Seeing Invisible Properties of Subsurface Oil and Gas Reservoir through Extensive Uses of Machine Learning Algorithms

Kwangwon Park

Abstract—**Current geostatistical simulation methods allow generating multiple realizations that honor all available data, such as hard and secondary data under certain geological scenarios (e.g. 3D training image-based models, multi-Gaussian law, Boolean models). However, it is difficult to simulate large models that honor highly nonlinear response functions (e.g. remote sensing data, geophysical data or flow in porous media data). The large CPU cost to evaluate the response function imposes limitations on the size of the model. This is particularly the case when one desires to generate a sizeable set of realizations matching the same data. The objective of this study is to generate multiple realizations all of which honor all available data (hard and secondary data and especially the non-linear response function) under certain geological scenarios. First, we generate a large ensemble of possible realizations describing the spatial uncertainty for given hard and secondary data. Any fast geostatistical simulation methods can be used for generating these prior models. Secondly, using multidimensional scaling, we map these models into a low-dimensional metric space by defining a distance between these prior models. We propose to make this distance a function of the response function. Next, kernel principal component analysis is applied to the coordinates of realizations mapped in this metric space to create a kernel, or feature, space with linear/Gaussian type variability between the input realizations. In this space, we can apply optimization algorithms, such as gradient-based algorithms or ensemble Kalman filtering or gradual deformation method to generate multiple realizations matching the same data. A back-transformation, first from the kernel space, then to metric space and finally to the actual space of realizations allows then the generation of multiple geostatistical models that match all data, hard, secondary and non-linear response. We apply the proposed framework to generate a realistic model which honors geologic information and dynamic (time-varying) response data. A flow simulator is used as the non-linear response function and may require several hours of CPU-time per simulation. We show how this technique applies to non-Gaussian (e.g. multiple-point or Boolean) geostatistical models. We also demonstrate the importance of using a distance function tailored to the particular response function used in creating a low-dimensional parameterization of the ensemble of geostatistical model in feature space.**

Index Terms—**CS229, Machine Learning, Optimization, Reservoir**

I. Introduction

IN order to estimate oil and gas reserves and maximize their production, it is essential to find out unknown reservoir properties, such as geologic subsurface structures. Reservoir engineers often use a reservoir simulator that provides future prediction about oil and gas flow performance, which is a kind of PDE (partial differential equation) solvers with various reservoir properties as input parameters. However, we can seldom have enough information about these input parameters due to the limitation to access very deep subsurface formation. Those parameters which can not be measured directly are often obtained by solving optimization problem because we can measure output data only, such as production rates or pressure variation at production wells.

Kwangwon Park is with the Department of Energy Resources Engineering, Stanford University, Stanford, CA 94305. Phone: +1 650 725–1324, e-mail: wonii@stanford.edu

However, optimization problems for ascertaining reservoir parameters are containing a lot of tough limitations. First, we usually have to determine 10^6 to 10^8 number of input parameters, which means input space is extremely high dimensional. Second, single evaluation of objective function (difference between observed and simulated, in other words misfit function) usually takes several hours to several days. Moreover there are lots of local minima in input space due to highly nonlinear and extremely large system of partial differential equations in reservoir simulation. Third, our optimization is limited by very few information about reservoir, as we have very few wells, which are the only way for us to measure the reservoir input and output data. Fourth, since all those parameters are related with geologic subsurface structures, the input parameters should be geologically realistic.

In the process of this tough optimization problem, various machine learning algorithms can be introduced extensively. The problem, theory, methodology, and simple results will be followed.

II. Problem Statements

THE ultimate objective is to solve an inverse problem, which can be represented by Equation 1.

$$x = f_\alpha^{-1}(y) \tag{1}$$

where, x is the unknown input parameters (reservoir properties) and y is the output (solution variables). α represents the known parameters. f means a PDE as Equation 2.

$$\frac{\partial}{\partial t}(\alpha y) - \nabla \cdot (\alpha x \nabla y) + \alpha = 0 \tag{2}$$

where, t represents time. Equation 2 can be solved by various numerical methods with the initial and boundary conditions that are controlled by α. Equation 1 and 2 have several interesting aspects:

Nonlinearity: Since α is a function of y, Equation 2 is a highly nonlinear equation, which also increases the simulation. Besides, although we know all the y's in

the domain (actually rarely possible in field), it is very difficult to solve the inverse problem (Equation 1).

Large number of parameters: Usually we have to determine several unknown properties at each node, which means the dimension of x is 10^6 to 10^8. In other words, we have to solve a highly nonlinear inverse partial difference equation in extremely high dimension. It is needless to say that a lot of local minima exist in the space.

Large number of nodes: In order to get the desired and meaningful resolution from Equation 2, we usually have to make 10^5 to 10^7 nodes at each of which the output y has to be calculated in 3-dimensional space. Large number of nodes result in dramatic increase in forward simulation time (hours to days per single simulation).

Limited output data: Actually, we can obtain very limited information of y, as we have to install a kind of measurement device in the deep subsurface, which costs a lot. In a reservoir, the measurement device is equivalent to wells (injectors or producers) and drilling one well costs usually millions to tens of millions of dollars. In many cases, we know a few (tens to thousands) measurements of y among 10^6 to 10^8 values.

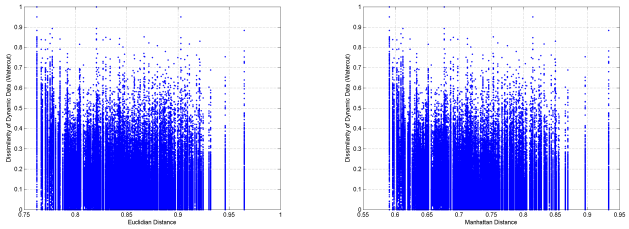
Lots of constraints: In this optimization problem, we have a lot of constraints on x. In a reservoir, x is reservoir properties, such as permeability or porosity, which are highly dependent upon subsurface rock properties. Therefore, the main constraint is that x should be geologically realistic.

With all these challenges, it is often impossible to solve this optimization problem using conventional optimization techniques themselves. Therefore, somewhat different methods will be applied and in each step the appropriate machine learning algorithms are going to be applied effectively.

III. Distances

THE distance is a measure of dissimilarity between two realizations. Simply, we can evaluate the dissimilarity between realizations $\mathbf{x_a}$ and $\mathbf{x_b}$ (discretized into N_{gb} gridblocks) through the Minkowski model, such as Euclidian space or City-Block (Manhattan) space. Although the Minkowski model is easy to calculate, it may not be well-correlated with dynamic data because the forward simulated dynamic data may change dramatically by perturbing a small portion of the realization. Figure 1 depicts the correlation between Euclidian or Manhattan distance and the dissimilarity between dynamic data (the difference between watercut curves). It turns out that both Euclidian and Manhattan distances are not correlated with the dynamic data.

In order to optimize an inverse solution efficiently in the distance space, it is necessary that the dynamic data are spatially correlated in the space. For this, various distances may be utilized. If we need Euclidian distance (actually we sometimes have to deal with Euclidian distance for satisfying the metric axioms or the positive definiteness of the



(a) Euclidian distance (b) Manhattan distance

Fig. 1. The distance and dissimilarity of dynamic data (watercut). On the y-axis is the difference in watercut between any two realizations. On the x-axis the distance between any two realizations.

distance matrix, for example when employing the RBF kernels), then any similarity distance can be easily converted to Euclidian distance by means of multidimensional scaling (MDS) or principal coordinate analysis.

IV. Parameterization

PRIOR to the parameterization of the geological model space, we start from an ensemble of realizations, $\mathbf{x}_j$, $(j = 1, ..., N_R$, if we generate N_R realizations). $\mathbf{x}$ could represent a facies porosity or permeability model or any combination of those. The initial ensemble can be generated by various geostatistical algorithms honoring the geologic information and conditioning to the static data (hard and soft data). For simplicity, define the matrix for the ensemble $\mathbf{X}$ as

$$[\mathbf{X}]_{:,j} = \mathbf{x}_j \quad , \tag{3}$$

where, $[\mathbf{X}]_{i,j}$ means (i, j) element of matrix $\mathbf{X}$ and $(:, j)$ is j-th column of matrix $\mathbf{X}$. The covariance of the ensembles is calculated numerically by Equation 4.

$$\mathbf{C} = \frac{1}{N_R}\sum_{j=1}^{N_R}\mathbf{x}_j\mathbf{x}_j^{\mathrm{T}} = \frac{1}{N_R}\mathbf{X}\mathbf{X}^{\mathrm{T}} \tag{4}$$

When we perform an eigenvalue decomposition on the covariance (Equation 5), then a new realization can be obtained by Equation 6 (Karhunen-Loeve expansion).

$$\mathbf{C}\mathbf{v} = \lambda\mathbf{v} \tag{5}$$

$$\mathbf{x}_{new} = \mathbf{E}\mathbf{\Lambda}^{1/2}\boldsymbol{\xi}_{new} \tag{6}$$

where, $\mathbf{v}$ and λ is the eigenvector and eigenvalue, respectively. $\mathbf{E}$ is a matrix each column of which is the eigenvector of the covariance. $\mathbf{\Lambda}$ is a diagonal matrix each diagonal element of which is the eigenvalue of the covariance. $\boldsymbol{\xi}_{new}$ is the parameter vector for the realization $\mathbf{x}_{new}$. The parameter $\boldsymbol{\xi}$ is Gaussian random vector and the size is determined by how many eigenvalues are chosen to retain. We do not have to use all the nonzero N_R eigenvalues; typically a few large eigenvalues are retained. By Equation 6, we can generate many models based on the same covariance.

In order to consider higher-order moments or spatial correlation beyond the point-by-point covariance, the feature expansions of the realizations can be introduced. Let ϕ be

the feature map from realizations space $\mathbb{R}$ to feature space $\mathbb{F}$ (Equations 7 and 8).

$$\phi : \mathbb{R} \rightarrow \mathbb{F} \tag{7}$$

$$\mathbf{x} \mapsto \boldsymbol{\phi} := \phi(\mathbf{x}) \tag{8}$$

where $\boldsymbol{\phi}$ is the feature expansion of realization. With the feature expansions of the ensemble $\phi(\mathbf{X})$ (defined by Equation 9), the new feature expansion can be generated in the same manner above (Equation 11). The covariance of the feature expansions ($\phi(\mathbf{x}_j)$) of the ensemble is calculated by Equation 10.

$$[\phi(\mathbf{X})]_{:,j} = \phi(\mathbf{x}_j) \tag{9}$$

$$\mathbf{C} = \frac{1}{N_R}\sum_{j=1}^{N_R}\phi(\mathbf{x}_j)\phi(\mathbf{x}_j)^{\mathrm{T}} = \frac{1}{N_R}\phi(\mathbf{X})\phi(\mathbf{X})^{\mathrm{T}} \tag{10}$$

$$\phi(\mathbf{x}_{new}) = \mathbf{E}\boldsymbol{\Lambda}^{1/2}\boldsymbol{\xi}_{new} \tag{11}$$

However, since the feature expansion is often very high-dimensional and sometimes infinite-dimensional, the eigenvalue decomposition of the covariance matrix is almost impossible. The kernel trick makes it possible to obtain the exactly equivalent solution to the eigenvalue decomposition of the covariance. If we define a kernel function as a dot product of two feature expansions (Equation 12), the kernel function can be evaluated without representing the high-dimensional feature expansions explicitly. Then, the kernel matrix (Equation 13) can be calculated efficiently.

$$k(\mathbf{x}_i, \mathbf{x}_j) := < \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) > \tag{12}$$

$$\mathbf{K} := \phi(\mathbf{X})^T\phi(\mathbf{X}) \tag{13}$$

where, $[\mathbf{K}]_{i,j}$ is $k(\mathbf{x}_i, \mathbf{x}_j)$ and $<\quad>$ means the dot product.

The main idea of the kernel trick is to assume that the new feature expansion is a linear combination of the feature expansions of the ensemble and represent all the elements in the equations as dot products of two feature expansions. Actually, Equation 11 means that a new feature expansion is a linear combination of the eigenvectors. Since the eigenvectors lie in the span of the feature expansions of the ensemble (Equation 14), it is true that the new feature expansion is a linear combination of the feature expansions of the ensemble. It turns out that the coefficient $\mathbf{a}_i$ is the i-th eigenvector of the kernel matrix and $N_R\lambda_i$ is the i-th eigenvalue of the kernel matrix (Equation 15).

$$\mathbf{v}_i = \sum_{j=1}^{N_R}[\mathbf{a}_i]_j\phi(\mathbf{x}_j) \tag{14}$$

$$\mathbf{K}\mathbf{a} = N_R\lambda\mathbf{a} \tag{15}$$

Therefore, we can acquire the new feature expansion without the costly eigenvalue decomposition of the $N_R \times N_R$ covariance matrix (Equation 16).

$$\begin{aligned}\phi(\mathbf{z}) = \mathbf{E}\boldsymbol{\Lambda}^{1/2}\boldsymbol{\xi} &= \sum_{i=1}^{N_R}[\boldsymbol{\xi}]_i\lambda_i^{1/2}\mathbf{v}_i = \sum_{i=1}^{N_R}[\boldsymbol{\xi}]_i\lambda_i^{1/2}\sum_{j=1}^{N_R}[\mathbf{a}_i]_j\phi(\mathbf{x}_j) \\ &= \sum_{j=1}^{N_R}\left\{\sum_{i=1}^{N_R}[\boldsymbol{\xi}]_i\lambda_i^{1/2}[\mathbf{a}_i]_j\right\}\phi(\mathbf{x}_j) = \sum_{j=1}^{N_R}[\mathbf{b}]_j\phi(\mathbf{x}_j) = \phi(\mathbf{X})\mathbf{b}\end{aligned} \tag{16}$$

where, $[\mathbf{b}]_j = \sum_{i=1}^{N_R}[\boldsymbol{\xi}]_i\lambda_i^{1/2}[\mathbf{a}_i]_j$

Once the new feature expansion is acquired, the new realization can be calculated from the new feature expansion ($\mathbf{x}_{new} = \phi^{-1}(\boldsymbol{\xi}_{new})$). Since ϕ^{-1} cannot often be calculated explicitly, we have to calculate the new model such that

$$\begin{aligned}\mathbf{x}_{new} &= \arg\min_{\mathbf{x}_{new}}\|\phi(\mathbf{x}_{new}) - \phi(\mathbf{X})\mathbf{b}\| \\ &= \arg\min_{\mathbf{x}_{new}}\left\{\phi(\mathbf{x}_{new})^T\phi(\mathbf{x}_{new}) - 2\phi(\mathbf{x}_{new})^T\phi(\mathbf{X})\mathbf{b} + \mathbf{b}^T\mathbf{K}\mathbf{b}\right\}\end{aligned} \tag{17}$$

This is another optimization problem which is called the pre-image problem. This optimization problem can be solved by the fixed point iteration method ((Schölkopf and Smola, 2002)). We find $\mathbf{x}_{new}$ such that

$$\nabla_{\mathbf{x}_{new}}\left\{\phi(\mathbf{x}_{new})^T\phi(\mathbf{x}_{new}) - 2\phi(\mathbf{x}_{new})^T\phi(\mathbf{X})\mathbf{b} + \mathbf{b}^T\mathbf{K}\mathbf{b}\right\} = 0 \tag{18}$$

by iterations (Equation 19).

$$\mathbf{x}_{new} = \frac{\sum_{j=1}^{N_R}[\mathbf{b}]_j k'(\mathbf{x}_j, \mathbf{x}_{new})\mathbf{x}_j}{\sum_{j=1}^{N_R}[\mathbf{b}]_j k'(\mathbf{x}_j, \mathbf{x}_{new})} \tag{19}$$

where k' means the differential of k. Since we have the kernel functions not the explicit feature expansion, these iterations can be done efficiently. In conclusion, the new realization can be obtained by a nonlinear combination of the ensemble members. Note that the nonlinear weight sum to unity.

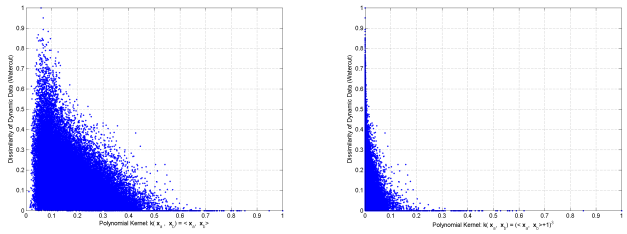
We can use various types of kernels, but the kernel matrix should be positive definite (Mercer theorem). Some widely used kernels are:

- Polynomial: $k(\mathbf{x}, \mathbf{z}) = (< \mathbf{x}, \mathbf{z} > + c)^d$
- Gaussian: $k(\mathbf{x}, \mathbf{z}) = \exp\left(-\frac{\|\mathbf{x}-\mathbf{z}\|^2}{2\sigma^2}\right)$
- Sigmoid: $k(\mathbf{x}, \mathbf{z}) = \tanh(\kappa < \mathbf{x}, \mathbf{z} > + \vartheta)$

Figure 2 shows the correlation between the polynomial kernel and the dissimilarity between dynamic data (the difference of watercut curves) of any two realizations. It turns out that polynomial kernels are not correlated with the dynamic data in this case. Since a kernel is the measure of similarity, it is desirable for the kernel to be negatively correlated with the dissimilarity between dynamic data.

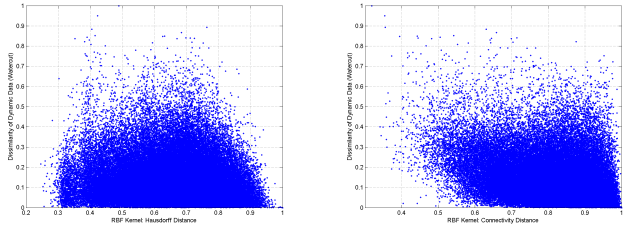
Likewise the Gaussian kernel, the kernel that is based on the Euclidian distance is called RBF kernel. Even though we know the Euclidian distance only, the RBF kernel function can be evaluated. Also, although the dissimilarity distance is not a Euclidian distance, we can map the ensemble

into the metric space by multidimensional scaling. Once the Euclidian distance in the metric space is well correlated with the dissimilarity distance, we can evaluate the kernel function by replacing the distance to the Euclidian distance in the metric space. Figure 3 depicts the correlation between the Gaussian kernel and the dissimilarity between dynamic data of any two realizations. The Hausdorff distance (Suzuki and Caers, 2006) and connectivity-based distance (Park and Caers 2007) after MDS (ten eigenvalues are retained) are used for the Euclidian distance in Gaussian kernel. The connectivity-based distance shows, to some extent, negative correlation with the dissimilarity of dynamic data.



(a) $k(\mathbf{x},\mathbf{z}) = <\mathbf{x},\mathbf{z}>$ (b) $k(\mathbf{x},\mathbf{z}) = (<\mathbf{x},\mathbf{z}>+1)^3$

Fig. 2. The polynomial kernels and dissimilarity of dynamic data (watercut). On the y-axis is the difference in watercut between any two realizations. On the x-axis the kernel between any two realizations.



(a) Hausdorff distance (b) Connectivity distance

Fig. 3. The RBF kernels and dissimilarity of dynamic data (watercut). On the y-axis is the difference in watercut between any two realizations. On the x-axis the kernel between any two realizations.

V. Further parameterization of $\boldsymbol{\xi}$

THE parameterized feature expansions of realizations ($\boldsymbol{\xi}$'s) are Gaussian random vectors, so the optimization can be accomplished by the sequential calibration with gradual deformation.

Hu (2000) developed the GDM for performing history matching on stochastic reservoir models. It consists in iteratively updating a combination of independent realizations of a random function until both static and dynamic data are matched.

Consider a Gaussian random vector $\boldsymbol{\xi}_i$ with zero mean and unit variance. The GDM consists in writing a new random vector, $\boldsymbol{\xi}_{new}$ as a linear combination of N independent random vectors (Equation 20).

$$\boldsymbol{\xi}_{new} = \sum_{i=1}^{N} \rho_i \boldsymbol{\xi}_i \tag{20}$$

with $\sum_{i=1}^{N} \rho_i^2 = 1$ (Hu and Blanc, 1998).

Considering for instance the gradual deformation of 2 random vectors $\boldsymbol{\xi}_1$ and $\boldsymbol{\xi}_2$, we have a single gradual deformation parameter as Equation 21.

$$\boldsymbol{\xi}^{new}(\theta) = \sum_{i=1}^{2} \rho_i \boldsymbol{\xi}_i = \boldsymbol{\xi}_1 cos(\theta) + \boldsymbol{\xi}_2 sin(\theta) \tag{21}$$

For the calibration of a stochastic model, the following iterative optimization procedure is often used (Equation 22).

$$\boldsymbol{\xi}_n(\theta) = \boldsymbol{\xi}_{n-1} \cos\theta + \boldsymbol{\zeta}_n \sin\theta \tag{22}$$

where $\boldsymbol{\xi}_{n-1}$ is the optimized parameterization vector at iteration $n-1$, and $\boldsymbol{\zeta}_n$ are a series of independent Gaussian random vectors. Then by minimizing the objective function with respect to parameter θ, we get a new parameterization vector $\boldsymbol{\xi}_n(\theta_{opt})$ that improves (or at least maintains) the calibration to the nonlinear data (Hu, 2000).

VI. The proposed workflow

BASED on the theories that are stated above, the proposed procedure for conditioning ensemble to dynamic data under realistic geologic scenarios is as follows (Figure 4):

1. Generate the initial ensemble (realization space)

   First we generate an initial ensemble. The initial ensemble should include models that are honoring geologic information and are conditioned to all available static data, that is, hard and soft data. To do this, we can choose a proper geostatistical algorithms, such as SGSIM, SISIM, DSSIM, and so on as variogram-based methods and SNESIM and FILTERSIM as multiple-point (MP) simulation methods. Generating the ensemble, we may have to consider the uncertainty in the static data. For example, if our geologic information is uncertain, we can use multiple training images for MP simulations.
2. Calculate the dissimilarity distances (distance space to metric space)

   From the initial ensemble, we calculate the dissimilarity distances and construct a distance matrix. At this step, it is important for the distances to be correlated with the dynamic data that we want to condition. If needed, we can apply multidimensional scaling to lower the dimension and get Euclidian distances, which make it possible to use RBF kernels.
3. Calculate the kernel matrix (to feature space)

   Based on the Euclidian distances, we calculate the kernel matrix. RBF kernel matrix would be easily calculated but a proper kernel should be chosen cautiously.
4. Parameterize the initial ensemble (to parameterization space)

   After obtaining the eigenvalues and eigenvectors of the kernel matrix, each member of the initial ensemble is parameterized to relatively short Gaussian random variables.

5. Optimize the parameters (in parameterization space)
   The optimization process would be done on the parameterization of the initial ensemble. Since the parameters are low-dimensional Gaussian random variables, we may apply various optimization methods, such as gradient-based methods using the sensitivity coefficients, probability perturbation method, gradual deformation method, ensemble Kalman filter, and so on. Since we already have an ensemble, EnKF would be applied effectively and provide multiple models which show the same dynamic data response. Additionally, the optimization might be accelerated by an efficient selection method through kernel PCA.
6. Solve the pre-image problems (to realization space)
   Now, the optimized parameters are converted into model state vectors. Using a proper minimization algorithm, such as a fixed-point iteration, we solve the pre-image problems for all the optimized parameters.
7. Analyze multiple models
   Finally, we obtain multiple models which satisfy all available data and geologic scenarios. We can use these multiple models in a variety of purposes. Since we generate an initial ensemble reflecting the uncertainty after conditioning to static data acquired so far, these final multiple models indicate the uncertainty (*a posteriori*) after conditioning to static and dynamic data. The multiple models may suggest which type of data should be acquired additionally, or a value of information question can be posed.

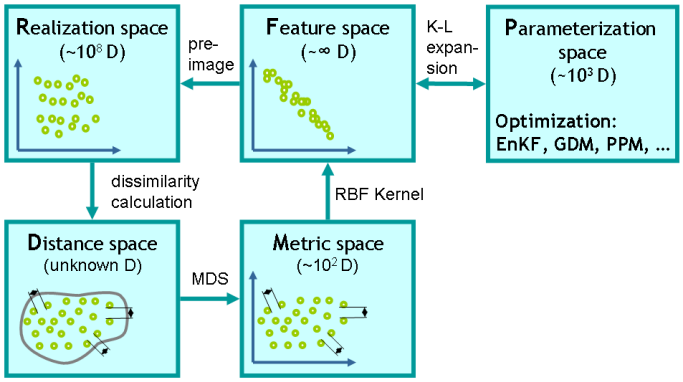
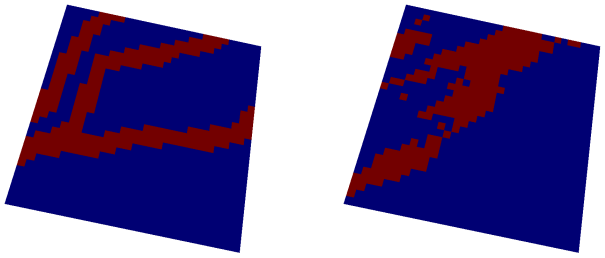


Fig. 4. The proposed workflow.

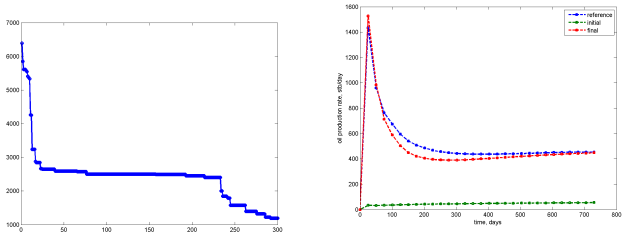
VII. Evaluations

We applied the proposed work flow to a simple case. Within 300 iterations, we could find a realization which are conditioned to the dynamic data (Figure 6). The final realization indicates similar channel locations and directions (Figure 5).



(a) reference (b) final

Fig. 5. The reference and final realizations.



(a) Decrease of the mismatch of the dynamic data with iterations.

(b) Dynamic data of the initial (green), final (red), and reference (blue) realizations.

Fig. 6. Decrease of the mismatch of the dynamic data with iterations and the dynamic data matching.

VIII. Conclusion

The objective of this research is to generate multiple models which are satisfying all available static and dynamic data. For the objective, multiple optimization methods will be combined and applied in kernel feature space based on a dissimilarity distance. The proposed method will be verified in both theoretical and experimental ways. This research has potential for applications in various areas of reservoir modeling and real-time production optimization.

Acknowledgements

The author would like to thank prof. Andrew Ng and five TA's for great lectures and helps.

References

[1] Deutsch, C.V. and Journel, A.G.: *Geostatistical Software Library and User's Guide*, Oxford University Press, NY (1998).
[2] Hu, L.Y.: *Gradual Deformaion and Iterative Calibration of Gaussian-Related Stochastic Models*, *Mathematical Geology* (2000) **32**, 1.
[3] Park, K. and Caers, J.: *History Matching in Low-Dimensional Connectivity Vector Space*, proceedings of EAGE Petroleum Geostatistics 2007 Conference, Cascais, Portugal.
[4] Schölkopf, B. and Smola, A.J.: *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*, The MIT Press, Cambridge, MA (2002).
[5] Suzuki, S. and Caers, J.: *History Matching with an Uncertain Geological Scenario*, paper SPE102154 presented at the 2006 SPE ATCE, TX.