute values for a simulation model may be determined by a process of sampling geological maps.

Uncertainty in the true values of the reservoir properties is sometimes investigated by constructing several different realizations of the sets of attribute values. The behaviour of the resulting simulation models can then indicate the associated level of economic uncertainty.

Simple Simulator "SimSim"

By combining mass balance and Darcy's theory and ignoring capillary pressures, the dynamics of a two phase (water and oil) reservoir under iso-thermal conditions can be described by the following system of PDE's:[13].

$$-\nabla\cdot[\frac{\alpha\cdot\rho_w\cdot k^0_{rw}}{\mu_w}\vec{\mathbf{K}}(\nabla-\rho_w\cdot g\cdot\nabla d)]+\alpha\cdot\rho_w\cdot\phi[S_w\cdot(c_w+c_r)\frac{\partial p}{\partial t}+\frac{\partial S_w}{\partial t}]-\alpha\cdot\rho_w\cdot q_w=0$$

$$-\nabla\cdot[\frac{\alpha\cdot\rho_o\cdot k^0_{ro}}{\mu_o}\vec{\mathbf{K}}(\nabla-\rho_o\cdot g\cdot\nabla d)]+\alpha\cdot\rho_o\cdot\phi[(1-S_w)\cdot(c_o+c_r)\frac{\partial p}{\partial t}-\frac{\partial S_w}{\partial t}]-\alpha\cdot\rho_o\cdot q_o=0$$

where:

- $-\nabla\cdot$ is the divergence operator
- $-\nabla$ is the gradient operator
- $-\alpha$ is the geometry factor
- $-\rho_o, \rho_w$ are the fluids densities
- $-\mu_o, \mu_w$ are the fluids viscosities
- $-k^0_{ro}, k^0_{rw}$ are the relative permeabilities
- $-\vec{\mathbf{K}}$ is the permeability tensor
- $-g$ is the acceleration of gravity
- $-d$ is the depth
- $-c_o, c_w, c_r$ are the compressibilities
- $-\phi$ is the porosity
- $-p$ is the oil/water pressure
- $-S_w$ is the water saturation
- $-q_o, q_w$ are the source terms
- $-t$ is the time

The in-house simple forward simulator "SimSim" used in this report is developed by Prof. J.D. Jansen at TU Delft. It solves the system of previous equations in the reservoir with five-spot injection-production configuration, which is shown in Figure 2.[21]

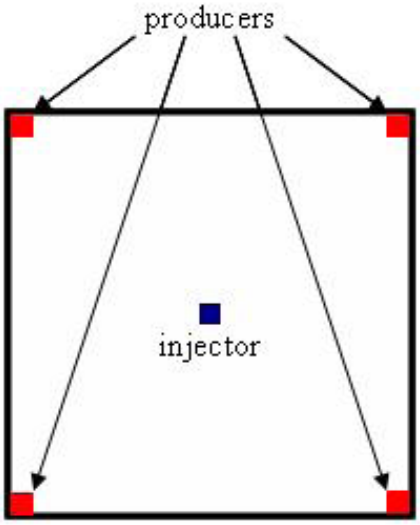


Figure 2: Five-spot injection-production model.

With the ensemble of permeability fields associated with this model, the simulator can be used as a black box to perform parameter and state estimation with different filtering techniques.

History Matching Methods

History matching is the act of adjusting a reservoir model until its simulated production response closely reproduces past behaviour. Once a reservoir model has been history matched, it can be used to simulate future reservoir behaviour with a higher degree of confidence, particularly if the adjustments are constrained by known geological properties of the reservoir. Expected future behaviour provides a basis for optimizing production scenarios.

The accuracy of the history matching process depends on the quality of the (prior) reservoir model and the quality and quantity of data, typically fluid production or pressures. Manually adjusting the reservoir parameters results in trial and error solutions, which is unsuitable for complex reservoir models, especially since quantification of uncertainty in the description of the reservoir is important. This requires a proper understanding of the main uncertainty drivers. One solution for this problem would be to apply methods that allow for multiple matched reservoir realisations and to deal with the uncertainty optimally.

Of the many history matching methods, the Ensemble Kalman filter (EnKF) [15], which supports multiple realisations, can deal rigorously with uncertainties and is easy to implement. The EnKF algorithm is a sequential method that updates the model with observations simultaneously, simulating reservoir behaviour for different sets of reservoir parameters. However, in order to get the good result from the EnKF, the state vectors are restricted to Gaussian distributions, which is usually not realistic in applications. Moreover, recent studies show that, in case of nonlinear model, the EnKF scheme is biased in the sense that the ensemble members will not be a sample from the true posterior distribution even in the limit of an infinite number of ensemble members. [5]

Another way to fulfil the history matching is by sequential importance sampling (SIS) or sequential importance resampling (SIR) filter, which is the most common particle filter method. Although there is no restriction of the model nonlinearity and the distributions of the state vectors for SIR and SIS, the main problem associated with them is the curse of the dimensionality in case of high dimension systems [2]. In order to avoid the filter collapse, the number of particles should grow exponentially with the system dimension [11, 30]. Therefore, within the affordable computational cost, the accuracy of SIS and SIR is limited.

The adaptive Gaussian mixture filter [5] is an improvement of the original Gaussian mixture filter by introducing a shrink parameter α. Besides, it can also be viewed as a combination of the EnKF and the particle filter. The AGM filter performs well in face of the nonlinear models and does not suffer from the degeneracy problem.

However, an important aspect that influences the behavior of AGM is the choice of the bandwidth parameter h. Previous research [26, 27] has shown that the optimal bandwidth h depends on the dimension of the state space, the nonlinearity of the model and the uncertainty in the prior distribution of state vector. For example, when the model is linear, it is better to choose big h to achieve good approximation in cost of little computation. In the contrast, if the model is quite nonlinear, then the general idea would be choosing small h and doing iterations with AGM in order to get reasonable result [6].

Based on the idea of IAGM, the iterative adaptive Gaussian smoother is introduced in [7]. Although the smoother method has the disadvantage of linear updates with large amount of data compared to the sequential techniques, it has the advantage of not stopping and restarting the numerical model caused by the "confirmation step", which is used to avoid the unexpected behavior of the geologic parameters and the model state vectors.

All history matching studies show that in the process of history matching there is no cookbook to solve a certain problem; expertise is needed to determine a suitable history matching strategy. In reality, we lack the knowledge of the true permeability distribution, so the results should be validated by looking at the prediction capacity of the history matched models and at the amount of geologically realistic information in the models. Therefore, the objective function introduced in next section is usually used to measure the result of the history matching process. The final goal of history matching is to obtain a model that matches the past data within measurement and model uncertainty, predicts the oil and water rates in the existing wells and in new wells with increased confidence, and honours the relevant geology.

In this report, I will mainly research on how to select the optimal bandwidth h for the IAGS. Based on an available method of measuring the model nonlinearity, I make some modification to make it more general and adaptive in the iterations. With this modified measure, the bandwidth h of IAGS can be updated at each iteration. The test results show that this strategy works well for reservoir simulations in the sense of decreasing the objective function value and the data mismatch, which are introduced in the following section.

Two Measures of the History Matching Performance

There are usually two ways to measure the performance of history matching process. The first one is called the Squared Normalized Data Mismatch, which mainly measures the distance between the assimilated models and the true state on the output points.

After the data assimilation process, the final parameter estimates, $\{\xi^i\}_{i=1}^N$, of each ensemble member is acquired. Reruns of the reservoir model with these estimated parameters produce simulated measurements $\mathcal{M}_t(\xi^i)$ at time t. Let y_t be the true observations at time t, $\mathbf{R}$ the covariance matrix of the observation error, N_d the number of measurements and N the number of ensemble members. Then the data mismatch measure D can be calculated by:

$$D = \sqrt{\frac{\sum_t \sum_{i=1}^N (y_t - \mathcal{M}_t(\xi^i))^T \mathbf{R}^{-1} (y_t - \mathcal{M}_t(\xi^i))}{N_d \times N}}.$$

If the ensemble memebers are not uniformly weighted, the computation of data mismatch D turns to be:

$$D = \sqrt{\frac{\sum_t \sum_{i=1}^{N} [(y_t - \mathcal{M}_t(\xi^i))^T \mathbf{R}^{-1} (y_t - \mathcal{M}_t(\xi^i)) \times w^i]}{N_d}}$$

where w^i denotes the weight of ensemble member i. When the model parameters are well estimated by the ensemble members, we will get small data mismatch D.

The other way to measure the assimilation result is called the normalized objective function. Rather than simply checking the mismatch of measurements, it also checks if the new ensemble parameter is consistent with the prior distribution, which contains some information about the real geology. The objective function $O(\xi^i)$ of each ensemble member reads:

$$O(\xi^i) = \frac{(\xi^i - \mu_p)^T \mathbb{C}_p^{-1} (\xi^i - \mu_p) + (y_t - \mathcal{M}_t(\xi^i))^T \mathbf{R}^{-1} (y_t - \mathcal{M}_t(\xi^i))}{N_d} \qquad i = 1, \cdots, N$$

where μ_p denotes the prior mean and $\mathbb{C}_p$ the prior covariance matrix. The weighted mean of the all objective function values $\{O(\xi^i)\}_{i=1}^N$ is given by:

$$\text{OB} = \sum_{i=1}^{N} O(\xi^i) w_i.$$

Ideally, via a good history matching technique, both the data mismatch D and the mean OB of the objective function values should decrease fast without introducing additional bias.

Chapter 1

Data Assimilation Methods

In the data assimilation process, we are interested in estimating the posterior distribution of the Markovian state vector $\mathbf{X}_k$ conditioned on some noisy measurements. It is always possible to make the state vector Markovian with a linear measurement operator $\mathbf{M}_k$ by extending the state vector to $\tilde{\mathbf{X}}_k$, which contains both the state vector $\mathbf{X}_k$ and the measurement data $\mathcal{M}_k(\mathbf{X}_k)$. The extended state vector reads:

$$\tilde{\mathbf{X}}_k = [\mathbf{X}_k \quad \mathcal{M}_k(\mathbf{X}_k)]. \tag{1.1}$$

The linear transformation matrix $\mathbf{M}_k$ can be constructed as:

$$\mathbf{M}_k = \begin{array}{c} \begin{array}{ccccccc} 1_{st} & \cdots & (d_s)_{th} & (d_s+1)_{th} & \cdots & (d_s+d_m)_{th} \end{array} \\ \begin{pmatrix} 0 & \cdots & 0 & 1 & & 0 \\ & \ddots & & & \ddots & \\ & & 0 & 0 & & 1 \end{pmatrix} \end{array} \tag{1.2}$$

where d_s is the length of original state vector $\mathbf{X}_k$ and d_m is the length of measurement vector $\mathbf{Y}_k$.

For simplicity, we define $\tilde{\mathbf{X}}_k$ by $\mathbf{X}_k$ in the following. The system can be described by the equations:

$$\mathbf{X}_0 \sim p(\mathbf{X}_0), \tag{1.3}$$

$$\mathbf{X}_k = f(\mathbf{X}_{k-1}) + \eta_k, \tag{1.4}$$

$$\mathbf{Y}_k = \mathbf{M}_k \mathbf{X}_k + \epsilon_k, \tag{1.5}$$

where $p(\mathbf{X}_0)$ is the initial distribution of the state vector, $f(\cdot)$ is the forward operator of the model state vector and $\mathbf{M}_k$ is the linear observation operator. The measurement error, ϵ_k, is assumed to be Gaussian white noise with covariance $\mathbf{R}$ and the model error, η_k, is Gaussian white noise with covariance $\mathbf{Q}$, which is usually set to be 0 in reservoir models. The optimal solution to the filtering problem is the posterior density $p(\mathbf{X}_k|\mathbf{Y}_{1:k} = y_{1:k})$, where $y_{1:k}$ denotes all the observations up to and including time step k.

Given the posterior density at time step $k-1$, the prior density at time step k can be calculated as

$$p(\mathbf{X}_k|\mathbf{Y}_{1:k-1}=y_{1:k-1}) = \int p(\mathbf{X}_k|\mathbf{X}_{k-1})p(\mathbf{X}_{k-1}|\mathbf{Y}_{1:k-1}=y_{1:k-1})\mathrm{d}\mathbf{X}_{k-1}. \tag{1.6}$$

When a new observation y_k arrives, the posterior density is updated via Bayes' rule:

$$p(\mathbf{X}_k|\mathbf{Y}_{1:k}=y_{1:k}) = \frac{p(\mathbf{Y}_k|\mathbf{X}_k)p(\mathbf{X}_k|\mathbf{Y}_{1:k-1}=y_{1:k-1})}{p(\mathbf{Y}_k|\mathbf{Y}_{1:k-1}=y_{1:k-1})} \tag{1.7}$$

where $p(\mathbf{Y}_k|\mathbf{Y}_{1:k-1}=y_{1:k-1})$ is the normalizing constant given by:

$$p(\mathbf{Y}_k|\mathbf{Y}_{1:k-1}=y_{1:k-1}) = \int p(\mathbf{Y}_k|\mathbf{X}_k)p(\mathbf{X}_k|\mathbf{Y}_{1:k-1}=y_{1:k-1})\mathrm{d}\mathbf{X}_k \tag{1.8}$$

For general models, the optimal solution $p(\mathbf{X}_k|\mathbf{Y}_{1:k})$ cannot be obtained analytically. Therefore, only approximative solutions can be computed via simulation techniques, where the prior and posterior densities are approximated by random samples. Usually the ensemble members $\{\xi_k^i\}_{i=1}^N$ are used to denote a sample from the prior density at time step k, and $\{\hat{\xi}_k^i\}_{i=1}^N$ a sample from the posterior density.

1.1. Ensemble Kalman Filter (EnKF)

The EnKF method has been widely used in the data assimilation field since it was first introduced in [15]. It is an extension of the Kalman Filter for large scale and nonlinear systems. Contrary to the extended Kalman Filter [18], which was also designed for the nonlinear cases, the ensemble Kalman Filter has no requirement for the tangent linear property of the model and takes affordable computation and memory cost.

The EnKF is a sequential Monte Carlo method for solving the nonlinear filtering problem with additional Gaussian assumption. Its process [9] can be presented as below:

- Initialization:

$$\xi_0^i \sim \mathcal{N}(\mathbf{X}_0, \mathbf{P}_0) \tag{1.9}$$

For time step k

- Time update (forecast step):

$$\xi_k^i = f_k(\hat{\xi}_{k-1}^i) + \eta_k^i, \quad i = 1, \cdots, N, \tag{1.10}$$

$$\mathbf{X}_k = \frac{1}{N}\sum_{i=1}^{N} \xi_k^i, \tag{1.11}$$

$$\mathbf{L}_k = [\xi_k^1 - \mathbf{X}_k, \cdots, \xi_k^N - \mathbf{X}_k]^T, \tag{1.12}$$

$$\mathbf{P}_k = \frac{1}{N-1}\mathbf{L}_k\mathbf{L}_k^T. \tag{1.13}$$

- Measurement update (analysis step):

$$\mathbf{K}_k = \mathbf{P}_k \mathbf{M}_k^T (\mathbf{M}_k \mathbf{P}_k \mathbf{M}_k^T + \mathbf{R})^{-1}, \tag{1.14}$$

$$\hat{\xi}_k^i = \xi_k^i + \mathbf{K}_k (y_k - \mathbf{M}_k \xi_k^i + \epsilon_k^i), \quad i = 1, \cdots, N, \tag{1.15}$$

$$\hat{\mathbf{X}}_k = \frac{1}{N} \sum_{i=1}^{N} \hat{\xi}_k^i, \tag{1.16}$$

$$\hat{\mathbf{L}}_k = [\hat{\xi}_k^1 - \hat{\mathbf{X}}_k, \cdots, \hat{\xi}_k^N - \hat{\mathbf{X}}_k]^T, \tag{1.17}$$

$$\hat{\mathbf{P}}_k = \frac{1}{N-1} \hat{\mathbf{L}}_k \hat{\mathbf{L}}_k^T. \tag{1.18}$$

In order to get the optimal result from the EnKF algorithm, the following assumptions of the problem have to be satisfied:

1. Model forward operator is linear, i.e. $f(X_k) = \mathbf{F}_k X_k$.
2. The measurement operator is linear.
3. The model and observation errors are uncorrelated and white Gaussian.
4. The initial state follows a multivariate Gaussian distribution with mean $\mathbf{X}_0$ and covariance matrix $\mathbf{P}_0$.

In most cases, the model describing the multi-phase fluid flow in the reservoir is not linear, which means that the first and second assumption are not satisfied, so that the EnKF is biased in the sense that the ensemble members will not be a sample from the true posterior distribution even in the limit of an infinite number of ensemble members. However, its robustness means that it does not suffer from the the dimensionality problem since every ensemble members carry the same weight. [5]

1.2. Sequential Importance Sampling and Particle Filters

1.2.1. Sequential Importance Sampling (SIS)

Sequential importance sampling (SIS) and particle filters [4, 2] provide an approximation of the optimal solution $p(\mathbf{X}_k | \mathbf{Y}_{1:k} = y_{1:k})$. Similar to the EnKF method, SIS and particle filters aim at building up sequentially a sample from the posterior density.[5]

Assume that, at given time step k, the sample $\{\hat{\xi}_{0:k}^i\}_{i=1}^N$ is from the joint posterior density function $p(\mathbf{X}_{0:k} | \mathbf{Y}_{1:k} = y_{1:k})$. In ordinary importance sampling, we construct

a sample from $p(\mathbf{X}_{0:k}|\mathbf{Y}_{1:k} = y_{1:k})$ by drawing N vectors $\{\xi^i_{0:k}\}^N_{i=1}$ vectors from a prescribed importance function $g(\mathbf{X}_{0:k}|\mathbf{Y}_{1:k} = y_{1:k})$ and attach an associated weight, w^i_t, which is given by:

$$w^i_k \propto \frac{p(\mathbf{X}_{0:k}|\mathbf{Y}_{1:k} = y_{1:k})}{g(\mathbf{X}_{0:k}|\mathbf{Y}_{1:k} = y_{1:k})}. \tag{1.19}$$

The EnKF can be viewed as an SIS algorithm restricting the weights w^i_k to be uniform. The method is valid for any importance function $g(\mathbf{X}_{0:k}|\mathbf{Y}_{1:k} = y_{1:k})$ as long as its support contains the support of $p(\mathbf{X}_{0:k}|\mathbf{Y}_{1:k} = y_{1:k})$, that is $p(\mathbf{X}_{0:k}|\mathbf{Y}_{1:k} = y_{1:k}) > 0$ implies $g(\mathbf{X}_{0:k}|\mathbf{Y}_{1:k} = y_{1:k}) > 0$. Otherwise, some part of the posterior distribution will never be taken into account in the sampling process and that will certainly cause biased estimates. On the other hand, all the weights, $\{w^i_k\}^N_{i=1}$ will be uniform, if the importance function $g(\mathbf{X}_{0:k}|\mathbf{Y}_{1:k} = y_{1:k})$ is exactly same as the posterior density function $p(\mathbf{X}_{0:k}|\mathbf{Y}_{1:k} = y_{1:k})$.

Although, in most cases, it is impossible to make all the weights be uniform, that is sampling from posterior density function $p(\mathbf{X}_{0:k}|\mathbf{Y}_{1:k} = y_{1:k})$, we want to find an importance function $g(\mathbf{X}_{0:k}|\mathbf{Y}_{1:k} = y_{1:k})$ as similar as the posterior density function $p(\mathbf{X}_{0:k}|\mathbf{Y}_{1:k} = y_{1:k})$, which will make the weights closer to uniform. In dynamic systems, due to the lack of information of the posterior density, we usually generate the initial importance function and update that sequentially with the coming information. That is we choose the importance function in the form

$$g(\mathbf{X}_{0:k}|\mathbf{Y}_{1:k} = y_{1:k}) = g(\mathbf{X}_k|\mathbf{X}_{k-1} = \xi_{k-1}, \mathbf{Y}_k = y_k)g(\mathbf{X}_{0:k-1}|\mathbf{Y}_{1:k-1} = y_{1:k-1}), \tag{1.20}$$

where ξ_{k-1} stands for the weighted mean of ensemble members $\{\xi^i_{k-1}\}^N_{i=1}$.

The sample $\{\xi^i_{0:k}\}^N_{i=1}$ is obtained by sampling $\{\xi^i_k\}^N_{i=1}$ from $g(\mathbf{X}_k|\mathbf{X}_{k-1} = \xi_{k-1}, \mathbf{Y}_k = y_k)$ and setting

$$\xi^i_{0:k} = [\xi^i_k \quad \xi^i_{0:k-1})].$$

The associated weights are updated sequentially to a proportionality constant via Bayes' theorem:[5]

$$\begin{aligned} w^i_k &= \frac{p(\xi^i_{0:k}|\mathbf{Y}_{1:k} = y_{1:k})}{g(\xi^i_{0:k}|\mathbf{Y}_{1:k} = y_{1:k})} \\ &\propto \frac{p(\mathbf{Y}_k|\mathbf{X}_k = \xi^i_k)p(\xi^i_k|\mathbf{X}_{k-1} = \xi^i_{k-1})p(\xi^i_{0:k-1}|\mathbf{Y}_{1:k-1} = y_{1:k-1})}{g(\xi^i_k|\mathbf{X}_{k-1} = \xi^i_{k-1}, \mathbf{Y}_k = y_k)g(\xi^i_{0:k-1}|\mathbf{Y}_{1:k-1} = y_{1:k-1})} \\ &\propto \frac{p(\mathbf{Y}_k|\mathbf{X}_k = \xi^i_k)p(\xi^i_k|\mathbf{X}_{k-1} = \xi^i_{k-1})}{g(\xi^i_k|\mathbf{X}_{k-1} = \xi^i_{k-1}, \mathbf{Y}_k = y_k)} w^i_{k-1} \end{aligned} \tag{1.21}$$

According to Eq 1.20 and 1.21, we see that, in contrast to the MCMC method [28], the sample can be obtained sequentially in time, which is very important that we do not have to restart the algorithm when new measurements arrive.

1.2.2. Sequential Importance Resampling Filter(SIR)

The SIS algorithm shown in Ch 1.2.1 may result in a filter degeneracy, when most of the weights become numerically zero after some time steps [23]. To avoid the filter degeneracy, an additional resampling step is introduced in the sequential importance sampling (SIR) filter, which is the most common particle filter. The only difference between SIS and SIR is the resampling step. The standard SIR filter use

$$g(\mathbf{X}_{0:k}|\mathbf{Y}_{1:k} = y_{1:k}) = p(\mathbf{X}_{0:k}) = p(\mathbf{X}_k|\mathbf{X}_{0:k-1} = \xi_{0:k-1})p(\mathbf{X}_{0:k-1})$$

as the importance function, and the corresponding weights are updated as

$$\bar{w}_k^i = w_{k-1}^i p(\mathbf{Y}_k|\mathbf{X}_k = \xi_k^i).$$

The normalized weights can be calculated as:

$$w_k^i = \frac{\bar{w}_k^i}{\sum_{j=1}^N \bar{w}_k^j}.$$

A measurement of the degree of degeneracy is the effective ensemble size, N_{eff},which is estimated by [3] :

$$\hat{N}_{\text{eff}} = \frac{1}{\sum_{i=1}^N (w_t^i)^2}. \tag{1.22}$$

When all the particles have equal weight, $\hat{N}_{\text{eff}}$ is equal to N, while if one of the weights is one and the others zero, $\hat{N}_{\text{eff}}$ is 1. If $\hat{N}_{\text{eff}}$ is smaller than some prescribed value N_c, resampling is then performed.

The resampling step is a selection process where the well fitted particles survive and the ill fitted particles die out. There are several ways to perform the resampling. The most used one is to generate N new particles under the multinomial distribution with the parameters $(N, w_k^1, \cdots, w_k^N)$. After the resampling step, all the weights are set to be N^{-1}.

Although resampling step is necessary in case of small $\hat{N}_{\text{eff}}$, it can be shown that it leads to underestimation of the posterior variance of the state variables [11]. On the other hand, since the resampling step is a stochastic procedure, it introduces more variance to the particles. Given that the variance of SIS estimator $\hat{\mu}_{\text{SIS}} = \sum_{i=1}^N \xi^i w^i$ is $\text{Var}(\sum_{i=1}^N \xi^i w^i)$, then the variance of the SIR estimator $\hat{\mu}_{\text{SIR}}$ with multinomial resampling is given by

$$\text{Var}(\hat{\mu}_{\text{SIR}}) = \text{Var}(\hat{\mu}_{\text{SIS}}) + \frac{\mathbf{E}\big[\sum_{i=1}^N (\xi^i)^2 w^i - (\sum_{i=1}^N \xi^i w^i)^2\big]}{N}.$$

However, the main problem of the SIR filter is the curse of dimensionality in high dimension systems. In order to avoid the filter collapse, the number of particles should grow exponentially with the system dimension [11, 30]. Therefore, in order to obtain informative particles for data assimilation, a large number of original particles is necessary. In practice this is not applicable for the high dimension system because of the limit of computational capacity.

1.3. Gaussian Mixture Filters

Gaussian mixture filters [10, 31, 24, 17] are based on the fact that any density function g defined on $\mathbb{R}^N$ (the state vector space) can be approximated in $L_1(\mathbb{R}^N)$ with the sample $\{\xi_i\}_{i=1}^N$ by the density of the form

$$\hat{g}(\mathbf{X}) = \sum_{i=1}^{N} w_i \Phi(\mathbf{X} - \xi^i, \mathbf{P}).$$

In general $\Phi(\cdot)$ can denote either Gaussian or non-Gaussian functions, but here we only consider the Gaussian case. So $\Phi(\mathbf{X} - \mu, \mathbf{P})$ denotes Gaussian density function with mean μ and covariance matrix $\mathbf{P}$. N is the number of particles (ensemble members) and $\{w_k^i\}_{i=1}^N$ are scalar weights with $\sum_{i=1}^N w_i = 1$ [12]. Each particle ξ_i represents the mean of a Gaussian kernel and the uncertainty associated with each particle ξ_i is given by the covariance matrix of the Gaussian kernel.

In filtering theory, with known one-step Markov transitions for the state vector and known likelihood function, the only unknown information needed to generate the posterior distribution $p(\mathbf{X}_k|\mathbf{Y}_{1:k} = y_{1:k})$ at time step k is the prior density function $p(\mathbf{X}_k|\mathbf{Y}_{1:k-1} = y_{1:k-1})$. The idea behind the Gaussian mixture filters is that, at each time step k, the prior density function is approximated by a Gaussian kernel density estimator

$$p(\mathbf{X}_k|\mathbf{Y}_{1:k-1}) \approx \sum_{i=1}^{N} w_{k-1}^i \Phi(\mathbf{X}_k - \xi_k^i, h^2\mathbf{P}_k), \tag{1.23}$$

where $\{\xi_k^i\}_{i=1}^N$ is the sample from the forecast step, i.e. $\{\xi_k^i\}_{i=1}^N = \{f_k(\hat{\xi}_{k-1}^i)\}$. $\{w_{k-1}^i\}_{i=1}^N$ is the corresponding weight for each ensemble particles, $\mathbf{P}_k$ the weighted covariance of ensemble members $\{\xi_k^i\}_{i=1}^N$, and h the bandwidth parameter.

1.3.1. Bayesian Update Step

When a new measurement arrives, the posterior density function is approximated by the Bayes' theorem,

$$p(\mathbf{X}_k|\mathbf{Y}_{1:k} = y_{1:k}) \propto p(\mathbf{X}_k|\mathbf{Y}_{1:k-1} = y_{1:k-1})\Phi(y_k - \mathbf{M}_k\mathbf{X}_k, \mathbf{R}_k). \tag{1.24}$$

By inserting Eq 1.23 into Eq 1.24, we can get:

$$p(\mathbf{X}_k|\mathbf{Y}_{1:k} = y_{1:k}) \propto \sum_{i=1}^{N} w_{k-1}^i \Phi(\mathbf{X}_k - \xi_k^i, \mathbf{P}_k)\Phi(y_k - \mathbf{M}_k\mathbf{X}_k, \mathbf{R}_k). \tag{1.25}$$

We can rewrite Eq 1.25 in a closed form:

$$p(\mathbf{X}_k|\mathbf{Y}_{1:k}) \approx \sum_{i=1}^{N} \tilde{w}_k^i \Phi(\mathbf{X} - \tilde{\xi}_k^i, \tilde{\mathbf{P}}_k), \tag{1.26}$$

where

$$\tilde{\xi}_k^i = \xi_k^i + \mathbf{K}_k(y_k - \mathbf{M}_k\xi_k^i), \quad , i = 1, \cdots, N,$$

$$\mathbf{K}_k = h^2\mathbf{P}_k\mathbf{M}_k^T\Sigma_k^{-1},$$

$$\Sigma_k = h^2\mathbf{M}_k\mathbf{P}_k\mathbf{M}_k^T + \mathbf{R}_k,$$

$$\tilde{\mathbf{P}}_k = h^2(\mathbf{I} - \mathbf{K}_k\mathbf{M}_k)\mathbf{P}_k,$$

$$\tilde{w}_k^i = \frac{\bar{w}_k^i}{\sum_{j=1}^N \bar{w}_k^j}, \quad , i = 1, \cdots, N,$$

$$\bar{w}_k^i = w_{k-1}^i \Phi(y_k - \mathbf{M}_k\xi_k^i, \Sigma_k).$$

1.3.2. Resampling Step

Similar to the SIS method shown in Ch 1.2.1, the Gaussian Mixture filter also suffers from the filter degeneracy problem. As shown in Ch 1.2.2, one possible solution to this problem is resampling. That is when the effective sample size N_{eff}, as defined in Eq 1.22, is smaller than some prescribed sample size N_c, then we perform the resampling step. The multinomial resampling of the Gaussian Mixture filter is to sample the new particles from the weighted kernel Gaussian density functions, i.e.:

$$\hat{\xi}_k^i \sim \sum_{i=1}^N w_k^i \Phi(\mathbf{X}_k - \tilde{\xi}_k^i, \tilde{\mathbf{P}}_k),$$

where $\{w_k^i\}_{i=1}^N$, $\{\tilde{\xi}_k^i\}_{i=1}^N$ and $\tilde{\mathbf{P}}_k$ are defined in Ch 1.3.1.

After resampling, the weights $\{w_k^i\}_{i=1}^N$ are reset to N^{-1} and the covariance matrix $\hat{\mathbf{P}}_k$ is based on the new particles $\{\hat{\xi}_k^i\}_{i=1}^N$. So the posterior distribution can be approximated by the new Gaussian Mixture filter as:

$$p(\mathbf{X}_k|\mathbf{Y}_{1:k}) \approx \frac{1}{N}\sum_{i=1}^N \Phi(\mathbf{X}_k - \hat{\xi}_k^i, \hat{\mathbf{P}}_k).$$

The resampling step may yield biased particles $\{\hat{\xi}_k^i\}_{i=1}^N$, since the covariance of the Gaussian mixture is larger than that before the resampling step. However, as mentioned in Ch 1.2.2, the resampling step generally leads to an underestimation of the posterior variance, so we will not do any bias correction here.

In the case when the effective sample size N_{eff} is bigger than N_c, the resampling step is then not necessary. And we can simply choose:

$$\hat{\xi}_k^i = \tilde{\xi}_k^i, \quad i = 1, \cdots, N$$

$$w_k^i = \tilde{w}_k^i, \quad i = 1, \cdots, N$$

$$\hat{\mathbf{P}}_k = \tilde{\mathbf{P}}_k.$$

for the next step.

1.3.3. Numerical Algorithm for Gaussian Mixture Filter

To sum up Ch 1.3.1 and 1.3.2, the numerical algorithm of Gaussian Mixture filter can be described as below:

- Initialization:

$$\text{Select} \quad h \quad \text{and} \quad N_c,$$

$$\xi_0^i \sim \mathcal{N}(\mathbf{X}_0, \mathbf{P}_0), \quad i = 1, \cdots, N,$$

$$\mathbb{X}_0 = [\xi_0^1, \cdots, \xi_0^N],$$

$$\hat{\xi}_0^i = \xi_0^i, \quad i = 1, \cdots, N,$$

$$\mathbf{T} = \begin{bmatrix} \mathbf{I}_{N-1} & \mathbf{0} \end{bmatrix}^T - \frac{\mathbf{1}_N \mathbf{1}_{N-1}^T}{N},$$

$$\hat{\mathbb{X}}_0 = \mathbb{X}_0,$$

$$\hat{\mathbf{L}}_0 = \hat{\mathbb{X}}_0 \mathbf{T},$$

$$w_0^i = N^{-1}, \quad i = 1, \cdots, N,$$

$$\mathbf{w}_0 = (w_0^i)_{i=1}^N,$$

$$\mathbf{W}_0 = diag(\mathbf{w}_0),$$

$$\mathbf{U}_0 = (\mathbf{T}^T \mathbf{W}_0^{-1} \mathbf{T})^{-1},$$

$$\hat{\mathbf{P}}_0 = h^2 \hat{\mathbf{L}}_0 \mathbf{U}_0 \hat{\mathbf{L}}_0^T.$$

For time step k

- Time update (forecast step):

$$\xi_k^i = f_k(\hat{\xi}_{k-1}^i),$$

$$\mathbb{X}_k = [\xi_k^1, \cdots, \xi_k^N],$$

$$\mathbf{L}_k = \mathbb{X}_k \mathbf{T},$$

$$\mathbf{P}_k = \mathbf{L}_k \mathbf{U}_{k-1} \mathbf{L}_k^T,$$

$$\Sigma_k = h^2 \mathbf{M}_k \mathbf{P}_k \mathbf{M}_k^T + \mathbf{R}_k,$$

$$\mathbf{K}_k = h^2 \mathbf{P}_k \mathbf{M}_k^T \Sigma_k^{-1},$$

- Measurement update (analysis step):

$$\tilde{\mathbb{X}}_k = \mathbb{X}_k + \mathbf{K}_k(y_k - \mathbf{M}_k\mathbb{X}_k),$$

$$\tilde{\mathbf{L}}_k = \tilde{\mathbb{X}}_k\mathbf{T},$$

$$\mathbf{B}_k = \mathbf{I}_{N-1} + [\mathbf{U}_{k-1} + (\mathbf{M}_k\mathbf{L}_k)^T\mathbf{R}_k^{-1}\mathbf{M}_k\mathbf{L}_k]^{-1}(\mathbf{M}_k\mathbf{L}_k)^T\mathbf{R}_k^{-1}[y_k - \mathbf{M}_k\mathbb{X}_k]\mathbf{T},$$

$$\mathbf{U}_k = (\mathbf{B}_k^T[\mathbf{U}_{k-1} + (\mathbf{M}_k\mathbf{L}_k)^T\mathbf{R}_k^{-1}\mathbf{M}_k\mathbf{L}_k]^{-1}\mathbf{B}_k)^{-1},$$

$$\tilde{\mathbf{P}}_k = \tilde{\mathbf{L}}_k\mathbf{U}_k\tilde{\mathbf{L}}_k^T,$$

$$\bar{w}_k^i = w_{k-1}^i\Phi(y_k - \mathbf{M}_k\xi_k^i, \Sigma_k), \quad i = 1, \cdots, N,$$

$$\tilde{w}_k^i = \frac{\bar{w}_k^i}{\sum_{j=1}^N \bar{w}_k^j}, \quad i = 1, \cdots, N,$$

$$\hat{\mathbf{N}}_{\text{eff}} = \frac{1}{\sum_{i=1}^N (w_k^i)^2},$$

- Resampling

  IF $\hat{\mathbf{N}}_{\text{eff}} < N_c$

$$\text{Index} = randsample((1, \cdots, N), (\tilde{w}_k^1, \cdots, \tilde{w}_k^N), replace = TRUE),$$

$$\hat{\xi}_k^i \sim \Phi(x - \xi_k^{\text{Index}(i)}, h^2\tilde{\mathbf{P}}_k), \quad i = 1, \cdots, N,$$

$$w_k^i = N^{-1}, \quad i = 1, \cdots, N,$$

$$\hat{\mathbb{X}}_k = \sum_{i=1}^N w_k^i\hat{\xi}_k^i,$$

$$\mathbf{U}_k = \mathbf{U}_0,$$

$$\hat{\mathbf{L}}_k = \hat{\mathbb{X}}_k\mathbf{T},$$

$$\hat{\mathbf{P}}_k = h^2\hat{\mathbf{L}}_k\mathbf{U}_k\hat{\mathbf{L}}_k^T.$$

  ELSE

$$\hat{\mathbb{X}}_k = \tilde{\mathbb{X}}_k,$$

$$\hat{\mathbf{P}}_k = h^2\tilde{\mathbf{P}}_k.$$

$$w_k^i = \tilde{w}_k^i, \quad i = 1, \cdots, N.$$

  END

1.4. Adaptive Gaussian Mixture (AGM) Filter

In order to estimate the posterior density precisely in nonlinear model, we prefer to choose small bandwidth parameter h. However, this may result in a degeneracy of

the filter due to the collapse of the weights. To solve this problem, a bias-variance tradeoff parameter α is introduced in [5]. The idea for that is to shrink the weights to uniform weights with additional parameter instead of increasing the bandwidth parameter h.

The update of the ensemble weights can be described as:

$$w_\alpha^i = \alpha w^i + (1-\alpha)N^{-1} \tag{1.27}$$

This process will reduce the variability of the weights, but introduces bias in the estimates. Therefore, when α decreases, the filter solution with updated weights moves further from the correct posterior distribution, and only takes some information from the likelihood function.

To make a balance between the variance and the bias, [5] gives out an objective function of α:

$$J(\alpha) = N\sum_{i=1}^{N}(w_\alpha^i)^2 + (\alpha-1)^2. \tag{1.28}$$

Minimizing this function gives the solution:

$$\alpha_{\text{opt}} = \frac{N}{\hat{N}_{\text{eff}}}, \tag{1.29}$$

With this α, the adaptively effective ensemble size $\hat{N}_{\text{eff}}^\alpha$ reads:

$$\hat{N}_{\text{eff}}^\alpha = \frac{1}{\sum_{i=1}^{N}(w_\alpha^i)^2} \tag{1.30}$$

$$= \frac{N^3}{\hat{N}_{\text{eff}}(N-\hat{N}_{\text{eff}})+N^2}. \tag{1.31}$$

where $\hat{N}_{\text{eff}}$ is the effective ensemble size given in Eq 1.22. With this choice of α, the new effective ensemble size is always above 80% of the ensemble size N, which will not cause a filter degeneracy problem any more.

To sum up, the difference between the numerical algorithm of the Gaussian Mixture filter and that of the AGM filter is that:

1. In AGM, the weights $\{\tilde{w}_k^i\}_{i=1}^N$ in the measurement update step are shrunk towards the uniform weights by parameter α.

2. Much fewer resampling steps are needed for AGM than for original Gaussian mixture.

1.5. Iterative Adaptive Gaussian Mixture (IAGM) Filter

As an iterative version of AGM, the Iterative Adaptive Gaussian Mixture filter is mainly developed for solving the assimilation problem with a limited sample size

in a large dimensional state space[6]. For the large dimension problem, the prior sample space is usually huge compared to the solution space, and therefore, almost all the samples from the prior distribution give some statistical mismatch to the data. In order to fix this problem, the sample size required is too large compared to the computational capacity we have. That is why a MCMC or a standard particle filter cannot be applied in this case.

The iterative adaptive Gaussian mixture filter is done by applying the AGM filter, introduced in Ch 1.4, in each iteration and bridging the gaps between two iterations by the resampling techniques. The process for the resampling between iterations can be described as:

$$p(\mathbf{X}) = \sum_{i=1}^{N} \Phi(x - \xi_{end}^i, h^2 \mathbf{P}_{end}) w_k^i, \quad i = 1, \cdots, N \tag{1.32}$$

$$\text{Sample} \quad \hat{\xi}_0^i \sim p(\mathbf{X}), \quad i = 1, \cdots, N \tag{1.33}$$

$$\hat{w}_0^i \propto \frac{p(\hat{\xi}_0^i)}{g(\hat{\xi}_0^i)}, \quad i = 1, \cdots, N. \tag{1.34}$$

The $\{\xi_{end}^i\}_{i=1}^N$ means the final ensemble members of the previous iteration with the weight $\{w_k^i\}_{i=1}^N$. The weighted covariance matrix of them is $\mathbf{P}_{end}$. The $\{\hat{\xi}_0^i\}_{i=1}^N$ denotes the initial ensemble members for the upcoming iteration, and the $\{\hat{w}_0^i\}_{i=1}^N$ shows the corresponding weight. $g(\cdot)$ is the prescribed importance function. As mentioned in [7], in the resampling step, usually we approximate the covariance matrix $\mathbf{P}_{end}$ of the updated state vectors by scaling the initial covariance matrix C_p, which means that we can get out of the ensemble subspace in each resampling.

Similar to the Sequential Importance Sampling method shown in Ch 1.2.1, the choice of the importance function $g(\cdot)$ will influence the initial weights $\{\hat{w}_0^i\}_{i=1}^N$. However, note that if the dimension of state space is large, computing these initial weights $\{\hat{w}_0^i\}_{i=1}^N$ will be time consuming. If the sample size N is not sufficiently large, the initial weights are usually set to be N^{-1} since the information contained in the weights is negligible in the AGM setting [6].

To sum up, if we modify the algorithm of original Gaussian Mixture shown in Ch 1.3.3 by introducing the shrunk weight w_α^i, which is calculated by Eq 1.27, and connect two separated AGM by resampling via Eq 1.32,1.33 and 1.34, we will get the numerical algorithm of IAGM. This algorithm is shown as a flowchart in Fig 1.1.

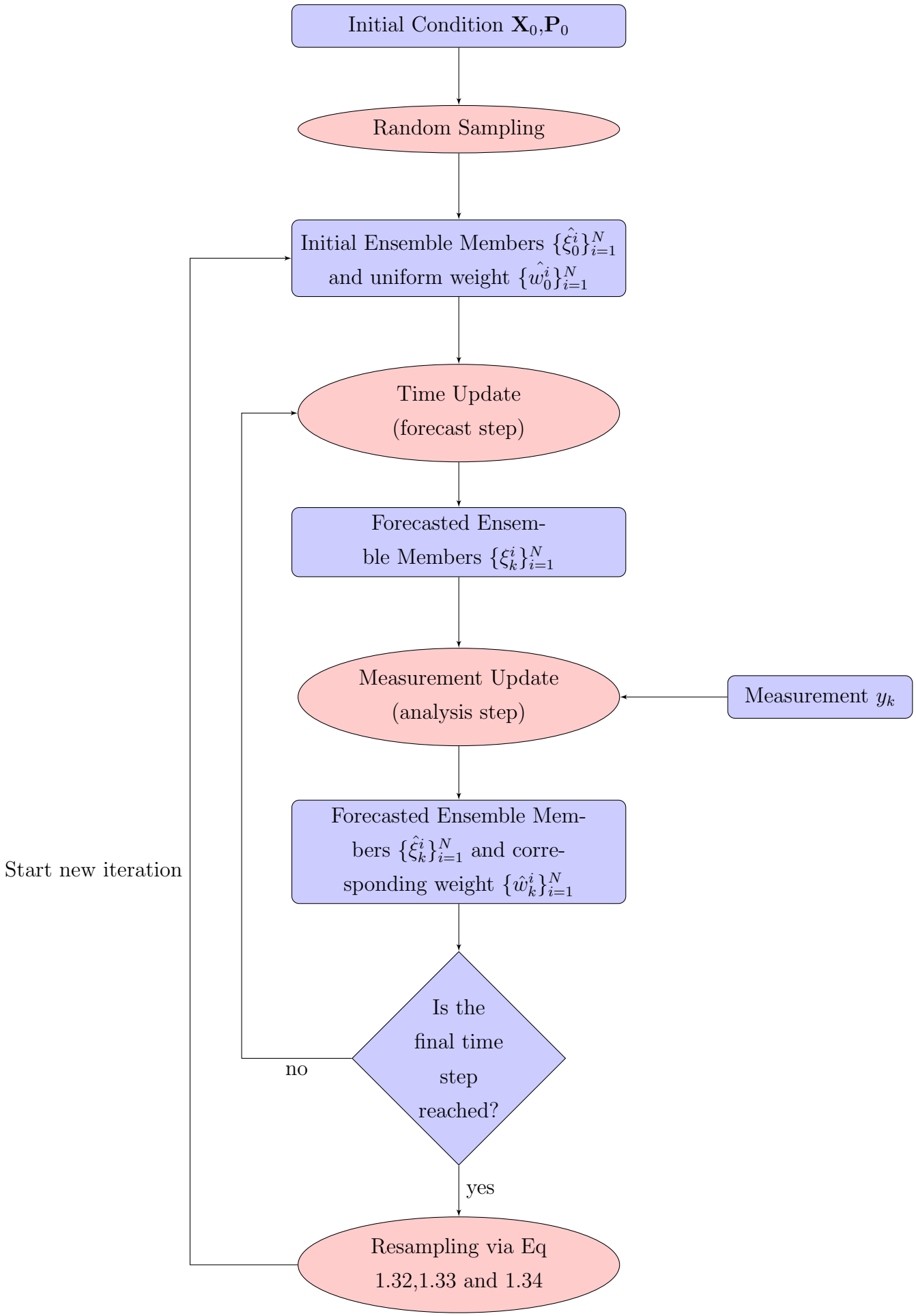


Figure 1.1: IAGM algorithm flowchart

1.6. Iterative Adaptive Gaussian Mixture Smoother (IAGS)

The methods introduced in the previous chapter are all sequential data assimilation methods, in which the data update is done sequentially in time. When applying sequential assimilation methods for the reservoir models, in order to avoid the unexpected behavior of the states, say fluid rates and pressures, it is always necessary to do "confirmation" update. That means we need to restart the reservoir simulation from the very beginning once we get some updated parameters. This will increase the computational cost dramatically. For example, for a reservoir problem with measurement arriving at time $[1, 2, \ldots, 10] \times t$, if we denote the computational load of AGM without "confirmation" as 10 units, the load of AGM with "confirmation" will be approximately 55 units. The difference between these two computational loads will increase in quadratic speed as the increasing of the measurement arrivals. Moreover, as for the iterative sequential method, the computation time would be scaled by the number of iterations, which means that it will be quite time consuming to get some reasonable results by IAGM.

The iterative adaptive Gaussian mixture smoother (IAGS) is defined as an approximation of adaptive importance sampling [16]. Although the smoother method has the disadvantage of linear updates with large amount of data compared to the sequential techniques, it has the advantage of not stopping and restarting the numerical model. For reservoir models, it is usually much faster to run throughout the whole test period than to stop and restart the simulator at each time when the measurement arrives. Moreover, it is shown in [7] that, compared to the other iterative smoother methods, the IAGS method is asymptotically optimal for nonlinear models and it does not have a Gaussian prior implicitly assumed in the methodology.

The derivation of IAGS is the same as IAGM for a deterministic system with one time step. Therefore the algorithm flowchart of IAGS shown in Fig 1.2 is quite similar to that of IAGM in Fig 1.1. It is shown in [7] that the posterior mean from IAGS converges to the correct posterior mean as the sample size and iterations increase as long as the bandwidth parameter h is selected as a decreasing function of the sample size. However, the optimal choice of bandwidth h for IAGS is still related to the model nonlinearity since it is a derivative of IAGM.

In the following chapters, I will research on how to choose the bandwidth h wisely to accelerate the IAGS based on some statistics of the model nonlinearity.

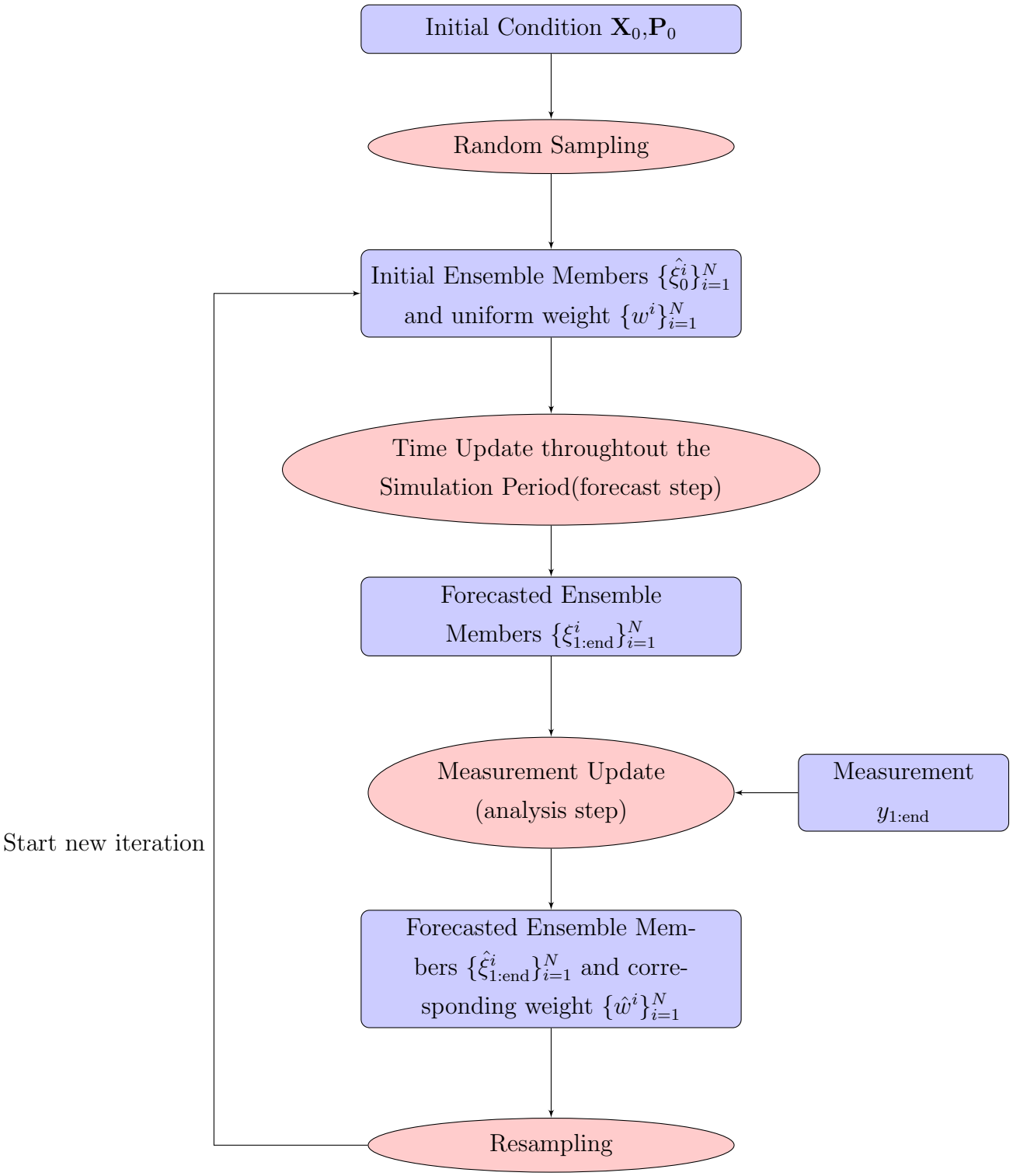


Figure 1.2: IAGS algorithm flowchart

Chapter 2

Measure of Model Nonlinearity

According to the introduction done in previous chapters about AGM and IAGS, the bandwidth parameter h could be a critical point that influences the linear update step. Previous research [26, 27] has shown that the optimal bandwidth h depends on the dimension of the state space, the nonlinearity of the problem and the uncertainty in the prior distribution. In this chapter, I will review the influence of the model nonlinearity on the optimal h through some tests. And then the method introduced in the internship report of mine will be shown briefly for measuring the model nonlinearity.

2.1. Influence of Model Nonlinearity on the Optimal Bandwidth h

In order to show that the optimal bandwidth h varies according to the model nonlinearity, I will use a toy model which can be described as below:

$$\mathbf{X} \sim \mathbb{N}(2, 2),$$
$$\mathbf{Y} = \mathbf{X}^k + \epsilon.$$

The ensemble members $\{\xi^i\}_{i=1}^N$ are randomly sampled from the normal distribution $\mathbb{N}(2, 2)$, where N denotes the sample size. The measurement error ϵ is white Gaussian noise with mean 0 and variance 5% of the true measurement which can be denoted by $(\mathbf{X}_{\text{true}})^k$. The transformation function $f(x) = x^k$ is used to introduce the nonlinearity into the model. When $k = 1$, the model is exactly linear, and when k is some bigger number, the model will be nonlinear. In the test shown below, I make k vary within $[1, 2, 3, 4, 5]$. To create the scenarios with different innovations, **Inn**, which can be defined by:

$$\mathbf{Inn} = ||f(\mathbf{X}_{\text{true}}) - \frac{1}{N}\sum_{i=1}^{N} f(\xi^i)||_1, \tag{2.1}$$

the true state $\mathbf{X}_{\text{true}}$ is chosen within 2 intervals $[2.1 : 0.3 : 3.9]$ and $[4.2 : 0.3 : 6]$. The former is close to the mean of the initial ensemble members and the latter further away, which also means that the latter has bigger innovation. The sample size N is chosen as 10000.

To quantify the update performance P_{update} of the different h, I used the updated data mismatch D_{new} divided by the original data mismatch D_{old} as the measurement, which means:

$$P_{\text{update}} = \frac{D_{\text{new}}}{D_{\text{old}}}.$$

Obviously, when P_{update} is close to 0, the update is quite good; when P_{update} goes larger, the performance of update turns to be worse; when $P_{\text{update}} > 1$, the update is wrong.

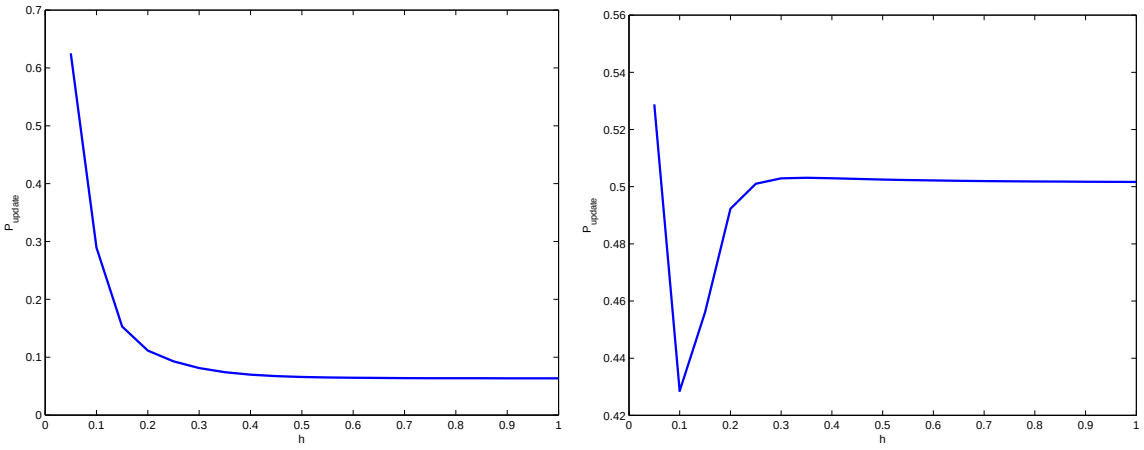
Consider the first scenario, where the true state $\mathbf{X}_{\text{true}}$ is chosen within $[2.1 : 0.3 : 3.9]$, the corresponding update performance P_{update} is plotted in Figure 2.1. As for the scenarios with big innovations, that is the true state $\mathbf{X}_{\text{true}}$ is chosen within $[4.2 : 0.3 : 6]$, the performance P_{update} is plotted in Figure 2.2.

It can be seen that: although the innovation may influence the performance of h in different scenarios, the optimal parameter h is still negatively related to the nonlinearity of the model. For both small and big innovations, the big bandwidth h always performs better than the small ones in the exactly linear case, see Figure 2.1a and 2.2a.

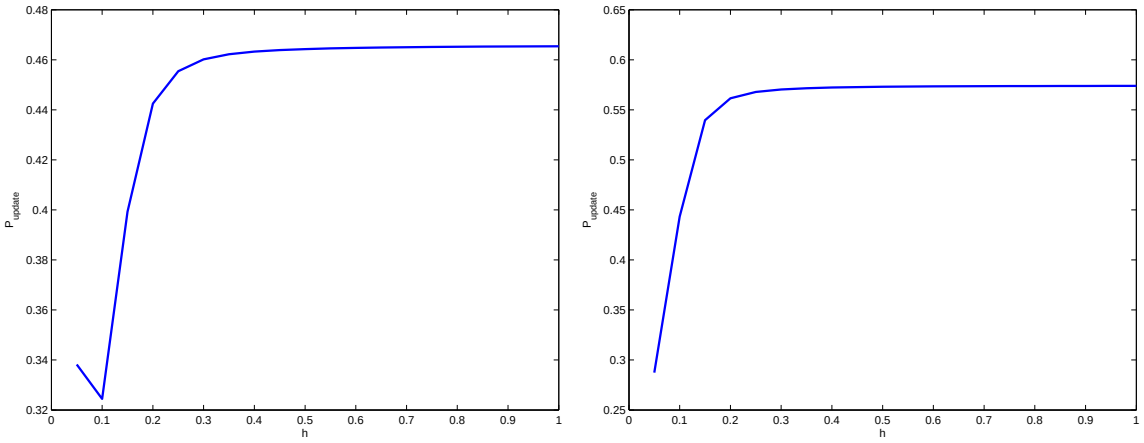
When the model nonlinearity goes larger, the optimal bandwidth h becomes smaller. For example, in the case of three order polynomial transformation $\mathbf{Y} = \mathbf{X}^3$, the optimal h is around 0.1 for small innovations and around 0.25 for big innovations.

When the model becomes more nonlinear, say the extreme case $\mathbf{Y} = \mathbf{X}^5$. The optimal h for small innovations becomes 0.05, and, for big innovations, it is around 0.15.

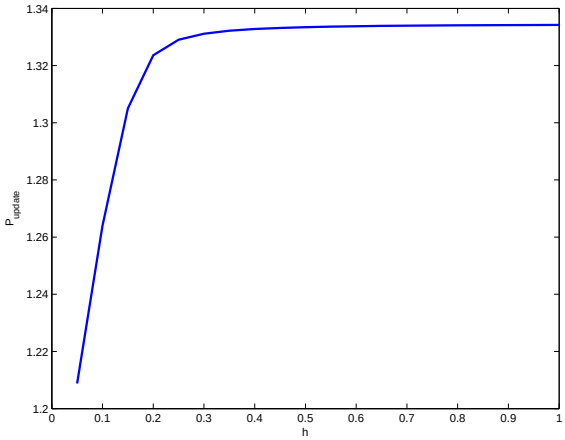
In all, the optimal bandwidth h is sensitive to the model nonlinearity, which means that if we choose good parameter h in the AGM (or IAGM, IAGS), the data mismatch D will decrease faster. Since it is mentioned in previous chapters that the optimal h is related to the model nonlinearity, if we can quantify the model nonlinearity and capture the quantitative relationship between that and the optimal h, we will get parameter h in an adaptive way.



(a) $\mathbf{Y} = \mathbf{X}$ **(b)** $\mathbf{Y} = \mathbf{X}^2$

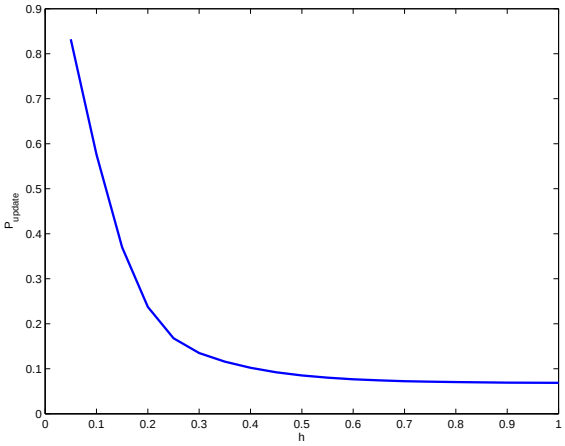


(c) $\mathbf{Y} = \mathbf{X}^3$ **(d)** $\mathbf{Y} = \mathbf{X}^4$

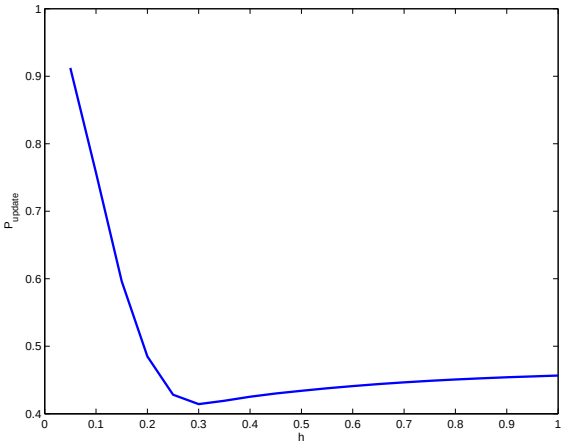


(e) $\mathbf{Y} = \mathbf{X}^5$

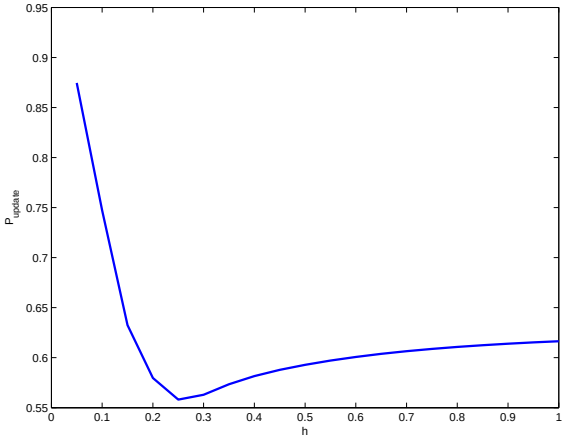
Figure 2.1: Update Performance for different h on Toy Problem I with small Innovation.



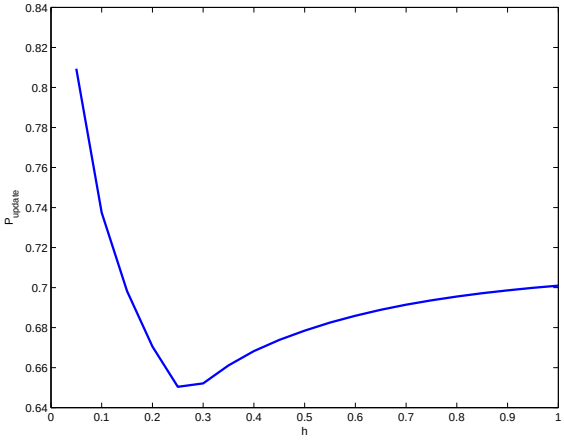
(a) $\mathbf{Y} = \mathbf{X}$



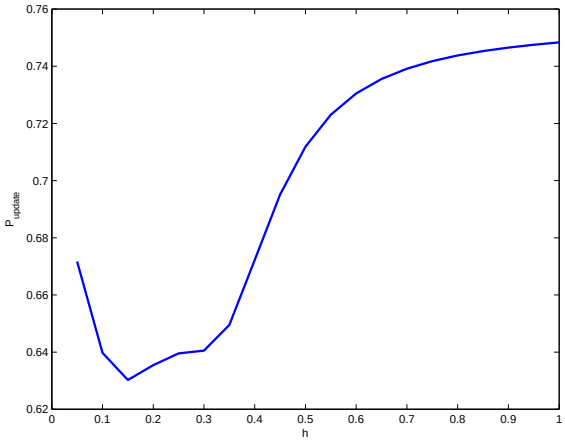
(b) $\mathbf{Y} = \mathbf{X}^2$



(c) $\mathbf{Y} = \mathbf{X}^3$



(d) $\mathbf{Y} = \mathbf{X}^4$



(e) $\mathbf{Y} = \mathbf{X}^5$

Figure 2.2: Update Performance for different h on Toy Problem I with big Innovation.

2.2. Measure of Model Nonlinearity

2.2.1. Original Method to Measure Model Nonlinearity

As introduced in [22, 25], the model nonlinearity can be approximated by the difference between the extended Kalman filter and the truncated second-order filter. For the linear model, this difference caused by the bias term in the truncated second-order filter is zero. When the degree of the model nonlinearity becomes larger, the difference will increase in the same way.

As for the ensemble case, the difference between the mean of the ensemble forecast and the forecast of the ensemble mean can be viewed as an estimate of the nonlinearity. Let X_i denote the state vector of ensemble member i, $\bar{X}$ denote the mean of the ensemble members, and $F(\cdot)$ denote the measurement operator. Then by using a Taylor expansion, the mean of the ensemble forecast could be described as:

$$\begin{aligned}\frac{1}{N}\sum_i F(X_i) &= F(\bar{X}) + \frac{\partial F(\bar{X})}{\partial X}((\frac{1}{N}\sum_i X_i) - \bar{X}) + \\ &\quad \frac{1}{2}\frac{\partial^2 F(\bar{X})}{\partial X^2}(\frac{1}{N}\sum_i (X_i - \bar{X})^T (X_i - \bar{X})) + \cdots . \end{aligned} \tag{2.2}$$

With some background information known, the ensemble mean can be viewed as a good estimation of the $\bar{X}$, which means:

$$F(\frac{1}{N}\sum_i X_i) \approx F(\bar{X}),$$

$$\frac{1}{N}\sum_i X_i \approx \bar{X},$$

$$\frac{1}{N}\sum_i (X_i - \bar{X})^T (X_i - \bar{X}) \approx \sigma(X).$$

Then the model nonlinearity, **NL**, can be denoted by:

$$\begin{aligned}\mathbf{NL} &= |\tfrac{1}{N}\textstyle\sum_i F(X_i) - F(\bar{X})| \\ &= |\tfrac{\partial F(\bar{X})}{\partial X}((\tfrac{1}{N}\textstyle\sum_i X_i) - \bar{X}) + \tfrac{1}{2}\tfrac{\partial^2 F(\bar{X})}{\partial X^2}(\tfrac{1}{N}\textstyle\sum_i (X_i - \bar{X})^T (X_i - \bar{X})) + \cdots| \\ &\approx |\tfrac{1}{2}\tfrac{\partial^2 F(\bar{X})}{\partial X^2}\sigma(X) + \cdots|. \end{aligned}$$

As shown in the equation, the first order term disappears in the difference between the "ensemble mean" and the "mean of ensembles". Therefore, only the nonlinear terms of Eq 2.2 are left in the measure **NL**.

2.2.2. Modified Method to Measure Model Nonlinearity

However, as mentioned in [14], the measure introduced in Ch 2.2.1 is not invariant to the linear transformations on the measurement data, which would limit the use

of that measure in general models. The possible solution is that: instead of simply calculating the measure $\mathbf{NL}$, we can calculate the ratio of $\mathbf{NL}_j$ and $\mathbf{NL}_{j-1}$, which separately denote the nonlinearity measure $\mathbf{NL}$ from iteration j and iteration $j-1$. So the new measure $\Delta\mathbf{NL}_j$ at the iteration j reads:

$$\Delta\mathbf{NL}_j = \frac{\mathbf{NL}_j}{\mathbf{NL}_{j-1}}.$$

The new measure $\Delta\mathbf{NL}_j$ is obviously built up on data from two iterations. However, it requires no rerun of the simulator when applied, so it is a quite cheap outcome from the simulation. Since there are usually more than one update inside one iteration of IAGM and every update may influence the nonlinearity $\mathbf{NL}$, we cannot simply apply this method into the IAGM. However, this method can be easily applied in one step IAGM, i.e. IAGS, where only one update is done inside the iteration.

According to [14], the relationship between the model nonlinearity and the measurement $\Delta\mathbf{NL}$ can be influenced by the parameter h used in the update and the innovation of the previous iteration, although it is not so sensitive to the latter.

When the parameter h is small, say $h < 0.1$, the correlation between the model nonlinearity and the measure $\Delta\mathbf{NL}$ is negative; when h becomes bigger, the correlation turns to be positive gradually. Therefore, when I use this method to measure the model nonlinearity in the following part, I consider two scenarios, $h \leqslant 0.1$ and $h > 0.1$. In the former case, I measure the model nonlinearity as proportion to $\Delta\mathbf{NL}^{-1}$, and in the latter, the model nonlinearity is measured proportional to $\Delta\mathbf{NL}$.

Chapter 3

Accelerated IAGS based on Two-step Data

The convergence speed of IAGS highly depends on the choice of bandwidth parameter h. For the linear model, choosing big h will make the assimilation done almost done in one step, see Figure 2.1a and Figure 2.2a. For the nonlinear model, choosing big h may result in a "over update" as shown in Figure 2.1e and 2.2e, and consequently, the assimilation result is worse than that done with small h.

For the usual IAGS method, we will do all the step with a certain h, say 0.01 or 0.05, since a small bandwidth h keeps the geological consistency while a large one might destroy that even if it helps to reduce the data mismatch D quickly. Moreover, choosing small h usually results in more iterations, which theoretically reduce the bias of the method.

However, in order to take advantage of the small bandwidth h, we should do more iterations than using some big bandwidth h, which means that the cost of good assimilation result is the computational load. To make a balance between the computational load and the accuracy of the method, I will introduce the accelerated IAGS.

The basic idea of accelerated IAGS is that if we introduce the nonlinearity measure shown in Ch 2.2 into IAGS and choose some bigger h based on this measure, then we can take advantage of the iteration strategy and also get the final result faster.

It has to be mentioned that although, according to [26, 27], the optimal h is related to the sample size, the dimension of state space, the model nonlinearity and the uncertainty of prior state, I will mainly consider the influence of last two parts in this report. There is also some research about selecting optimal bandwidth based on the theory of density estimation [20], but in that conclusion the optimal h is just a function of the prior uncertainty and the sample size. We can simply see it is not optimal in data assimilation.

3.1. Determine Optimal h based on Two-step Data

Since the new nonlinearity measure $\Delta\mathbf{NL}_{\mathrm{j}}$ is based on two-step data, $\mathbf{NL}_{\mathrm{j}}$ and $\mathbf{NL}_{\mathrm{j}-1}$, we also take this idea into choosing the optimal h. Instead of finding the relationship between the $\Delta\mathbf{NL}_{\mathrm{j}}$ and h_{opt}, I will try to figure out the relationship between $\Delta\mathbf{NL}_{\mathrm{j}}$ and $\frac{h_{\mathrm{opt}}}{h_{\mathrm{pre}}}$.

On the other hand, in order to take the update performance of the previous parameter h_{pre} into account, I will also introduce the innovation measurement $\Delta\mathbf{Inn}$ as:

$$\Delta\mathbf{Inn}_{\mathrm{j}} = \frac{\mathbf{Inn}_{\mathrm{j}}}{\mathbf{Inn}_{\mathrm{j}-1}},$$

where $\mathbf{Inn}$ is calculated in Eq 2.1. Similar to $\Delta\mathbf{NL}$, $\Delta\mathbf{Inn}$ is also a cheap outcome from the simulation.

When the previous update is good, the new innovation $\mathbf{Inn}_{\mathrm{j}}$ will be much smaller than $\mathbf{Inn}_{\mathrm{j}-1}$, and the innovation measurement $\Delta\mathbf{Inn}_{\mathrm{j}}$ will be quite small, then we have no reason to vary the parameter much away from h_{pre}. However, if the innovation measurement $\Delta\mathbf{Inn}$ is close to 1, which shows that the previous update is not good, then we should vary the bandwidth h.

To sum up, the ratio of the optimal h in the upcoming iteration and the h_{pre} used in the previous version should be some function of $\Delta\mathbf{Inn}$ and $\Delta\mathbf{NL}$, that is:

$$\frac{h_{\mathrm{opt}}}{h_{\mathrm{pre}}} = H(\Delta\mathbf{NL}, \Delta\mathbf{Inn}) \tag{3.1}$$

Since there is no special reason to choose some complicated functions as the function $H(\cdot)$, I will just consider linear functions. Meanwhile, since we know that for $h < 0.1$, the model nonlinearity is negatively related to $\Delta\mathbf{NL}$ and for $h > 0.1$, the correlation is more like positive, the function $H(\cdot)$ can be described as:

$$\frac{h_{\mathrm{opt}}}{h_{\mathrm{pre}}} = \begin{cases} a_1 \times (\Delta\mathbf{NL}) + b_1 \times (\Delta\mathbf{Inn}) & \text{if } h \leqslant 0.1 \\ a_2 \times (\Delta\mathbf{NL})^{-1} + b_2 \times (\Delta\mathbf{Inn}) & \text{if } h > 0.1 \end{cases}$$

If we do the previous iteration with small h and get a big measure $\Delta\mathbf{NL}$, then we know that the model is close to linear, which suggests that we can use some bigger h in the upcoming iteration. However, if the previous iteration is done with big h and it yields a big measure $\Delta\mathbf{NL}$, then the model is quite nonlinear, which suggests that some smaller h should be applied in the upcoming iteration.

The test result shown in the following chapter also suggests that doing the regression analysis separately gives better approximation than other choices.

3.2. Regression Analysis on Toy Model I

In order to get the reasonable analysis of the parameters a_1, a_2, b_1 and b_2 shown in Eq 3.1, we need to find a model with controllable nonlinearity and innovation, which is usually impossible in real geologic models. Therefore, I build up a Toy Model as shown below to finish this test.

3.2.1. Introduction of the Toy Model I

Compared with a realistic model, say the reservoir model, the processing of the toy model is less time consuming. Moreover, it is much easier to control the model nonlinearity and innovation. For example, when we generate the measurement, we could use almost linear transformation to get a linear model, Or we could use exponential transformation to create a nonlinear model.

To make the Toy Model similar to the realistic model, I develop it as follows:

The input value (state value) is generated from the normal distribution. The model output (measurement data) is generated from the input value via transformations with controllable nonlinearity. In all, the toy model can be described as below:

$$\begin{aligned} Y &= F(X) + \epsilon \\ X &\sim \mathbb{N}(\mu, \sigma^2) \\ \epsilon &\sim \mathbb{N}(\mu_{err}, \sigma_{err}^2) \end{aligned}$$

In the simulation, I choose $\mu = 2$, $\sigma = 1$ and $\mu_{err} = 0$. In order to make the variance of measurement error vary in different cases, I make $\sigma_{err} = Y_{\text{true}}/20$. However, it is also possible to fix the error as some constant. As for the transformation function $F(X)$, I used a polynomial transformation, whose nonlinearity can be clearly shown by its parameter of highest order term. The transformation reads:

$$polynomial \quad : \quad F(X) = a * X^3 + b * X^2 + c * X$$

In order to vary the nonlinearity, I fix $b = 0.5$ and $c = 1$, since the exactly linear model is hardly seen in real applications. Further I choose a within $[0.01 : 0.1 : 3.01]$, so that the model nonlinearity will be controlled by this parameter.

On the other hand, for varying the innovation, the true state value X_{true} is chosen within $[2.1 : 0.1 : 5]$. By using the transformation function $F(\cdot)$, the true measurement Y_{true} can be generated by the true state value X_{true}.

3.2.2. Test Design

Since the toy problem is quite cheap in computation, I choose the sample size as 100000, which is big enough to make the Sequential Importance Sampling (SIS)

result as a good approximation of the best update and to avoid the monte carlo effects in the results.

With the Toy Model introduced in Ch 3.2.1, we can build up a test with controllable model nonlinearity and innovation. In the test, we apply the IAGM (since there is only one time step in the test, it is also IAGS) with prescribed bandwidth parameter h_{pre} in the initial iteration, after which we could get the nonlinearity measure $\Delta\mathbf{NL}$ and the innovation measure $\Delta\mathbf{Inn}$.

For the following iteration, different bandwidth h will be used to generate different scenarios. Without considering weights, the bandwidth h, which gives out the minimal Hellinger Distance [8] between the update result of AGS and that of SIS, will be viewed as the optimal bandwidth h_{opt}.

Then we have $\frac{h_{\text{opt}}}{h_{\text{pre}}}$, $\Delta\mathbf{NL}$ and $\Delta\mathbf{Inn}$. Estimates of the parameters in Eq 3.1 can be obtained by Multivariate Linear Regression Analysis [19].

3.2.3. Test Results

The initial bandwidth h_{pre} is chosen within $[0.1, 0.3, 0.5, 1]$ and the test interval for h_{opt} is chosen within $[0.05 : 0.05 : 1]$. For the case where we start with $h_{\text{pre}} = 0.1$, the four kinds of possible multivariate linear regression analysis can be described as:

- For $h_{\text{pre}} = 0.1$:

$$\begin{aligned}
\frac{h_{\text{opt}}}{h_{\text{pre}}} &= 5.3579 * \Delta\mathbf{NL} + 1.5130 * \Delta\mathbf{Inn} + \epsilon, \qquad \epsilon \sim \mathbb{N}(0, 8.6109) \\
\frac{h_{\text{opt}}}{h_{\text{pre}}} &= 0.3937 * \Delta\mathbf{NL}^{-1} + 2.7624 * \Delta\mathbf{Inn} + \epsilon, \qquad \epsilon \sim \mathbb{N}(0, 9.1916) \\
\frac{h_{\text{opt}}}{h_{\text{pre}}} &= 7.5320 * \Delta\mathbf{NL} - (4.8065 \times 10^{-4}) * \Delta\mathbf{Inn}^{-1} + \epsilon, \qquad \epsilon \sim \mathbb{N}(0, 9.2414) \\
\frac{h_{\text{opt}}}{h_{\text{pre}}} &= 0.7201 * \Delta\mathbf{NL}^{-1} - 0.0072 * \Delta\mathbf{Inn}^{-1} + \epsilon, \qquad \epsilon \sim \mathbb{N}(0, 12.9135)
\end{aligned}$$

As discussed before, when h_{pre} is small, the fraction $\frac{h_{\text{opt}}}{h_{\text{pre}}}$ should be positively related to $\Delta\mathbf{NL}$ and $\Delta\mathbf{Inn}$ from the perspective of analysis. The first equation given above also suggests that with smallest residual variance, since the other regressions are less reliable with bigger variance.

When we consider the h_{pre} to be 0.3, the four kinds of possible multivariate linear regression analysis read:

- For $h_{\text{pre}} = 0.3$:

$$\begin{aligned}
\frac{h_{\text{opt}}}{h_{\text{pre}}} &= 2.8649 * \Delta\mathbf{NL} + 0.2739 * \Delta\mathbf{Inn} + \epsilon, \qquad \epsilon \sim \mathbb{N}(0, 1.7116) \\
\frac{h_{\text{opt}}}{h_{\text{pre}}} &= 0.2075 * \Delta\mathbf{NL}^{-1} + 0.7167 * \Delta\mathbf{Inn} + \epsilon, \qquad \epsilon \sim \mathbb{N}(0, 1.3587) \\
\frac{h_{\text{opt}}}{h_{\text{pre}}} &= 3.3948 * \Delta\mathbf{NL} + 0.0014 * \Delta\mathbf{Inn}^{-1} + \epsilon, \qquad \epsilon \sim \mathbb{N}(0, 1.7276) \\
\frac{h_{\text{opt}}}{h_{\text{pre}}} &= 0.2758 * \Delta\mathbf{NL}^{-1} - (3.1766 \times 10^{-4}) * \Delta\mathbf{Inn}^{-1} + \epsilon, \qquad \epsilon \sim \mathbb{N}(0, 1.7137)
\end{aligned}$$

The second equation also shows the best approximation result as that concluded in Ch 3.1. The same situation happens in the case $h_{\text{pre}} = 0.5$ and $h_{\text{pre}} = 1$ as well.

- For $h_{\text{pre}} = 0.5$:

$$\begin{aligned}
\frac{h_{\text{opt}}}{h_{\text{pre}}} &= 1.6404 * \Delta\mathbf{NL} + 0.2220 * \Delta\mathbf{Inn} + \epsilon, \qquad \epsilon \sim \mathbb{N}(0, 0.7229) \\
\frac{h_{\text{opt}}}{h_{\text{pre}}} &= 0.1346 * \Delta\mathbf{NL}^{-1} + 0.4272 * \Delta\mathbf{Inn} + \epsilon, \qquad \epsilon \sim \mathbb{N}(0, 0.4924) \\
\frac{h_{\text{opt}}}{h_{\text{pre}}} &= 2.1011 * \Delta\mathbf{NL} + (5.5139 \times 10^{-4}) * \Delta\mathbf{Inn}^{-1} + \epsilon, \qquad \epsilon \sim \mathbb{N}(0, 0.7339) \\
\frac{h_{\text{opt}}}{h_{\text{pre}}} &= 0.1743 * \Delta\mathbf{NL}^{-1} - (3.2477 \times 10^{-4}) * \Delta\mathbf{Inn}^{-1} + \epsilon, \qquad \epsilon \sim \mathbb{N}(0, 0.6235)
\end{aligned}$$

- For $h_{\text{pre}} = 1$:

$$\begin{aligned}
\frac{h_{\text{opt}}}{h_{\text{pre}}} &= 0.7412 * \Delta\mathbf{NL} + 0.1300 * \Delta\mathbf{Inn} + \epsilon, \qquad \epsilon \sim \mathbb{N}(0, 0.1896) \\
\frac{h_{\text{opt}}}{h_{\text{pre}}} &= 0.0683 * \Delta\mathbf{NL}^{-1} + 0.2072 * \Delta\mathbf{Inn} + \epsilon, \qquad \epsilon \sim \mathbb{N}(0, 0.1184) \\
\frac{h_{\text{opt}}}{h_{\text{pre}}} &= 1.0109 * \Delta\mathbf{NL} + (8.2141 \times 10^{-4}) * \Delta\mathbf{Inn}^{-1} + \epsilon, \qquad \epsilon \sim \mathbb{N}(0, 0.1922) \\
\frac{h_{\text{opt}}}{h_{\text{pre}}} &= 0.0869 * \Delta\mathbf{NL}^{-1} - (2.5001 \times 10^{-4}) * \Delta\mathbf{Inn}^{-1} + \epsilon, \qquad \epsilon \sim \mathbb{N}(0, 0.1502)
\end{aligned}$$

Generally, for small h_{pre}, the fraction $\frac{h_{\text{opt}}}{h_{\text{pre}}}$ turns to be bigger. If we have initial bandwidth $h_{\text{pre}} = 0.05$ and use the transformation function generated from $h_{\text{pre}} = 0.1$, then we will get the optimal bandwidth h_{opt} underestimated. If we apply the transformation equation generated from $h_{\text{pre}} = 0.1$ on the initial bandwidth $h_{\text{pre}} = 0.2$, then h_{opt} will be overestimated. In the IAGS algorithm, its iteration strategy suggests that we should do the update with small bandwidth h for couple of times instead of using the big bandwidth h in one run. Therefore, we accept the possible underestimation of the optimal bandwidth h_{opt} and make the transformation function for any h_{pre} can be described by:

- when $h_{\text{pre}} \in (0, 0.1]$,

$$h_{\text{opt}} = min((5.3579 * \Delta\mathbf{NL} + 1.5130 * \Delta\mathbf{Inn}) \times h_{\text{pre}}, 1)$$

- when $h_{\text{pre}} \in (0.1, 0.3]$,

$$h_{\text{opt}} = min((0.2075 * \Delta\mathbf{NL}^{-1} + 0.7167 * \Delta\mathbf{Inn}) \times h_{\text{pre}}, 1)$$

- when $h_{\text{pre}} \in (0.3, 0.5]$,

$$h_{\text{opt}} = min((0.1346 * \Delta\mathbf{NL}^{-1} + 0.4272 * \Delta\mathbf{Inn}) \times h_{\text{pre}}, 1)$$

- when $h_{\text{pre}} \in (0.5, 1]$,

$$h_{\text{opt}} = min((0.0683 * \Delta\mathbf{NL}^{-1} + 0.2072 * \Delta\mathbf{Inn}) \times h_{\text{pre}}, 1)$$

In Ch 4, I will apply the above strategy for generating the optimal h separately on a Toy Model and a Reservoir Model. The test result suggests that the IAGS done with adaptive choice of h behaves better than fixed h and gives out good assimilation result.

Chapter 4

Simulation Studies

4.1. Simulation wtih Toy Model II

4.1.1. Introduction of Toy Model II

First, the results from Ch 3.2 will be tested with a different Toy Model II. The basic idea of Toy Model II is almost similar to that used in Ch 3.2. The difference are in the transformation function $F(\cdot)$ and the choice of some parameters. In Ch 3.2, in order to build up couples of models with different nonlinearity, the transformation function is chosen as:

$$F(X) = aX^3 + 0.5X^2 + X$$

where parameter a varies from 0.1 to 3.1 to change the model nonlinearity in small step. The model parameters are chosen as:

$$\begin{aligned} \mu &= 2 \\ \sigma &= 1 \\ \mu_{err} &= 0 \\ \sigma_{err} &= Y_{true}/20 \\ X_{true} &\in [2.1 : 0.1 : 5] \end{aligned}$$

In the Toy Model II, instead of considering so many transformations as used in Toy Model I, I will simply vary the order of the transformation, which can be denoted by:

$$F(X) = X^k$$

where k varies in $[2, 3, 4]$.

The parameters of Toy Model II are :

$$\begin{aligned} \mu &= 2 \\ \sigma &= 2 \\ \mu_{err} &= 0 \\ \sigma_{err} &= Y_{true}/20 \\ X_{true} &\in [3, 4, 5] \end{aligned}$$

Since I will mainly focus on the performance of IAGS with updating h and IAGS without updating h, I choose the sample size as 10000 instead of 100000 to save some computation time.

4.1.2. Test Design

The test is about applying the two kinds of IAGS algorithm on the Toy Model II, which is developed in Ch 4.1.1. The first kind of IAGS is done with the bandwidth parameter h updated after every iteration, and the other one is done in the original way with fixed bandwidth h.

With the determined X_{true} and the model nonlinearity order k, the test is done in three scenarios with bandwidth $h = [0.01, 0.05, 0.1, 0.2, 0.3]$. In each scenario, the minimal data mismatch from all the iterations are chosen to indicate the performance of its corresponding IAGS. Besides, I also take the result from EnKS as the control group. For each IAGS algorithm, the total number of iterations is 15.

4.1.3. Test Results

The three scenarios for different true states X_{true} are shown in Table 4.1, 4.2 and 4.3. It can be seen that, in most cases, the IAGS with updating h performs better than that without updating. Even if the algorithm starts with quite small bandwidth $h = 0.01$, finally it gives out good result which is also better than that got from EnKS, or one step EnKF.

Table 4.1: $X_{true} = 3$. Minimal Data mismatch within 15 iterations.

h	update	k=2	k=3	k=4
0.01	yes	0.3621	0.2687	0.3761
0.01	no	0.6731	0.4170	1.9017
0.05	yes	0.3226	0.2838	0.3242
0.05	no	0.5216	0.3080	0.8008
0.1	yes	0.2739	0.2713	0.3927
0.1	no	0.3806	0.2830	0.5259
0.2	yes	0.3669	0.2623	0.3206
0.2	no	0.2993	0.2653	0.4668
0.3	yes	0.2753	0.2643	0.4978
0.3	no	0.3343	0.2655	0.2931
EnKS	NA	0.9909	0.9923	1.0062

Table 4.2: $X_{true} = 4$. Minimal Data mismatch within 15 iterations.

h	update	k=2	k=3	k=4
0.01	yes	0.3591	0.2749	0.2700
0.01	no	0.6395	0.4404	0.8461
0.05	yes	0.2620	0.2607	0.2824
0.05	no	0.4136	0.3108	0.3279
0.1	yes	0.2619	0.2554	0.2945
0.1	no	0.2856	0.2844	0.2923
0.2	yes	0.2693	0.2607	0.2792
0.2	no	0.3054	0.2745	0.2931
0.3	yes	0.2616	0.2637	0.2819
0.3	no	0.2508	0.2521	0.2629
EnKS	NA	0.9861	0.9970	1.0078

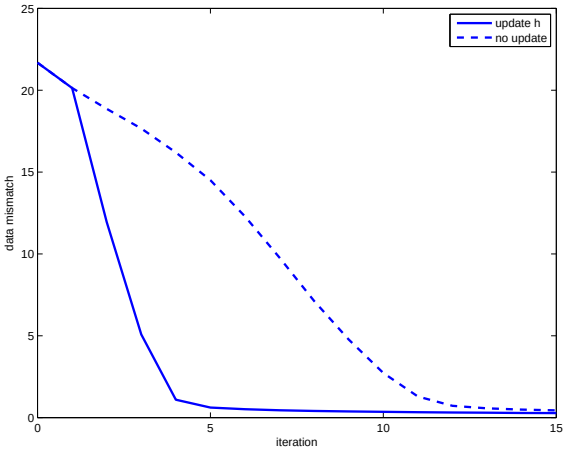
Table 4.3: $X_{true} = 5$. Minimal Data mismatch within 15 iterations.

h	update	k=2	k=3	k=4
0.01	yes	0.3230	0.2511	0.2836
0.01	no	0.5129	0.4542	0.9777
0.05	yes	0.3188	0.2645	0.2740
0.05	no	0.5856	0.3281	0.4091
0.1	yes	0.2711	0.2649	0.2885
0.1	no	0.2876	0.2753	0.2868
0.2	yes	0.2870	0.2475	0.2772
0.2	no	0.2598	0.2552	0.2694
0.3	yes	0.2925	0.2650	0.2740
0.3	no	0.2723	0.2476	0.2764
EnKS	NA	1.0147	1.0032	1.0035

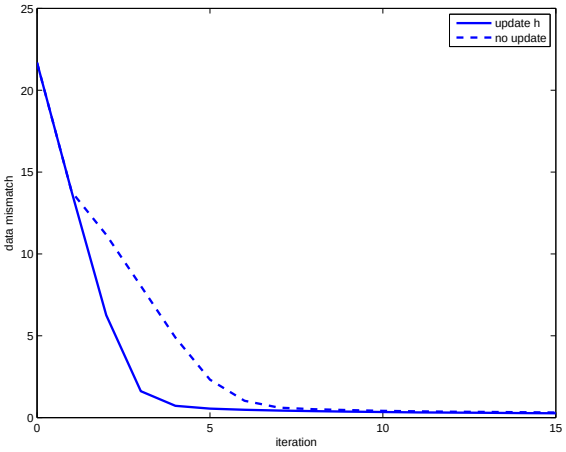
In order to look into one specified case about how the data mismatch changes after each iteration and how the updated h is, I will take the case with $k = 3$ and $X_{true} = 4$ as an example. See Table 4.4 and 4.5 and Figure 4.1.

Table 4.4: Data Mismatch when $k = 3$ and $X_{true} = 4$.

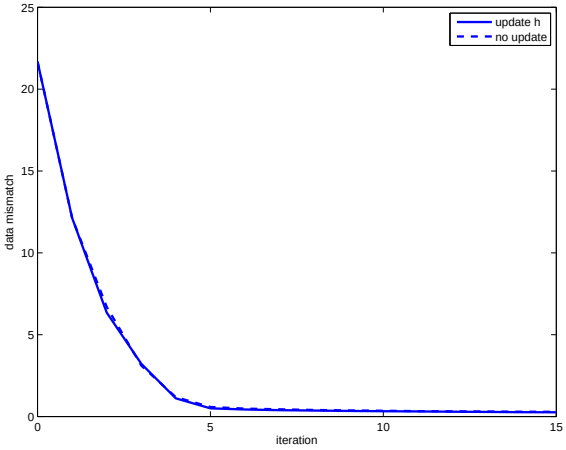
h	0.01	0.01	0.05	0.05	0.10	0.10	0.20	0.20	0.30	0.30
update	yes	no	yes	no	yes	no	yes	no	yes	no
initial	21.6751	21.6751	21.6751	21.6751	21.6751	21.6751	21.6751	21.6751	21.6751	21.6751
iter=1	20.1358	20.1358	13.7967	13.7967	12.1358	12.1358	11.9491	11.9491	11.7085	11.7085
2	11.9496	18.8601	6.2588	11.1659	6.3545	6.7001	3.8071	15.9249	11.3993	6.9221
3	5.0771	17.6612	1.6116	8.0462	3.2190	3.1147	1.0026	1.8885	2.2912	1.5207
4	1.0932	16.1971	0.7202	4.8900	1.1099	1.1822	0.5674	0.7943	0.7677	0.7352
5	0.6105	14.4988	0.5465	2.3151	0.5010	0.5820	0.4881	0.5350	0.4656	0.5496
6	0.5094	12.2842	0.4766	1.0252	0.4276	0.4837	0.4339	0.4587	0.4152	0.4705
7	0.4483	9.7651	0.4271	0.6049	0.3903	0.4387	0.4013	0.4134	0.3843	0.4196
8	0.4077	7.1235	0.3932	0.5093	0.3641	0.3976	0.3689	0.3806	0.3601	0.3834
9	0.3735	4.7663	0.3648	0.4568	0.3412	0.3682	0.3425	0.3575	0.3410	0.3557
10	0.3492	2.7222	0.3427	0.4127	0.3235	0.3500	0.3229	0.3384	0.3245	0.3347
11	0.3278	1.2884	0.3156	0.3830	0.3078	0.3291	0.3086	0.3206	0.3094	0.3128
12	0.3116	0.7210	0.3046	0.3599	0.2917	0.3155	0.2964	0.3079	0.2971	0.2940
13	0.2962	0.5639	0.2943	0.3456	0.2803	0.3018	0.2832	0.2954	0.2829	0.2792
14	0.2832	0.4874	0.2731	0.3268	0.2632	0.2946	0.2725	0.2837	0.2741	0.2652
15	0.2749	0.4404	0.2607	0.3108	0.2554	0.2844	0.2607	0.2745	0.2637	0.2521



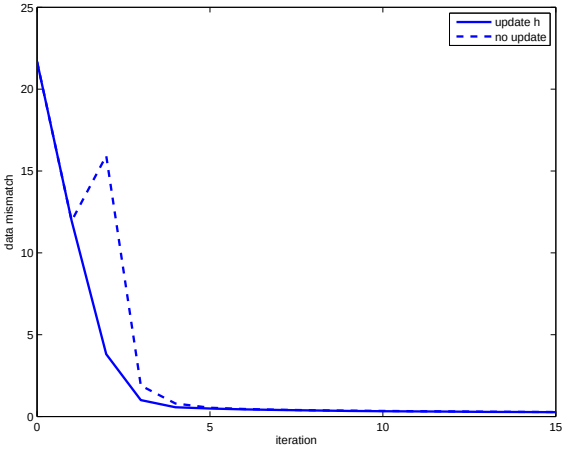
(a) $h_{initial} = 0.01$



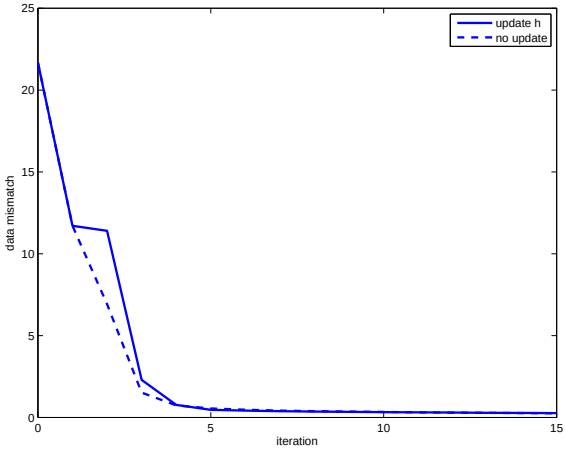
(b) $h_{initial} = 0.05$



(c) $h_{initial} = 0.1$



(d) $h_{initial} = 0.2$



(e) $h_{initial} = 0.3$

Figure 4.1: Toy Model II Data Mismatch of IAGS with different strategies.

Table 4.5: The bandwidth h of IAGS applied on Toy Model II, when $k = 3$ and $X_{true} = 4$.

h	0.01	0.01	0.05	0.05	0.10	0.10	0.20	0.20	0.30	0.30
update	yes	no	yes	no	yes	no	yes	no	yes	no
iter=1	0.0100	0.0100	0.0500	0.0500	0.1000	0.1000	0.2000	0.2000	0.3000	0.3000
2	0.0674	0.0100	0.2386	0.0500	0.1038	0.1000	0.1959	0.2000	0.1846	0.3000
3	0.2833	0.0100	0.1930	0.0500	0.0816	0.1000	0.3156	0.2000	0.1560	0.3000
4	0.3595	0.0100	0.6849	0.0500	0.1463	0.1000	0.8982	0.2000	0.2596	0.3000
5	0.6547	0.0100	0.4449	0.0500	0.4769	0.1000	0.3221	0.2000	0.8129	0.3000
6	0.2885	0.0100	0.1986	0.0500	0.4614	0.1000	0.2021	0.2000	0.2550	0.3000
7	0.2470	0.0100	0.1237	0.0500	0.2150	0.1000	0.2343	0.2000	0.2218	0.3000
8	0.2214	0.0100	0.1518	0.0500	0.1701	0.1000	0.2231	0.2000	0.1804	0.3000
9	0.2103	0.0100	0.1352	0.0500	0.1725	0.1000	0.2068	0.2000	0.1598	0.3000
10	0.1740	0.0100	0.0960	0.0500	0.1247	0.1000	0.2290	0.2000	0.1677	0.3000
11	0.1888	0.0100	0.5482	0.0500	0.0858	0.1000	0.1763	0.2000	0.1413	0.3000
12	0.1652	0.0100	0.1259	0.0500	0.5258	0.1000	0.1482	0.2000	0.1117	0.3000
13	0.1288	0.0100	0.0450	0.0500	0.0885	0.1000	0.1285	0.2000	0.1164	0.3000
14	0.1150	0.0100	0.9918	0.0500	0.6068	0.1000	0.0585	0.2000	0.1148	0.3000
15	0.0651	0.0100	0.3707	0.0500	0.0934	0.1000	0.3331	0.2000	0.1049	0.3000

According to Table 4.4 and Figure 4.1, we can see that the IAGS with updating h is a way to accelerate the original IAGS. For example, when we start with $h = 0.01$, the result of IAGS with updating h is better than that got from original IAGS only after 7 iterations. The similar situation happens when $h = 0.05$. For the case $h = 0.1, 0.2$ or 0.3, although the acceleration is not significant, the result from IAGS with updating h is still slightly better than that without updating.

Moreover, if we look into the Table 4.1, 4.2 and 4.3 for some general review of all the scenarios, we can see that starting the IAGS with $h = 0.01$ and doing update on h after each iteration will yield a good assimilation result. In general, starting IAGS with $h = 0.01$ will cause no bias problem for any models, so we can view that starting with small h and updating h according to $\Delta\mathbf{NL}$ and $\Delta\mathbf{Inn}$ as a stable strategy for any cases.

4.2. Simulation with a Reservoir Model

To test the performance of IAGS with updating bandwidth h in a more realistic model, I use the "simsim_inp_het_five_spot" model from the "SimSim" simulator, which is introduced in the introduction chapter. The simulation period is chosen as 240 ($days$), with the measurement data taken on $t_{ob} = [30 : 30 : 240](day)$.

4.2.1. Sample Size $N_{sample} = 30$

The IAGS method is performed with different sample sizes N_{sample}. When choosing the $N_{sample} = 30$, the objective function values of IAGS with different strategies are shown in Table 4.6. After the iterations where the objective function value begins

Table 4.6: Objective Function Value, $N_{sample} = 30$

	0.01	0.01	0.05	0.05	0.1	0.1	EnKS
update	Yes	No	Yes	No	Yes	No	NA
Initial	1422.22	1422.22	1422.22	1422.22	1422.22	1422.22	1422.22
Iter=1	1172.90	1172.90	215.22	215.22	79.65	79.65	134.45
2	84.83	1103.52	87.76	157.51	(87.71)	74.53	
3	(100.31)	493.29	(90.12)	118.49		(83.25)	
4		309.02		119.36			
5		246.76		104.90			
6		(271.11)		(110.81)			

to increase, the assimilation result is not shown in Table 4.6. It can be seen that, similar to the Toy Model II, when applying the IAGS with updating h, the objective function value decreases to its minimal value within 3 iterations. However, if we use simply the original IAGS method, the decreasing of the objective function value is quite slow.

For the IAGS with updating bandwidth parameter h,the h is shown in Table 4.7.

Table 4.7: $N_{sample} = 30$, bandwidth h in the updated IAGS

Iteration	1	2	3
h=0.01	0.0100	0.1038	0.0484
h=0.05	0.0500	0.2425	0.2158
h=0.1	0.1000		

The plot of the corresponding updated permeability is shown in Figure 4.2 and 4.3. For the case where $h_{initial} = 0.1$, the result from updated IAGS and the original IAGS is same, so I did not display the permeability plot of it.

4.2.2. Sample Size $N_{sample} = 100$

Since there is no confirmation step, the computation of smoother methods is relatively cheaper compared to other Bayesian methods. I also test the performance of IAGS with some large sample size. When choosing the $N_{sample} = 100$, the objective function values of IAGS with different strategies are shown in Table 4.8.

Table 4.8: Objective Function Value, $N_{sample} = 100$

	0.01	0.01	0.05	0.05	0.1	0.1	EnKS
update	Yes	No	Yes	No	Yes	No	NA
Initial	1608.84	1608.84	1608.84	1608.84	1608.84	1608.84	1608.84
Iter=1	1462.33	1462.33	405.56	405.56	78.08	78.08	77.05
2	346.83	1553.96	55.87	219.63	58.88	53.88	
3	76.86	1464.40	(57.76)	146.99	48.68	51.13	
4	76.55	1221.90		114.27	(53.55)	(59.47)	
5	(76.92)	928.43		99.46			
6		660.57		93.02			
7		424.1156		86.1092			
8		378.6593		(103.089)			
9		358.3235					
10		(359.4592)					

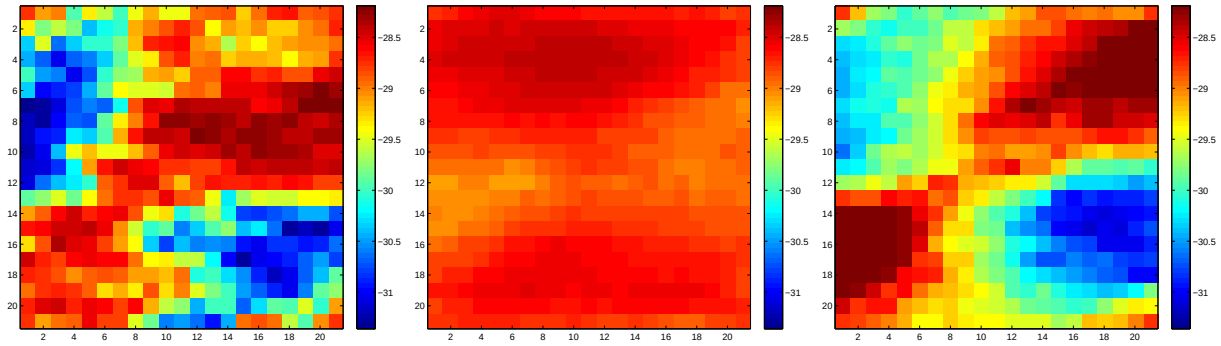
For the IAGS with updating bandwidth parameter h,the h is shown in Table 4.9.

Table 4.9: $N_{sample} = 100$, bandwidth h in the updated IAGS

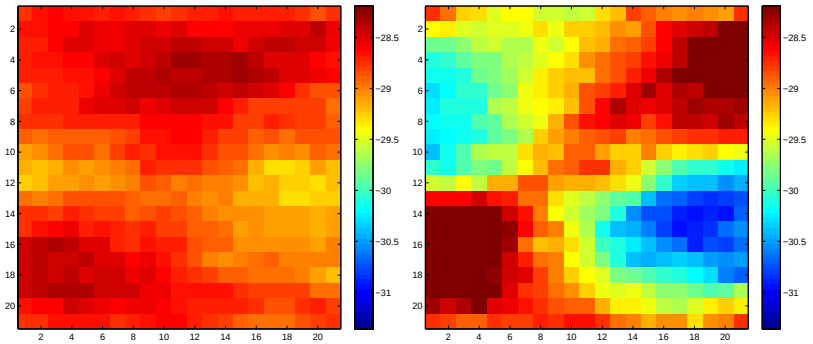
Iteration	1	2	3	4	5
h=0.01	0.0100	0.0572	0.4341	0.1648	0.1385
h=0.05	0.0500	0.3113			
h=0.1	0.1000	0.0623	0.3958		

The permeability plots for different $h_{initial}$ are shown in Figure 4.4, 4.5 and 4.6.

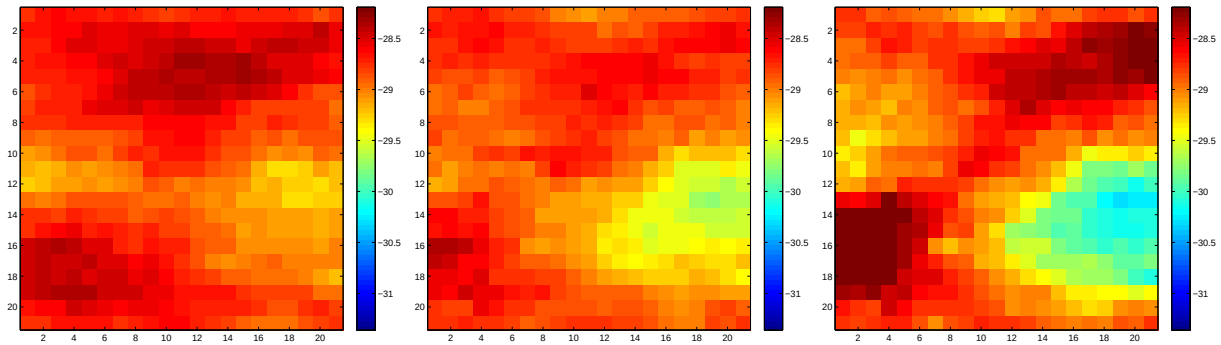
To sum up, the updated IAGS still works better than the original IAGS, and also better than the EnKS method.



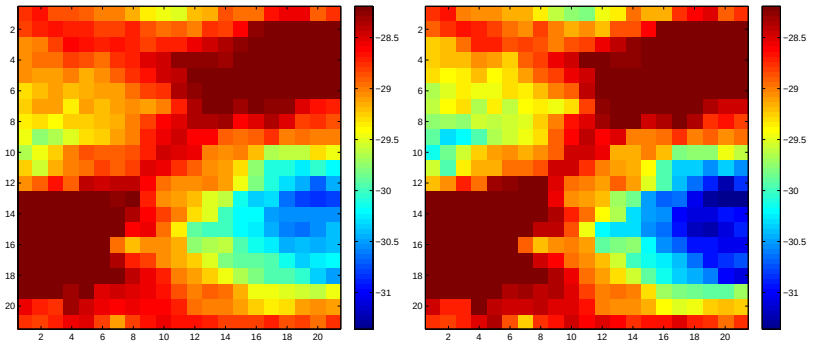
(a) True **(b)** Prior **(c)** EnKS



(d) update h, $Iter = 1$ **(e)** update h, $Iter = 2$

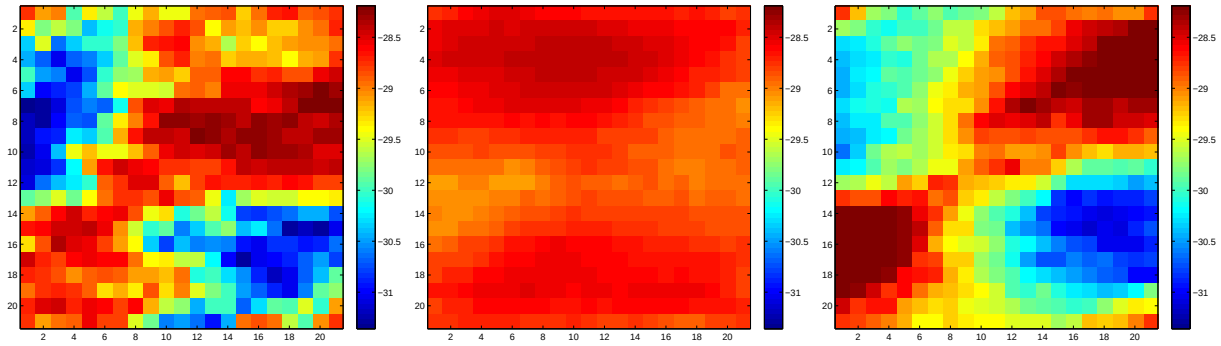


(f) no update, $Iter = 1$ **(g)** no update, $Iter = 2$ **(h)** no update, $Iter = 3$

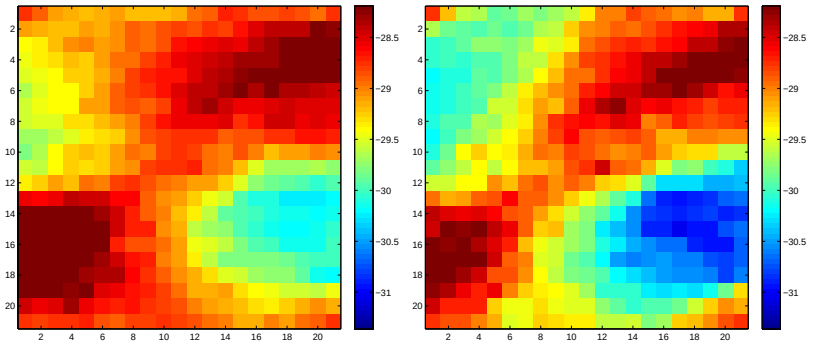


(i) no update, $Iter = 4$ **(j)** no update, $Iter = 5$

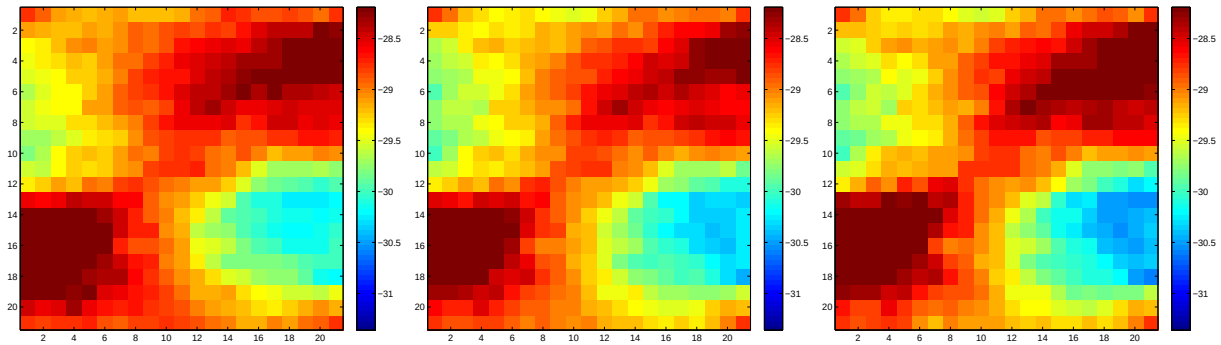
Figure 4.2: $N_{sample} = 30$, $h_{initial} = 0.01$, Permeability Plot of Reservoir Model



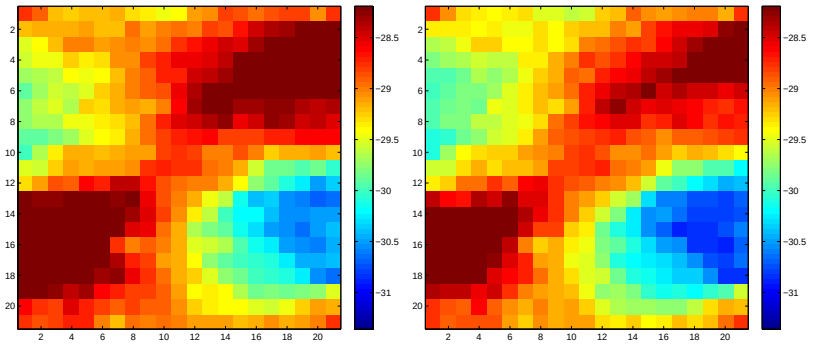
(a) True **(b)** Prior **(c)** EnKS



(d) update h, $Iter = 1$ **(e)** update h, $Iter = 2$

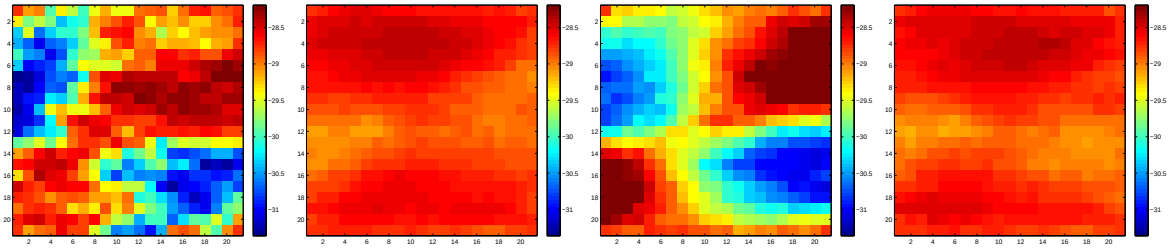


(f) no update, $Iter = 1$ **(g)** no update, $Iter = 2$ **(h)** no update, $Iter = 3$

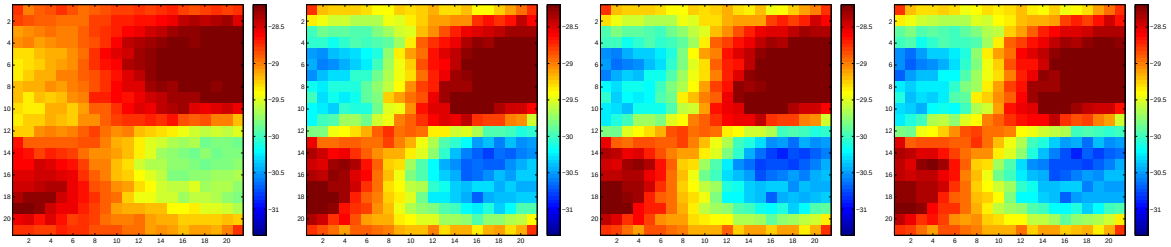


(i) no update, $Iter = 4$ **(j)** no update, $Iter = 5$

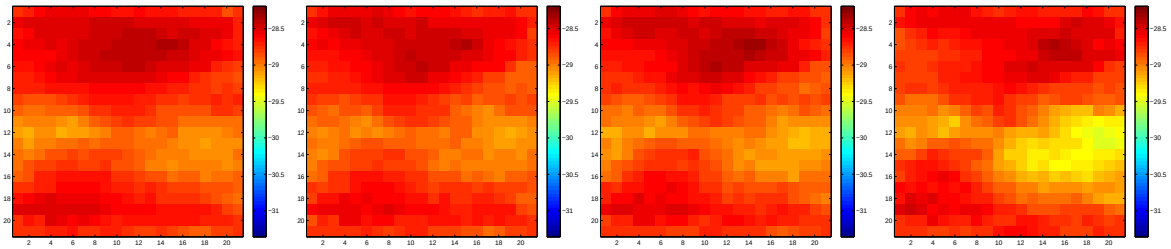
Figure 4.3: $N_{sample} = 30$, $h_{initial} = 0.05$, Permeability Plot of Reservoir Model



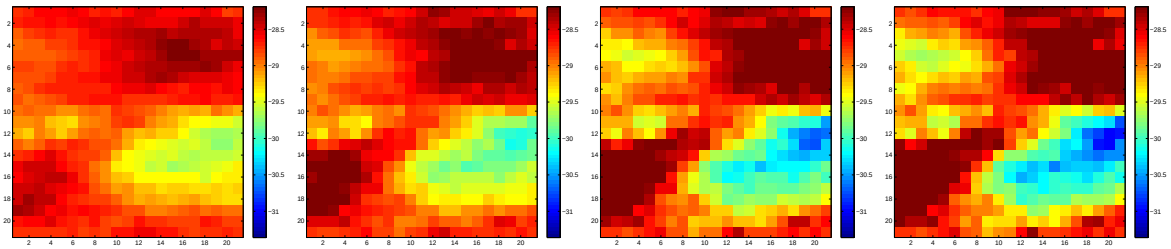
(a) True **(b)** Prior **(c)** EnKS **(d)** update h, $Iter = 1$



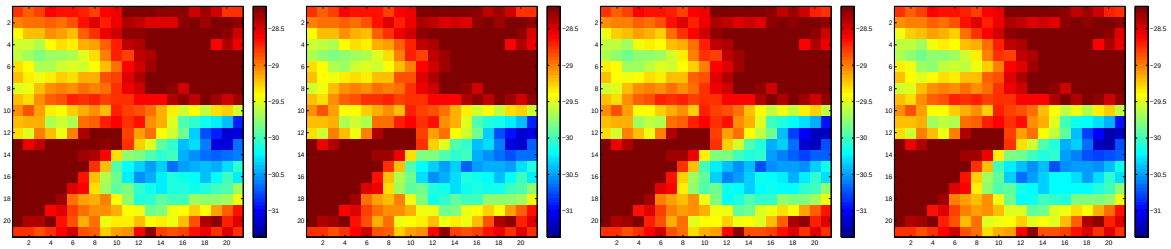
(e) update h, $Iter = 2$ **(f)** update h, $Iter = 3$ **(g)** update h, $Iter = 4$ **(h)** update h, $Iter = 5$



(i) No update, $Iter = 1$ **(j)** No update, $Iter = 2$ **(k)** No update, $Iter = 3$ **(l)** No update, $Iter = 4$

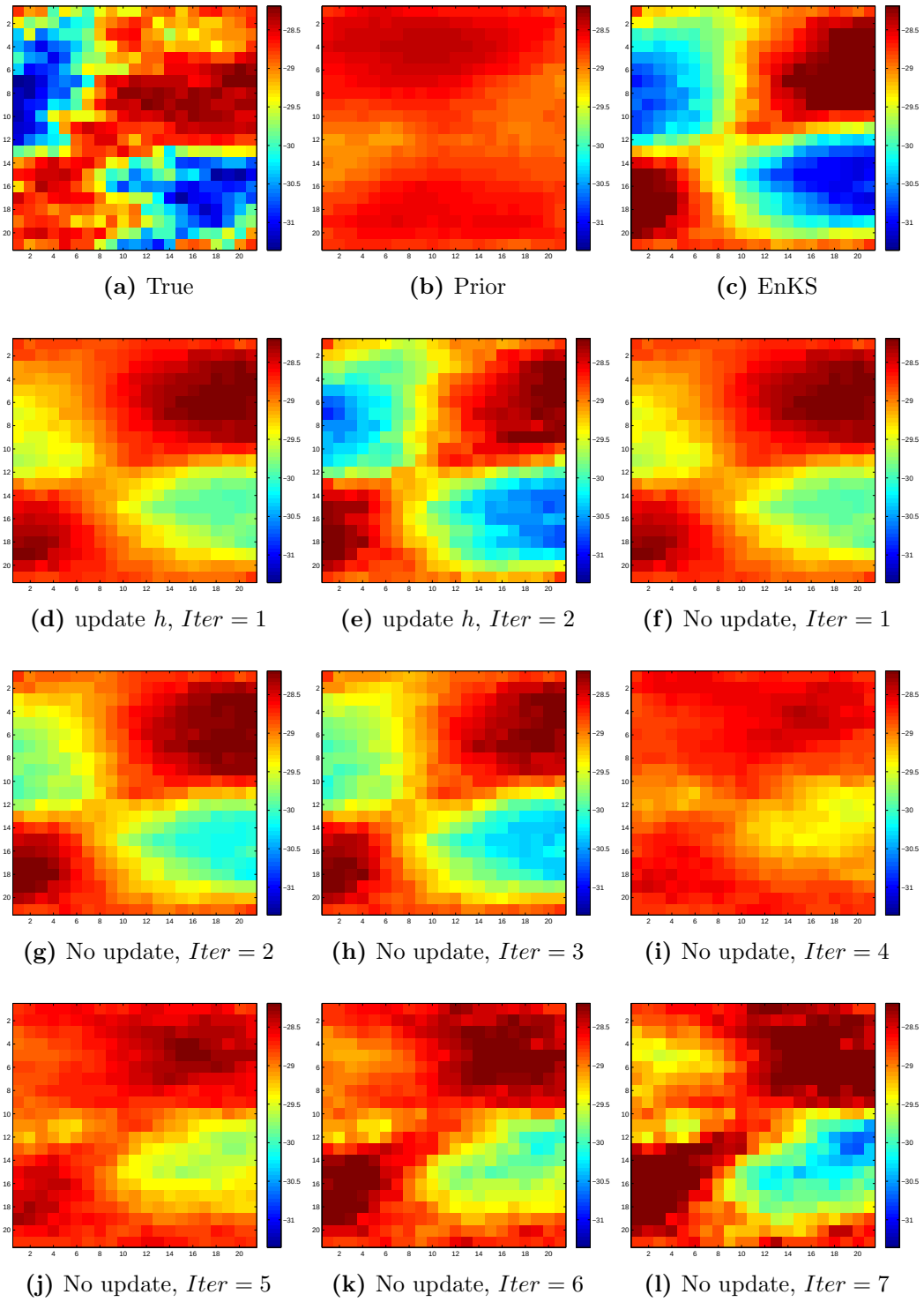


(m) No update, $Iter = 5$ **(n)** No update, $Iter = 6$ **(o)** No update, $Iter = 7$ **(p)** No update, $Iter = 8$



(q) No update, $Iter = 9$ **(r)** No update, $Iter = 10$ **(s)** No update, $Iter = 11$ **(t)** No update, $Iter = 12$

Figure 4.4: $N_{sample} = 100$, $h_{initial} = 0.01$, Permeability Plot of Reservoir Model



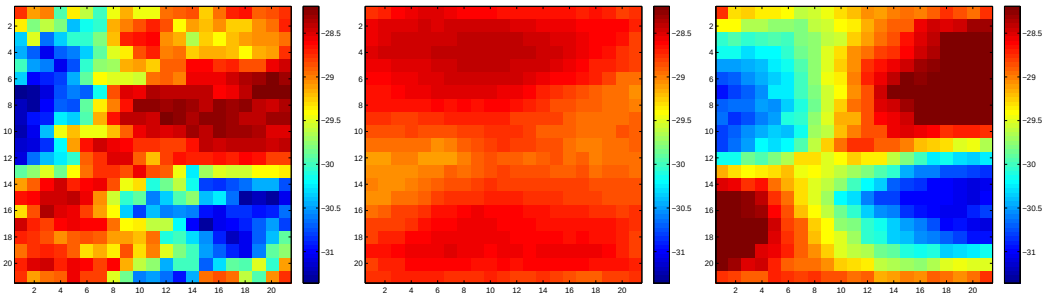
(a) True **(b)** Prior **(c)** EnKS

(d) update h, $Iter = 1$ **(e)** update h, $Iter = 2$ **(f)** No update, $Iter = 1$

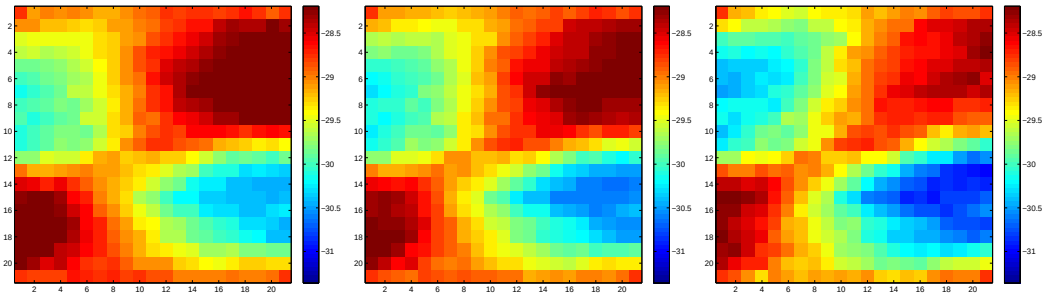
(g) No update, $Iter = 2$ **(h)** No update, $Iter = 3$ **(i)** No update, $Iter = 4$

(j) No update, $Iter = 5$ **(k)** No update, $Iter = 6$ **(l)** No update, $Iter = 7$

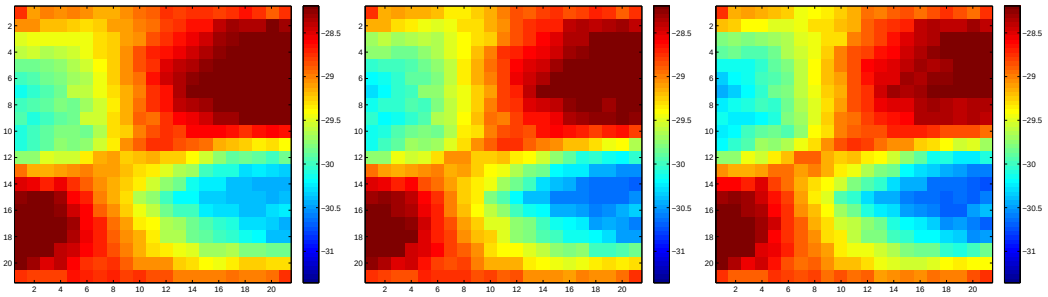
Figure 4.5: $N_{sample} = 100$, $h_{initial} = 0.05$, Permeability Plot of Reservoir Model



(a) True **(b)** Prior **(c)** EnKS



(d) update h, $Iter = 1$ **(e)** update h, $Iter = 2$ **(f)** update h, $Iter = 3$



(g) No update, $Iter = 1$ **(h)** No update, $Iter = 2$ **(i)** No update, $Iter = 3$

Figure 4.6: $N_{sample} = 100$, $h_{initial} = 0.1$, Permeability Plot of Reservoir Model

Conclusion

In this report, we research on how to accelerate the IAGS method by choosing the bandwidth parameter h_{opt} adaptively. The research starts with how to measure the model nonlinearity in a uniform way, so that the studies associated with this measure can be applied in other cases.

Since this new measure is based on the statistics from both the previous iteration and the upcoming iteration, we generate the adaptive bandwidth h_{opt} for the upcoming iteration by updating the h_{pre} for the previous one. Moreover, in order to take the assimilation performance of the previous iteration into account, we also consider the ratio of the innovations from the two iterations as an aspect influencing the updating of bandwidth h. To sum up, the adaptive bandwidth h_{opt} is computed based on three data: the newly developed nonlinearity measure, the ratio of the innovations and the previous bandwidth h_{pre}.

The multivariate regression analysis is done by splitting the domain of the previous bandwidth h_{pre} and then considering the relationship between the ratio $\frac{h_{\text{opt}}}{h_{\text{pre}}}$ and the other two statistics within each domain.

The accelerated IAGS is promising from a theoretical point of view. It starts with small bandwidth h, so that little bias will be introduced in the initial step. Besides, in the following iterations, the update of bandwidth h is highly related to the model nonlinearity and, in order to avoid introducing bias, we prefer to underestimate the adaptive bandwidth h rather than calculating that accurately with the risk of overestimation.

The result from the accelerated IAGS is satisfactory in both the Toy Model and the reservoir model experiment. The data mismatch and the objective function value decrease faster to smaller values with the accelerated IAGS than with the original one. And Both of them behave better than the EnKS.

The accelerated IAGS method requires more investigation and especially comparison with some other reasonable methods than EnKS, for example the MCMC method. This adaptive way of selecting h can also be tested on other iterative methods which are available with a dampened Kalman Update, such as ES-MDA[1] and LM-EnRML[32]. Moreover, it is possible to investigate the multivariate regression function of h_{opt} with not only splitting the domain of the previous bandwidth h_{pre}

but also splitting that of the nonlinearity measure and the ratio of the innovations. And more types of the regression functions may be considered, say a higher order polynomial function or some functions with crossterms.

Bibliography

[1] A.A. Emerick, and A.C. Reynolds, 2012: History matching time-lapse seismic data using the ensemble Kalman filter with multiple data assimilations. Comput Geosci (2012) 16:639659.

[2] A. Doucet, N. de Freitas, and N. Gordon, 2001: Sequential MonteCarlo Methods in Practice. Springer, New York.

[3] A. Kong, J. Liu, and W. Wong, 1994: Sequential imputations and Bayesian missing data problems. J. Am. Stat. Assoc. 89(425), 278288.

[4] A.S. Stordal, 2008: Sequential Monte Carlo Methods for Nonlinear Bayesian Filtering. M.Sc. thesis, Department of Mathematics, University of Bergen.

[5] A.S. Stordal, H. Karlsen, G. Nævdal, H. Skaug, and B. Vallès, 2011a: Bridging the ensemble Kalman filter and particle filters. Computational Geosciences, 15(2), 293305.

[6] A.S. Stordal, and R.J. Lorentzen, 2012: An Iterative version of the Adaptive Gaussian Mixture Filter.

[7] A.S. Stordal, 2012: On Iterative Smoothers for Bayesian Estimation.

[8] A.W. van der. Vaart, 2000: Asymptotic Statistics (Cambridge Series in Statistical and Probabilistic Mathematics). Cambridge, UK: Cambridge University Press.

[9] A.W. Heemink, and P. Wilders, Ch12: Environmental simulation and data assimilation. Unpublished.

[10] B.D. Anderson, and J.B. Moore, 2005: Optimal Filtering. Dover, New York.

[11] C. Snyder, T. Bengtsson, P. Bickel, and J. Anderson, 2008 : Obstacles to high-dimensional particle filtering. Mon. Weather Rev. 136, 46294640.

[12] D. Titterington, 1985: Statistical Analysis for Finite Mixture Distributions. Wiley, Chichester.

[13] E.R. Lichiardopol, 2009: Estimation of Relative Permeability Parameters in Reservoir Engineering Applications. M.Sc Thesis, Delft Institute of Applied Mathematics, Delft University of Technology.

[14] F. Cong, 2013: Nonlinearity Measurement based on IAGM. Internship Report of Course WI5012 in TU Delft.

[15] G. Evensen, 1994: Sequential data assimilation with nonlinear quasi-geostrophic model using monte carlo methods to forecast error statistics. J.Geophysics. Res.99.

[16] G. Givens and A. Raftery, 1996: Local adaptive importance sampling for multivariate densities with strong nonlinear relationships. Journal of American Statistical Association, 91(433):132141.

[17] J.H. Kotecha, and P.M. Djuri´c, 2003: Gaussian sum particle filtering. IEEE 51(10), 26022612.

[18] L. Ljung, 1979: Asymptotic behavior of the extended Kalman filter as a parameter estimator for linear systems. Automatic Control, IEEE Transactions on, 24(1), 36-50.

[19] Little, Roderick J. A., and Donald B. Rubin, 2002: Statistical Analysis with Missing Data. 2nd ed., Hoboken, NJ: John Wiley & Sons, Inc.

[20] L. Yang,, and R. Tschernig, 1999: Multivariate bandwidth selection for local linear regression. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 61.4: 793-815.

[21] M.V. Krymskaya, 2007: Parameter Estimation in Reservoir Engineering Models via Data Assimilation Techniques. M.Sc Thesis, Delft Institute of Applied Mathematics, Delft University of Technology.

[22] M. Verlaan, and A.W. Heemink, 1999: Nonlinearity in data assimilation applications:A practical method for analysis. Mon. Wea. Rev., 129, 15781589.

[23] N. Gordon, 1993: Bayesian Methods for Tracking. Ph.D. thesis, University of London.

[24] R. Chen, and J.S. Liu, 2000: Mixture Kalman filters. J. R. Stat. Soc., Ser. B Stat. Methodol. 60, 493508.

[25] R.G. Hanea, G.J.M. Velders, A.J. Segers, M. Verlaan, and A.W. Heemink, 2006: A Hybrid Kalman Filter Algorithm for Large-Scale Atmospheric Chemistry Data Assimilation. Mon. Wea. Rev., 135, 140151.

[26] R. Valestrand, G. Nævdal, A. Shafieirad, A.S. Stordal, and L. Dovera, 2012a: Refined adaptive Gaussian mixture filter - Application on a real field case. EAGE Annual Conference & Exhibition incorporating SPE Europec, Copenhagen, Denmark.

[27] R. Valestrand, G. Nævdal, and A.S. Stordal, 2012b: Evaluation of EnKF and variants on a field model. Oil & Gas Science and Technology- Revue dIFP Energies nouvelles 2012.

[28] Stephen P. Brooks, 1998: Markov chain Monte Carlo method and its application. Journal of the Royal Statistical Society. Series D (The Statistician), Vol. 47, No. 1, 69-100.

[29] Stephen Tyson, 2007: An Introduction to Reservoir Modeling, ISBN 978-1-906928-07-0.

[30] T. Bengtsson, P. Bickel, and B. Li, 2008: Curse-of-dimensionality revisited: collapse of particle filter in very large scale systems. IMS Collect. 2, 316334.

[31] T. Bengtsson, C. Snyder, and D. Nychka, 2003: Toward a nonlinear ensemble filter for high-dimensional systems. J. Geophys. Res. 108, 3545.

[32] Y. Chen, and D.S. Oliver, 2013: History Matching Of The Norne Full Field Model Using An Iterative Ensemble Smoother. 75th EAGE Conference & Exhibition incorporating SPE EUROPEC , June 10 - 13, 2013, London, United Kingdom.

Appendix A

List of Symbols and Abbreviations

Table A.1: List of Abbreviations

Abbreviation	Description
EnKF	Ensemble Kalman Filter
EnKS	Ensemble Kalman Smoother
SIS	Sequential Importance Sampling
SIR	Sequential Importance Reampling
AGM	Adaptive Gaussian Mixture
IAGM	Iterative Adaptive Gaussian Mixture
IAGS	Iterative Adaptive Gaussian Smoother

Table A.2: List of Symbols I

Sympol	Description
ξ^i	ensemble member i
ξ_k^i	prior ensemble member i at time step k
$\hat{\xi}_k^i$	posterior ensemble member i at time step k
$\mathcal{M}_k(\xi_i)$	measurement from ensemble member i at time step k
$\mathbf{R}$	covariance matrix of measurement error
y_k	true measurement at time step k
N	ensemble size
N_d	number of measurements
D	data mismatch
$O(\xi_i)$	objective function of ensemble member i
OB	mean objective function value of all the ensemble members
w_k^i	weight of ensemble member i at time step k
$\mathbf{X}_k$	prior state vector at time step k
$\hat{\mathbf{X}}_k$	posterior state vector at time step k
$\tilde{\mathbf{X}}_k$	extended state vector at time step k
$\mathbf{Y}_k$	measurement vector at time step k
$\mathbf{M}_k$	transformation matrix from extended state vector $\tilde{\mathbf{X}}_k$ to measurement vector $\mathbf{Y}_k$
$f(\cdot)$	forward operator of the state vector
η_k	model error
ϵ_k	measurement error
$\mathbb{Q}$	covariance matrix of model error
μ_p	prior mean of state vector
$\mathbb{C}_p$	prior covariance matrix of state vector
$\mathbf{P}_k$	approximated covariance matrix of ensemble members
d_s	length of state vector
d_m	length of measurement vector
$\mathbf{F}_k$	linear approximation of forward operatior $f(\cdot)$
$p(\mathbf{X}_{0:k}\|\mathbf{Y}_{1:k} = y_{1:k})$	posterior density function of state vectors $\{\mathbf{X}_j\}_{j=0}^k$ based on the measurement $\{y_j\}_{j=1}^k$
$g(\mathbf{X}_{0:k}\|\mathbf{Y}_{1:k} = y_{1:k})$	prescribed importance function of state vectors $\{\mathbf{X}_j\}_{j=0}^k$ based on the measurement $\{y_j\}_{j=1}^k$
$\hat{N}_{\text{eff}}$	effective ensemble size
$\Phi(\mathbf{X} - \mu, \mathbf{P})$	Gaussian density function with mean μ and convariance matrix $\mathbf{P}$
N_c	prescribed ensemble size

Table A.3: List of Symbols II

Sympol	Description
h	bandwidth parameter of AGM
α	adaptive shrink parameter of AGM
α_{opt}	optimal adaptive shrink parameter of AGM
$\mathbf{X}_{true}$	true parameter in Toy Problem
D_{new}	updated data mismatch in Toy Problem
D_{old}	original data mismatch in Toy Problem
P_{update}	update performance of the AGM in theToy Problem
$\mathbf{NL}_j$	original nonlinearity measurement at iteration j
$\mathbf{\Delta NL}_j$	modified nonlinearity measurement at iteration j
$\mathbf{\Delta Inn}_j$	innovation measurement at iteration j
h_{opt}	optimal bandwidth parameter
h_{pre}	bandwidth used in the previous iteration
σ_{err}	standard variance of measurment error in Toy Problem
μ_{err}	mean of measurment error in Toy Problem
σ	standard variance of state parameter in Toy Problem
μ	mean of state parameter in Toy Problem
h_{initial}	initial bandwidth parameter used in IAGS or IAGM
N_{sample}	ensemble size used in the Reservoir model