PLOS ONE

RESEARCH ARTICLE

A Hybrid Optimization Method for Solving Bayesian Inverse Problems under Uncertainty

Kai Zhang[1], Zengfei Wang[1], Liming Zhang[1]*, Jun Yao[1], Xia Yan[2]

1 China University of Petroleum, 66 Changjiang West Road, Qingdao, Shandong, 266555, China,
2 PetroChina Coalbed Methane Company Limited, Beijing, 100028, China

* reservoirs@163.com

Citation: Zhang K, Wang Z, Zhang L, Yao J, Yan X (2015) A Hybrid Optimization Method for Solving Bayesian Inverse Problems under Uncertainty. PLoS ONE 10(8): e0132418. doi:10.1371/journal.pone.0132418

Editor: Lixiang Li, Beijing University of Posts and Telecommunications, CHINA

Received: March 17, 2015

Accepted: June 12, 2015

Published: August 7, 2015

Data Availability Statement: All relevant data are within the paper.

Funding: China Important National Science & Technology Specific Projects" under Grant 2011ZX05024-002-008, "The National Natural Science Foundation of China" under Grant 61004095 and 61104170, the Specialized Research Fund for the Doctoral Program of Higher Education of China" under Grant 20100133120003, "the Fundamental Research Funds for the Central Universities" under Grant 13CX02053A, and "Changjiang Scholars and Innovative Reserch Team in University" under Grant IRT1294, and "863 Important Project" under Grant

Abstract

In this paper, we investigate the application of a new method, the Finite Difference and Stochastic Gradient (Hybrid method), for history matching in reservoir models. History matching is one of the processes of solving an inverse problem by calibrating reservoir models to dynamic behaviour of the reservoir in which an objective function is formulated based on a Bayesian approach for optimization. The goal of history matching is to identify the minimum value of an objective function that expresses the misfit between the predicted and measured data of a reservoir. To address the optimization problem, we present a novel application using a combination of the stochastic gradient and finite difference methods for solving inverse problems. The optimization is constrained by a linear equation that contains the reservoir parameters. We reformulate the reservoir model's parameters and dynamic data by operating the objective function, the approximate gradient of which can guarantee convergence. At each iteration step, we obtain the relatively 'important' elements of the gradient, which are subsequently substituted by the values from the Finite Difference method through comparing the magnitude of the components of the stochastic gradient, which forms a new gradient, and we subsequently iterate with the new gradient. Through the application of the Hybrid method, we efficiently and accurately optimize the objective function. We present a number numerical simulations in this paper that show that the method is accurate and computationally efficient.

Introduction

Subsurface geology is always uncertain. Uncertainty assessment of geological description and reservoir production prediction is an inverse problem and is usually performed by generating a suite of plausible realizations of the reservoir model that are consistent with the available data (Fig 1). Randomized maximum likelihood (RML) was introduced by Oliver [1] and is used to build an a posteriori probability density function (PDF). Generating a realization with RML involves minimizing an objective function with an optimization method [2, 3]. In the early studies of history matching, the least square method was used for optimization. Because the objective function usually contains a large number of parameters, the descending dimension method was introduced into the optimization process to reduce the computation time.

2013AA09A215. PetroChina Coalbed Methane Company Limited provided support in the form of salary for author XY, but did not have any additional role in the study design, data collection and analysis, decision to publish, or preparation of the manuscript. The specific role of this author is articulated in the 'author contributions' section.

Competing Interests: Co-author Xia Yan is employed by PetroChina Coalbed Methane Company Limited. There are no patents, products in development or marketed products to declare. This does not alter the authors' adherence to all the PLOS ONE policies on sharing data and materials.

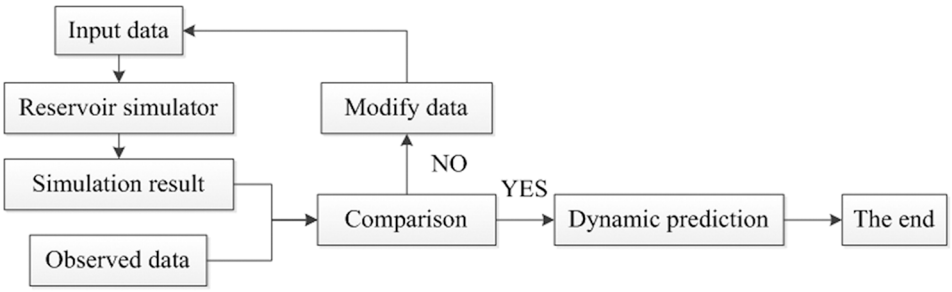

Fig 1. The process of history matching. Input data are the parameters generated based on the probability distribution. Observed data are the production data measured on site.

doi:10.1371/journal.pone.0132418.g001

There are two different types of methods for optimization in history matching: the gradient-based method and the randomized method. The gradient obtained from the gradient-based method can be calculated by either using the Jacobi or adjoint method. Due to the complexity of solving the Hessian matrix, which is required in the gradient-based method, it could take many iteration steps to obtain a small value of the objective function. The Quasi-Newton method, which does not require calculating the Hessian matrix, was used by P. Yang [4] to minimize the objective function; however, due to having a long computation time, it can only be used in one- or two-dimensional flows. The Gauss-Newton method was later applied to simulate three-phase flows and extend the scope of the history matching, even though the computation of the gradients is often more expensive than solving the flow equations. To overcome this problem, some have used fast streamline-based simulation methods for history matching [5–7].

The randomized methods have overcome some of the drawbacks of gradient-based methods because there is no need to calculate the accurate gradient; instead, one only obtains the stochastic gradient by the value of the objective function. Simulated Annealing and Evolutionary Algorithms such as genetic algorithms [8] and Evolution Strategies have been adapted in various reservoir performance optimization frameworks, Ant colony foraging optimization algorithm [9, 10]which is drived from ant colony foraging process is one of global search methods. The ant will leave a kind of hormone in the process of foraging and the subsequent ants can identify the amount of these hormones to judge whether this path is the best route. Through this algorithm we can get the optimal solution. What they have in common is that a large amount of calculation and a slow rate of convergence. For large practical optimization instances, they have a narrow applicability because the computation time is long. For this reason, the Ensemble Kalman filter method was introduced into the reservoir history matching field. The Ensemble Kalman filter (EnKF) is one of the Monte Carlo approaches for history matching [11, 12]. The EnKF method has resolved many intractable reservoir problems in recent years.

The simultaneous perturbation stochastic approximation (SPSA) algorithm, which was developed on the basis of the Kiefer-Wolfowitz stochastic approximation algorithm [13], has also attracted much attention for challenging optimization problems in which it is not easy to directly obtain a accurate gradient of the objective function for the variavles being optimized [13, 14]. SPSA is an easily implemented and highly efficient stochastic gradient that relies on an objective function. In contrast, the finite difference approach requires a lot of function measurements for which the amount is proportional to the dimension of the gradient vector. Applications of the SPSA include training neural networks, monitoring signals, et al. H. Klie [15] and Gao [16] introduced SPSA into the reservoir history matching field. Without considering the correlation of the model parameters, SPSA reaches an unsatisfactory goal. For this reason, Li and Reynolds [17] replaced the Bernoulli distribution with a Gaussian distribution of

disturbance variables and introduced a covariance matrix into SPSA; this approach has a better matching result but has a slow rate of convergence.

At present, those methods which combine many different kinds of optimization algorithms have been widely applied in the reservoir production optimization field [18, 19]. We adopt an approach by combining partial finite differences with the stochastic method based on a directional derivative to solve the inverse problem and quantify the difference frequency in order to increase the stability of this method. This method (called the Hybrid method) mainly replaces the corresponding important components of the stochastic gradient in sequence and integrates them to create a new approximation gradient for optimization. Examples of using the Bayesian approach to solve engineering problems can be found in [19–26] When quantifying the uncertainty in the model parameter identification problems, the novelty of the present work rests in the demonstration of the potential for accelerating the solving time of inverse solution. By employing the Hybrid method and the improved algorithm in the inverse solution process, the solution time can be considerably reduced.

Problem Formulation

Objective function

The Randomized Maximum Likelihood Estimation (RMLE) is one of the methods that can be performed to find the value of x by Probability theory, where the overall probability function $p(X,\theta)$ is known, and θ is an unknown parameter of the set Φ.

We filter N_e random samples, which are defined as $X_1, X_2, \cdots, X_{N_e}$, from the population X, and obtain the corresponding actual observed values $x_1, x_2, \cdots, x_{N_e}$. Then, the probability that $X_1 = x_1, X_2 = x_2, \cdots, X_{N_e} = x_{N_e}$ occurs simultaneously is

$$p(\theta) = \prod_{i=1}^{N_e} p(X_i = x_i) \tag{1}$$

Due to Probability theory, the probability function $p(\theta)$ can obtain the maximum value.

In engineering problems, parameters (such as porosity, permeability) usually have a type of probability distribution. From experience, we know that these parameters usually obey a Gaussian distribution in an actual application.

The Probability Density Function is

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{2}$$

Where $x \sim N(\mu,\sigma^2)$, μ is the mean of x, and σ^2 is the variance.

Eq 2 can also be expressed as

$$p(x) \propto \exp\left[-\frac{1}{2}(x - x_{av})^T C_X^{-1}(x - x_{av})\right] x \in N(x_{av}, C_X) \tag{3}$$

Where x denotes the N_e dimensional vector, which includes uncertain parameters; and x_{av} denotes the prior model parameters; C_X is the $N_e \times N_m$ covariance matrix of the measurement errors that are generated by the statistical methods, and $C_X \in R^{N_e \times N_m}$.

Based on statistical theory, the relationship between the observed production data and the reservoir model parameters in the oilfield development process is

$$d_{\text{obs}} = g(x) + \varepsilon_r \quad \varepsilon_r \in N(0, C_D) \tag{4}$$

Where d_{obs} denotes the N_{d} dimensional vector that includes the actual observed data; g represents the numerical simulator of the reservoir system; ε_r is the measurement error; and C_D is the covariance matrix.

Maximum likelihood estimation can be performed to find the parameters that maximize the probability that the calculated value is the same as the observed value; thus, the conditional probability distribution function of d_{obs} under the condition of x in the reservoir model is

$$p(d_{\text{obs}}|x) \propto \exp\left[-\frac{1}{2}(d_{\text{obs}} - g(x))^T C_D^{-1}(d_{\text{obs}} - g(x))\right] \tag{5}$$

Gradient-based optimization and regularization techniques are mainly used toward this goal, generally by maximizing the logarithm of the likelihood and thus, resulting in a single estimated value for a local maximum. A Bayesian approach is adopted herein, which leads not only to a point estimate of the parameters of interest but also to a probability distribution. By attributing a prior distribution to the model parameters and applying the Bayesian theorem, we obtain their posterior distribution, which enables one to quantify the uncertainty concerning the parameter values.

The Bayesian theorem provides the solution of the inverse problem

$$\begin{aligned} p(x|d_{\text{obs}}) &\propto p(d_{\text{obs}}|x)p(x) \\ &\propto \exp\left[-\frac{1}{2}(d_{\text{obs}} - g(x))^T C_D^{-1}(d_{\text{obs}} - g(x)) - \frac{1}{2}(x - x_{av})^T C_M^{-1}(x - x_{av})\right] \end{aligned} \tag{6}$$

where $p(x)$ is the prior distribution. To obtain the maximum value of the conditional probability of x under d_{obs}, the objective function is

$$P(x) = \frac{1}{2}(x - x_{av})^T C_X^{-1}(x - x_{av}) + \frac{1}{2}(d_{\text{obs}} - g(x))^T C_D^{-1}(d_{\text{obs}} - g(x)) \tag{7}$$

When the conditional probability is the maximum value, the corresponding objective function becomes the minimum value. Therefore, reservoir history matching problems can be translated into resolving minimum value problems of the objective function $P(x)$. When the objective function is the minimum, the corresponding variables x are close to the actual parameters.

Singular value decomposition

For actual reservoir history matching problems, the dimension N_m of the reservoir parameters that require inversion is typically tens of thousands. Therefore, it is extremely difficult to optimize the objective function, and the computational cost can become tremendously expensive due to the large size of the linear systems, especially the large amount of computation that is required to calculate the inverse matrix $C_X^{-1}(x - x_{av})$. Therefore, this paper utilizes the singular value decomposition method. Through this method, C_X^{-1} and $C_X^{-1}(x - x_{av})$ can be transformed into low-dimensional matrixes, which avoids complex computation.

According to the definition of covariance, the covariance matrix of the initial model can be approximately calculated as

$$C_X = \frac{1}{N_e - 1}\sum_{i=1}^{N_e}(x_i - x_{av})(x_i - x_{av})^T = \frac{1}{N_e - 1}\delta X \delta X^T \tag{8}$$

where $\delta X = [\delta x_1, \delta x_2, \ldots, \delta x_i, \ldots \delta x_{N_e}]_{N_m \times N_e}$, and the jth column vector is $(x - x_{av})$. With

Singular Value Decomposition,

$$\delta X = U\Lambda V^T \tag{9}$$

Where U and V are the singular vectors of δX, and Λ is the singular value of δX. U is composed of orthogonal unit characteristic vectors of C_X. $\Lambda^T\Lambda$ is composed of characteristic values.

Because $V^T V = I_{N_e}$,

$$C_X \approx \frac{1}{N_e - 1} U\Lambda\Lambda^T U^T \tag{10}$$

Assuming that there are N_S non-zero singular values in Λ, then

$$\Lambda = \begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \cdots & \\ & & & \sigma_{\min(N_m,N_e)} \end{bmatrix}_{N_m \times N_e} = \begin{bmatrix} \sigma_1 & & & & & & \\ & \sigma_2 & & & & & \\ & & \cdots & & & & \\ & & & \sigma_{Ns} & & & \\ & & & & 0 & & \\ & & & & & \cdots & \\ & & & & & & 0 \\ & & & & & & \\ & & & & & & \end{bmatrix}_{N_m \times N_e} = \begin{bmatrix} \Delta & 0 \\ 0 & 0 \end{bmatrix} \tag{11}$$

Usually, N_m is far larger than N_e. Therefore, N_s is less than or equal to N_e. If we only consider the singular value vectors of those models, then

$$C_X \approx \frac{1}{N_e - 1} U_s \Lambda_s^2 U_s^T \tag{12}$$

The approximate inverse matrix of C_X is

$$C_X^{-1} = (N_e - 1) U_s \Lambda_s^{-2} U_s^T \tag{13}$$

By substituting C_X of the objective function with C_X^{-1}, the N_m dimensional optimization problem about x is reduced to N_s dimensions, which avoids the process of solving the inverse of C_X (its dimension can reach up to tens of millions). This method has greatly simplified the difficulty of calculation and improved the computational efficiency. The final function becomes

$$P(x) = \frac{1}{2}(x - x_{av})^T (N_e - 1) U_s \Lambda_s^{-2} U_s^T (x - x_{av}) + \frac{1}{2}(d_{\text{obs}} - g(x))^T C_D^{-1} (d_{\text{obs}} - g(x)) \tag{14}$$

Solution

Linear sequential solutions

This paper uses the linear search method for optimization. Here, the linear search consists of two key elements: one is the search step size, which ensures the convergence of the search, and the other is the search direction, which determines the rate of convergence. In actual application, the normalization method of the search direction is usually used for iterative calculation. Now, we introduce the optimization method of the linear search.

In the process of optimizing the objective function $P(x)$, set the initial variable vector x^0 to zero and calculate the initial objective function value $P(x^0)$ first. At the kth(k = 1,2,3,⋯,$K_{\max}$) iteration step, do the following: First, set the initial search step $\lambda^k = 1$, and calculate the stochastic gradient $\hat{g}_k(x)$ as the search direction D^k through the optimization algorithm; then, normalize D^k;

Second, update the variables vector x^k by $x^k = x^{k-1} + \lambda_k \frac{D^k}{\|D^k\|_\infty}$;

Third, calculate the objective function value $P(x^k)$ and compare $P(x^k)$ with $P(x^{k-1})$. Then, determine whether $P(x^k)$ satisfies $0 \le \frac{|P(x^k)-P(x^{k-1})|}{\max\{P(x^k),1\}} \le \varepsilon_p$. If the result is YES, then exit the iteration; if the result is NO, then keep the values of x^k and $P(x^k)$, and then, set $k = k + 1$ and continue the next iteration step. If $P(x^k) > P(x^{k-1})$, then reduce λ_k by half and continue to calculate the objective function value $P(x^k)$ until it reaches the maximum number of times m that λ_k can be halved; if beyond, exit and continue another iteration.

Algorithm I. Linear optimization algorithm.
1. Generate the initial value x^0.
2. For $k = 1,2,3,\cdots$ until convergence
(a) Calculate the stochastic gradient $\hat{g}_k(x)$ as the search direction D^k through the optimization algorithm and set the initial search step to $\lambda^k = 1$.
(b) For $i = 1,2,\cdots,m$, calculate

$$x^k = x^{k-1} + \lambda_k \frac{D^k}{\|D^k\|_\infty}$$

Calculate the objective function value $P(x^k)$,
If $P(x^k) > P(x^{k-1})$, then check if

$$0 \le \frac{|P(x^k) - P(x^{k-1})|}{\max\{P(x^k), 1\}} \le \varepsilon_p$$

If the result is Yes, then exit the iteration.
If the result is No, then

$$\lambda_k = \frac{\lambda_k}{2}$$

End for
End for

In the above algorithm, ε_p is approximately 10^{-4}, and $\|\bullet\|_\infty$ is the infinite norm value.

Because the stochastic gradient $\hat{g}_k(x)$ in the optimization process is always calculated by the stochastic algorithm, we introduce the stochastic algorithm first.

Stochastic algorithm based on the directional derivative

The basic principle of this algorithm is to disturb the argument $x = [x_1, x_2, \cdots, x_{N_e}]^T$ and obtain a new variable vector $x' = [x_1', x_2', \cdots, x_{N_e}']^T$, which is

$$\begin{bmatrix} x_1' \\ x_2' \\ \vdots \\ x_{N_e}' \end{bmatrix} = \begin{bmatrix} x_1 + \alpha\Delta x_1 \\ x_2 + \alpha\Delta x_2 \\ \vdots \\ x_{N_e} + \alpha\Delta x_{N_e} \end{bmatrix} \tag{15}$$

Where α is the disturbance step, and $\Delta x = [\Delta x_1, \Delta x_2, \cdots, \Delta x_{N_e}]^T$ is the disturbance variables vector, with the element Δx_i(i = 1,2,⋯,N_e) in accordance with the Bernoulli distribution of 1 or -1; therefore, Δx_i^{-1} is equal to Δx_i. By calculating the corresponding objective function values and taking the difference of each increment, we can obtain the stochastic gradient $\hat{g}(x)$.

In each iteration step, $\hat{g}(x)$ of $P(x)$ at x can be expressed as

$$\hat{g}(x) = \begin{bmatrix} \frac{P(x_1') - P(x_1)}{\alpha \Delta x_1} \\ \frac{P(x_2') - P(x_2)}{\alpha \Delta x_2} \\ \vdots \\ \frac{P(x_{N_e}') - P(x_{N_e})}{\alpha \Delta x_{N_e}} \end{bmatrix} \tag{16}$$

The stochastic gradient is

$$\hat{g}(x) = \frac{P(x + \alpha \Delta x) - P(x)}{\alpha} \times \Delta x^{-1} \tag{17}$$

The directional derivative reflects the rate of change of the objective function value in a specific direction. The formulation is

$$\begin{aligned} \frac{\partial P}{\partial l} &= \frac{P(x + \alpha \Delta x) - P(x)}{\sqrt{(\alpha \Delta x_1)^2 + (\alpha \Delta x_2)^2 + \cdots + (\alpha \Delta x_{N_e})^2}} = \frac{P(x + \alpha \Delta x) - P(x)}{\alpha \Delta x} \times \frac{\Delta x}{\|\Delta x\|_2} \\ &= \hat{g}_1 \cdot \cos\varphi_1 + \hat{g}_2 \cdot \cos\varphi_2 + \cdots \hat{g}_{N_e} \cdot \cos\varphi_{N_e} = (\cos\langle\varphi\rangle)^T \cdot \hat{g} \end{aligned} \tag{18}$$

where $\frac{\partial P}{\partial l}$ is the directional derivative of the objective function in the $\vec{l}$ direction; and $\cos\theta_i$ is the cosine value of $\langle e_i, \vec{l} \rangle$.

The directional derivative of x by using the true gradient can be expressed as

$$\frac{\partial P}{\partial l} = \hat{g}_1 \cdot \cos\varphi_1 + \hat{g}_2 \cdot \cos\varphi_2 + \cdots \hat{g}_{N_e} \cdot \cos\varphi_{N_e} = (\hat{g})^T \cos\langle\varphi\rangle \tag{19}$$

From Eqs 18 and 19, we can obtain

$$\left(\frac{\partial P}{\partial l}\right)^2 = (\hat{g})^T \cdot \cos\langle\varphi\rangle \cdot (\cos\langle\varphi\rangle)^T \cdot \hat{g} \geq 0 \tag{20}$$

Set $g' = \cos\langle\varphi\rangle \cdot (\cos\langle\varphi\rangle)^T \cdot \hat{g}$. From Eq 20, we can see that g' is the increasing direction, which ensures the convergence of this algorithm.

We usually obtain the average of the stochastic gradient through several disturbances in pursuit of improving the accuracy of the gradient, and then, we resolve the problem by the above optimization method (Algorithm I) after obtaining a stochastic gradient.

$$\bar{g}(x) = \frac{\sum_{k=1}^{N} g_k'}{N} \tag{21}$$

Algorithm II. Stochastic algorithm based on directional derivative.

1. For $k = 1, 2, \cdots N$, generate random Δx^k
   (a) Calculate $x^k = x + \alpha \Delta x^k$ and $p(x^k)$,
   (b) Then, compute
   $$\hat{g}_k = \frac{P(x^k) - P(x)}{\alpha \Delta x^k}$$

(c) Calculate the cosine value $\cos\varphi$.
(d) Set $g_k' = \cos\langle\varphi\rangle \cdot (\cos\langle\varphi\rangle)^T \cdot \hat{g}_k$.
End for
(2) Calculate $\bar{g} = \sum_{k=1}^{N} g_k'/N$

Because the gradient obtained from Algorithm II is approximate, the stochastic gradient with a perturbation and directional derivative has a high uncertainty. Although Algorithm II has improved some of the shortcomings that other algorithms have, it also has many deficiencies, such as a large number of iterations and slow convergence of the objective function. To avoid these shortcomings, we propose a new algorithm by modifying Algorithm II.

Principle of the hybrid algorithm

The main idea of the Hybrid algorithm is to first calculate the stochastic gradient of the objective function by using Algorithm II and, then, to replace the components with the largest (in magnitude) stochastic gradient with approximate values of the gradient from the finite difference in a proper sequence until the direction of the modified gradient is nearly the same as the unknown real gradient direction.

According to Taylor's formula, $P(x+\alpha\Delta x) - P(x)$ can be expanded as

$$\begin{aligned} P(x+\alpha\Delta x) - P(x) &\approx \alpha\Delta x \frac{\partial P(x)}{\partial x} \\ &= \alpha \sum_{k=1}^{N_e} \Delta x_k \frac{\partial P(x)}{\partial x_k} \\ &= \alpha(\Delta x)^T \nabla P(x) \end{aligned} \tag{22}$$

Then,

$$\begin{aligned} \frac{\partial P(x)}{\partial x_k} &= \frac{P(x+\alpha\Delta x_k e_k) - P(x)}{\alpha\Delta x_k e_k} \\ &\approx \frac{P(x+\alpha\Delta x_k e_k) - P(x)}{\alpha\Delta x_k} \end{aligned} \tag{23}$$

where e_k is a unit vector in the kth direction. Substituting Eq 23 into Eq 22, we obtain

$$\begin{aligned} \Delta P(x) &= P(x+\alpha\Delta x) - P(x) \\ &\approx \sum_{k=1}^{N_e} [P(x+\alpha\Delta x_k e_k) - P(x)] \\ &= \sum_{k=1}^{N_e} \Delta P_k(x) \end{aligned} \tag{24}$$

Here, $\Delta P_k(x)$ is the increment of the objective function value in the kth direction.

The conclusion is that the increment of the objective function value with the simultaneous perturbation of all of the components is approximately equal to the sum of each function value increment with the separate disturbance of each component of the vector x.

Hybrid algorithm

A simple equation is introduced to vertify the changement og some components has a great impact on the result of function.

$$y(x) = (x-a)(x-a)^T \tag{25}$$

Where a = [1,1,1,1,1]. If x_0 = [2,3,2.5,5,1], then we will obtain $y(x_0)$ = 23.25. Set Δx = [0.5,0.5,0.5,0.5,0.5] and calculate the function value of $x_0 + \Delta x$ and the increments of the function value with each component.

From Table 1, we can realize that $\Delta y(x)$ takes the maximum value when the 4th component is changed, which has the greatest impact on the result. Here, $x_0^{(4)}$ is the largest component of all of the components, and thus, we regard $x_0^{(4)}$ as the 'important' element. We could find out the 'relatively important' elements of the stochastic gradient by relying on this standard.

The index collection of all of the components in the stochastic gradient can be denoted as U = {1,2,⋯,N_e}; I_u and I_d are its two subsets, which satisfy $I_u \cup I_d = U$ and $I_u \cap I_d = \phi$.I_u denotes the relatively 'important' components of the stochastic gradient, and I_d denotes the 'unimportant' components.

The hybrid algorithm is used to calculate the elements of I_u with the finite difference method and replace the elements of I_d by the stochastic gradient.

Determining the search direction is a process of continuous iteration; thus, we set the initial values $I_u^0 = \phi$ and $I_d^0 = U$, $N_u^0 = 0$ and $N_d^0 = N_e$. N_u^0 and N_d^0 denote the number of elements in I_u^0 and I_d^0, respectively, and the superscript denotes the iteration step.

From Eq 22, we can obtain

$$\Delta P_u^{(j)}(x) = \alpha \sum_{k \in I_u} \Delta x_k \frac{\partial P(x)}{\partial x_k} \tag{26}$$

$$\Delta P_d^{(j)}(x) = \alpha \sum_{k \in I_d} \Delta x_k \frac{\partial P(x)}{\partial x_k} \tag{27}$$

where j is the number of samples $\Delta P^j(x)$, j = 1,2,⋯$N_{\max}$.

When m = 0, $\Delta P_u^{(j),0}(x)$ is equal to zero, and $\Delta P_d^{(j),0}(x) = \Delta P^{(j)}(x)$. At the mth iteration step,

$$\Delta P^{(j)}(x) = \Delta P_u^{(j),m}(x) + \Delta P_d^{(j),m}(x) \tag{28}$$

Now, we consider the mth iteration step.

Table 1. The increment of the function value at each component of *x*

$\Delta y(x_1)$	$\Delta y(x_1^{(1)})$	$\Delta y(x_1^{(2)})$	$\Delta y(x_1^{(3)})$	$\Delta y(x_1^{(4)})$	$\Delta y(x_1^{(5)})$
9.75	1.25	2.25	1.75	4.25	0.25

Here, $x_1^{(i)}$ is the increment of the ith(i = 1,2,3,4,5) component of x_0.

doi:10.1371/journal.pone.0132418.t001

First, generate a stochastic gradient by using Algorithm II:

$$\begin{aligned}\hat{g}(x^k) &= \frac{1}{N_{\max}}\sum_{j=1}^{N_{\max}}\left[\frac{P(x^k+\alpha e^{(j)})-P(x^k)}{\alpha}\right]e^{(j)}\cos(\varphi)\\ &= \frac{1}{N_{\max}}\sum_{j=1}^{N_{\max}}\left[\frac{\Delta P^{(j)}(x^k)}{\alpha}\right]e^{(j)}\cos(\varphi)\end{aligned} \tag{29}$$

The lth component of the gradient $\hat{g}(x^k)$ is

$$\hat{g}_l(x^k) = \frac{1}{N_{\max}}\sum_{j=1}^{N_{\max}}\frac{\Delta P^{(j)}(x^k)}{\alpha}e_l^{(j)}\cos(\varphi) \tag{30}$$

Replace $\Delta P^{(j)}$ with $\Delta P^{(j),m}$; then,

$$\hat{g}_l(x^k) = \frac{1}{N_{\max}}\sum_{j=1}^{N_{\max}}\frac{\Delta P^{(j),m}(x^k)}{\alpha}e_l^{(j)}cos(\varphi), l\in I_d^m \tag{31}$$

Therefore, we can use Eq 13 to calculate the lth component of the stochastic gradient in I_d^m.

Second, calculate each component of the stochastic gradient $\hat{g}(x^k)$ and sort them according to their absolute values. Take the first N_{key} elements as the 'important' elements, and record their subscripts in I_u^{m+1}; then, record the subscripts of the remaining elements in I_d^{m+1}, and update I_u^{m+1} and I_d^{m+1}.

Third, recalculate the ith($i = 1,2,\cdots,N_{key}$) 'important' element in I_u^{m+1} by the finite difference approach:

$$\frac{\partial P(x^k)}{\partial x_i} = \frac{P(x^k+\alpha e_i)-P(x^k)}{\alpha}, i\in I_u^{m+1} \tag{32}$$

Then, obtain the realistic gradient $\nabla P_u^{m+1}(x^k)$, where the ith element ($i \in I_u^{m+1}$) can be obtained by finite differences, and record the other elements as zero.

Calculate $\Delta P_u^{(j),m+1}$ and $\Delta P_d^{(j),m+1}$

$$\Delta P_u^{(j),m+1}(x^k) = \sum_{i\in I_u^{(j)}}\alpha e_l^{(j)}\frac{\partial P(x^k)}{\partial(x_i)} \tag{33}$$

$$\Delta P_d^{(j),m+1}(x^k) = \Delta P^{(j),m+1}(x^k) - \Delta P_u^{(j),m+1}(x^k) \tag{34}$$

Where $j = 1,2,\cdots N_{\max}$.

Fourth, calculate the angle θ between the approximate gradient and the unknown true gradient, as follows:

$$\cos(\theta) = \sqrt{1-\frac{E[(\Delta P_d^m(x^k))^2]}{E[(\Delta P(x^k))^2]}} \tag{35}$$

If $\cos(\theta) \geq \varepsilon$, then set $D^k = \nabla P_u^m$ and exit the Loop.

If $\cos(\theta) < \varepsilon$, then determine whether the 'important' elements in I_u^m obtained by the stochastic gradient correspond to the important elements of the true gradient. Check each ith

element in the collection I_u^m, as follows:

$$\left(\frac{P(x^k+\alpha e_i)-P(x^k)}{\alpha}\right)^2 \geq \frac{1}{N_{\max}}\frac{E[(\Delta P(x^k))^2]}{\alpha^2} \tag{36}$$

If there exists one component of the stochastic gradient that is 'important', but the value recalculated by finite differences does not satisfy Eq 36, then mark this identification of the 'important' component as *failure* and add the number of *failure* times to 1. The maximum number of *failure* times allowed in the mth iteration step is $N_{f,\max}$.

If the number of *failure* times N_i exceeds $N_{f,\max}$, then we should reselect new samples, recalculate the significant elements and determine the 'important' elements. If $N_i < N_{f,\max}$, then reselect N_{key}'important' elements from I_d^m by finite differences, and set $m = m+1$; continue this iteration until $\cos(\theta) \geq \varepsilon$.

The standard of judgment

As mentioned earlier, P_u is the approximation gradient that is in almost the same direction as the true gradient. The cosine of the angle between these two gradients is given by

$$\cos(\theta) = \frac{(\nabla P_u)^T \nabla P}{\|\nabla P_u\| \cdot \|\nabla P\|} \tag{37}$$

where ∇P_u is the approximation gradient that is obtained by using finite differences, and ∇P is the unknown true gradient.

Set $\nabla P_{u,l}(x)$ to be the lth component of $\nabla P(x)$. When the element l satisfies $l \in l_u$ in $\nabla P(x)$, then calculate $\nabla P(x)$ by finite differences; if $l \in l_d$, then set it to zero.

$$\nabla P_{u,l}(x) = \begin{cases} \dfrac{P(x^k+\alpha e_l)-P(x^k)}{\alpha}, l \in I_u \\ 0, l \in I_d \end{cases} \tag{38}$$

Then,

$$\begin{aligned} (\nabla P_u)^T \nabla P &= \sum \nabla P_{u,l} \frac{\partial P(x^k)}{\partial x_l} \\ &\approx \sum \left[\frac{P(x^k+\alpha e_l)-P(x^k)}{\alpha}\right]^2 \\ &\approx \sum [\nabla P_{u,l}(x^k)]^2 \\ &\approx \|\nabla P_u\|^2 \end{aligned} \tag{39}$$

Replacing $(\nabla P_u)^{\mathrm{T}} \nabla P$ by $\|\nabla P_u\|^2$ in Eq 37,

$$\cos(\theta) \approx \frac{\|\nabla P_u\|^2}{\|\nabla P_u\|\|\nabla P\|} = \frac{\|\nabla P_u(x^k)\|}{\|\nabla P(x^k)\|} \tag{40}$$

with

$$\Delta P(x^k) = \alpha Z^T \nabla P(x^k) = \alpha \nabla P(x^k)^T Z \tag{41}$$

Because Z obeys the Gaussian distribution, therefore

$$E(ZZ^T) = I \tag{42}$$

Then,

$$\begin{aligned} E[(\Delta P(x^k))^2] &= \alpha^2 \nabla P(x^k)^T \nabla P(x^k) \\ &= \alpha^2 \|\nabla P(x^k)\|^2 \end{aligned} \tag{43}$$

with

$$\|\nabla P(x^k)\|^2 = \|\nabla P_u(x^k)\|^2 + \|\nabla P_d(x^k)\|^2 \tag{44}$$

and

$$E[(\Delta P_d)^2] = \alpha^2 \|\nabla P_d(x^k)\|^2 \tag{45}$$

We can obtain

$$\begin{aligned} \cos(\theta) &= \sqrt{\frac{\|\nabla P_u(x^k)\|^2}{\|\nabla P(x^k)\|^2}} \\ &= \sqrt{\frac{E[(\Delta P(x^k))^2] - E[(\Delta P_d(x^k))^2]}{E[(\Delta P(x^k))^2]}} \\ &= \sqrt{1 - \frac{E[(\Delta P_d(x^k))^2]}{E[(\Delta P(x^k))^2]}} \end{aligned} \tag{46}$$

The angle between the approximation gradient and the true gradient can be calculated by Eq 46.

According to Pradlwarter (2007)[27], to determine whether the lth component of the gradient in I_u is 'important', we can calculate the square value of this component and compare it with the average of all of the components of $\nabla P(x^k)$, as follows:

$$\left(\frac{\partial P(x^k)}{\partial x_l^k}\right)^2 \geq \frac{\|\nabla P(x^k)\|^2}{N_{\max}} \tag{47}$$

If the result is 'YES', then mark it as 'important'. Conversely, if the result is 'NO', then consider it to be unimportant. Because of the unknown true gradient $\nabla P(x^k)$, according to Eq 43, Eq 47 can be expressed as

$$\left(\frac{P(x^k + \alpha e_l) - P(x^k)}{\alpha}\right)^2 \geq \frac{1}{N_{\max}} \frac{E[(\Delta P(x^k))^2]}{\alpha^2} \tag{48}$$

With the above equation, we can determine whether the lth element is 'important' or not.

The verification and improvement

Function I. The first mathematical function is to illustrate the reliability of the Hybrid algorithm. The objective is to minimize the following function:

$$P(x) = \sum_{i=1}^{N} 10x_i^2 \tag{49}$$

Table 2. Behavior of the Hybrid algorithm.

Iteration Index s	Component Index *i*	Algorithmm II	Hybrid algorithm	Flag	Failure time	cos(*θ*)
1	71	-1613.82	81	Failure		
	128	-1496.35	65	Failure		
	55	1223.8	821	Success		
	183	-756.242	185	Success		
	85	687.782	1880	Success	2	0.59
2	10	594.311	2200	Success		
	25	734.968	520	Success		
	116	-660.272	81	Failure		
	70	284.032	1040	Success		
	45	703.37	560	Success	1	0.85

doi:10.1371/journal.pone.0132418.t002

Where $x = [x_1,x_2,\cdots x_N]^T$, and the initial guess is $x_i = 2,i = 1,\cdots,N$. We set the number of variables to be $N = 100$. The minimum of Eq 49 is 0, which occurs when $x = [0,0,\cdots 0]^T$.

In Algorithm II, set the initial search step to $\lambda_0 = 1$, regard the average of the stochastic gradient calculated with 5 stochastic perturbations as the search direction, and set the disturbance variable α to 0.015. Record the objective function values of each iteration step and the cosine values.

In the Hybrid algorithm: (1) Obtain the stochastic gradient by using Algorithm II, and then, select 5 'important' components of the stochastic gradient, which constitutes a new approximate gradient to iterate; (2) The angle between the approximate gradient and the true gradient should satisfy $\cos(\theta) \geq 0.8$. If not, then recalculate and reselect the 'important' elements; (3) The maximum number of *failure* allowed is 2; (4) The maximum number of simulations is 25; if this number is exceeded, then recalculate the stochastic gradient. Calculate the objective function value for the optimization in both Algorithm II and the Hybrid algorithm, and compare the rate of convergence and $\cos(\theta)$ for the two methods.

Table 2 shows the behavior and results obtained from Hybrid algorithm for the first iteration $m = 1$. The first column refers to the iteration index s during the calculation of the approximate gradient ∇P_u, and two iterations are used to obtain ∇P_u until satisfies $\cos(\theta) \geq 0.8$. Column 2 to Column 5 show that five components of the gradient from Algorithm II are recomputed by using finite difference at each iteration, meanwhile, the component index and the specific gradient values are shown in column 2, column3 and column 4, respectively. Because the gradient is particular stochastic, the chosen components of the gradient by Algorithm II may not necessarily refer to the actual "important components". As shown at iteration s = 1, the 71th component value of the gradient by Algorithm II is as large as -1613.82 while the recomputed approximate gradient is a much smaller 81, notice that in Eq 47, the 71th component turns out to be *failure*. When the *failure* time reaches the maximum allowable number 2, it suggests that the approximate gradient is not accurate enough, then continue another iteration until $\cos(\theta) \geq 0.8$. During the two iterations, components 71,128,55,183,85,10,25,116,70 and 45 are recomputed by finite difference and the rest components of are set equal to zero.

Fig 2 shows that the cosine value of θ obtained by Algorithm II is approximately 0.3, which illustrates that the angle between the approximate gradient and the true gradient is approximately 73 degrees; however, the cosine value of θ obtained by the Hybrid algorithm is larger than 0.8 (in other words, the angle between the approximate gradient and the true gradient does not more than 35 degrees). The result has demonstrated that the direction of the gradient

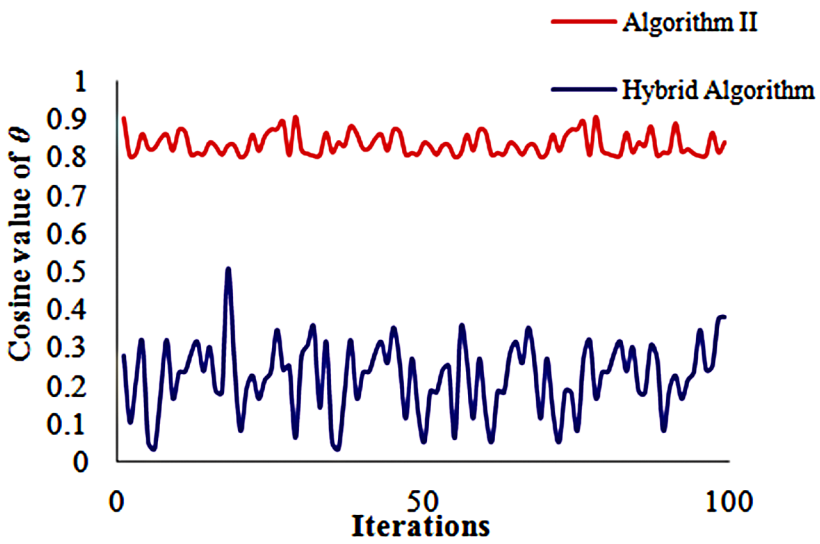

Fig 2. Accuracy of cos(θ) of both algorithms. The blue line represents the cosine value calculated by Hybrid algorithm and the red line represents the cosine value calculated by Algorithm II.

doi:10.1371/journal.pone.0132418.g002

from the Hybrid algorithm is closer to the true gradient direction than it from Algorithm II, which illustrates that the Hybrid algorithm has a higher accuracy than Algorithm IIin the search direction at each iteration step.

Fig 3(A) shows that the objective function value will decrease in both Algorithm II and the Hybrid algorithm. This finding means that the objective function also converges with the Hybrid algorithm. From the above Fig, we can find that the Hybrid algorithm behaves better than Algorithm II and has a higher rate of convergence. At approximately the 400th computation time in the Hybrid algorithm, the objective function value is close to the final value, while Algorithm II requires probably over 800 computation times to reach the same value. Thus, the Hybrid algorithm reaches the same result of convergence as Algorithm II with less simulation runs.

Fig 3(B) shows that to achieve the same value of the objective function, Algorithm III requires 20 iteration steps, while Algorithm II requires more than 100 iteration steps.

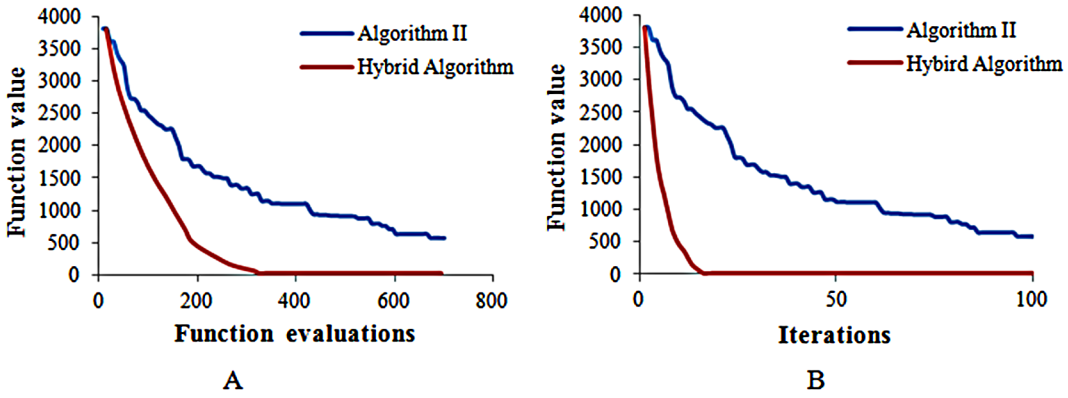

Fig 3. The convergence of the objective function. A:The objective function value versus function evaluations. B: The objective function value versus function iterations.

doi:10.1371/journal.pone.0132418.g003

Function II. The Rosenbrock function is a nonconvex function that is often used to test the performance of an optimization algorithm. Hence, the objective is to minimize the Rosenbrock function for the N-variable case, which is given by

$$P(x) = \sum_{i=1}^{N/2} \left[(x_{2i} - x_{2i-1}^2)^2 + (1 - x_{2i-1})^2 \right] \tag{50}$$

Where $x = [x_1, x_2, \cdots x_N]^{\mathrm{T}}$, and the initial guess is $x_i = 2, i = 1, \cdots, N$ We consider four cases, in which the number of variables is $N = 100,200,400,800$, and we compare the optimization results by using the Hybrid algorithm. In each step, we require the accuracy in the direction of the estimated gradient to satisfy $\cos(\theta) \geq 0.8$.

Fig 4(A) shows that the objective function can obtain a minimum value with the Hybrid algorithm in different cases, and the function evaluations increase with the increase in the number of variables. We also could attain the same property from Fig 4(B). The result has verified that this algorithm is acceptable with a different number of variables.

Next, we consider the four cases with respect to both Eqs 49 and 50, to optimize and compare the cosine values and finite difference frequency (Table 3). From Table 3, we can realize that the estimated gradient by the Hybrid algorithm satisfies $\cos(\theta) \geq 0.8$ with different numbers of variables; the finite difference (F-D) times are not the same in different steps, and the average number of finite differences increases with the increase in the number of variables.

From Table 3, we can realize that the finite difference frequency can be excessive, which would reduce the convergence in a certain iteration step, while in another iteration step, the finite difference frequency could be very small. Because of this uncertainty, we introduce the average number of finite differences in each step instead. Now, we discuss the relationship between the differential rate and the number of variables.

Fig 5 illustrates the differential rate (the ratio of the difference frequency and the number of variables in each step) in different cases, which we have discussed. Through this Fig, we can obtain that in most cases, the differential rate is less than 0.15, which means that we can obtain a much more accurate gradient by finite differences for 15 times per 100 variables. Thus, we take 15% of the number of all the variables for finite differences to replace the judgment of the cosine value.

Because we use quantitative finite differences to replace the judgment of the cosine value in every step, the Hybrid algorithm can become the following:

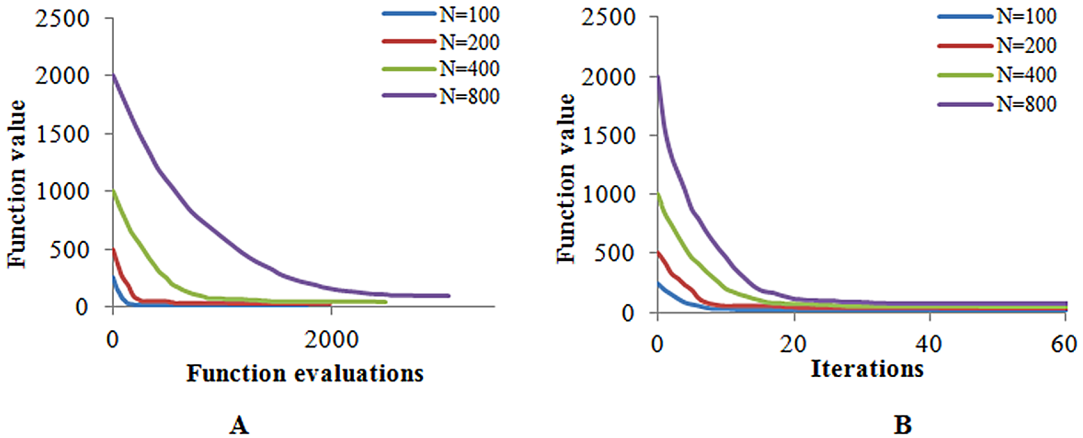

Fig 4. The optimization results in different cases. Each line represents the convergence of the objection function in different numbers of the variables.

doi:10.1371/journal.pone.0132418.g004

Table 3. The contradistinction results in different cases.

Equation	Example Index *i*	The number of variables	The average cosine value	Max time (F-D)	Min time (F-D)	Average time ineach step (F-D)
49	1	100	0.85602	35	5	14
	2	200	0.82892	50	10	27
	3	400	0.81852	105	15	51
	4	800	0.81554	180	25	94
50	5	100	0.83359	45	5	17
	6	200	0.82296	55	10	26
	7	400	0.81859	110	15	49
	8	800	0.81549	220	20	98

doi:10.1371/journal.pone.0132418.t003

Algorithm III. The improvement of the Hybrid algorithm.
1. Initialization: $I_u = \phi$ and $I_d = U$, $N_u = 0$ and $N_d = N_e$.
2. For $j = 1, 2, \cdots N_{max}$, generate N_{max} samples.
Calculate $\hat{g}(k)$ by using Algorithm II.
3. Compute each component of $\hat{g}(k)$ and sort them, and then, filter the front $0.15N_e$ elements into I_u and put the remaining elements into I_d.
4. Replace the elements of I_u with the elements by finite difference and set the other elements to zero.
5. Generate a new gradient to replace $\hat{g}(k)$.

According to the above analysis, we can realize the following: (1) the approximate gradient obtained by introducing finite differences into Algorithm II has better convergence than that generated only by Algorithm II; (2) With fewer iterations and less computation time, the Hybrid algorithm has a better optimization result than Algorithm II; and (3) the quantitative finite difference frequency can improve the stability of the Hybrid algorithm. Those

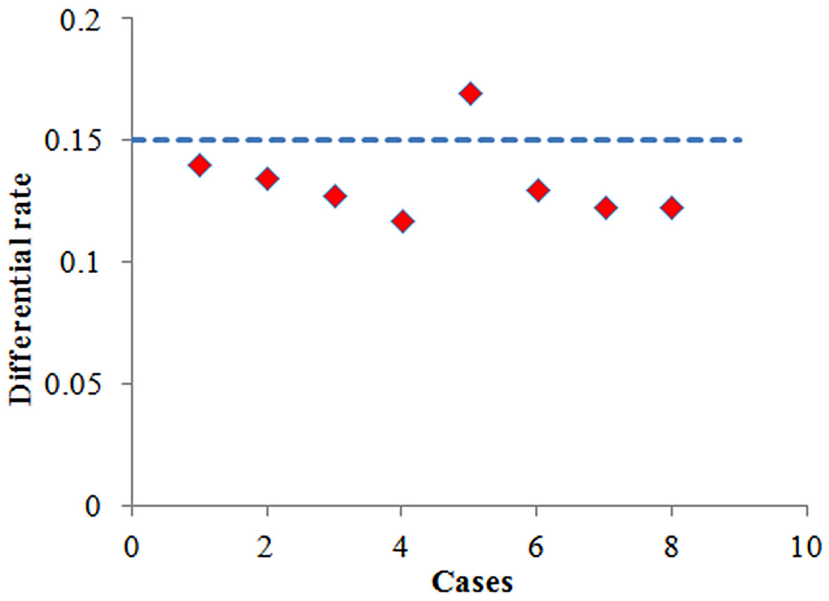

Fig 5. The differential rate in different cases. There are a total of eight kinds of cases. The maximum value is no more than 20% and The minimum value is no less than 10%.

doi:10.1371/journal.pone.0132418.g005

characteristics provide the possibility for the application of solving the inverse problem of reservoir history matching.

Numerical Examples

Case1: Theoretical reservoir model

This reservoir simulation model is a simple 2D model with 25×25 two-dimensional log-permeability distributions with great heterogeneity. It includes a total of 9 injection wells and 4 production wells. In this model, the period of history matching is 1200 days, and every control step is 120 days. In other words, the well control parameter will be adjusted per every 120 days. In the process of history matching, the permeability at each of the 625 cells in the grid must be calculated and updated. The production data required for matching includes the flowing bottom-hole pressure and the oil and water production of the wells. We can regard the simulation result as the observed data of the reservoir by adding the errors of the normal distribution to the result of the initial model. We generate 100 initial reservoir models based on prior geological information by using the Sequential Gaussian Simulation Method (Fig 6).

From Fig 6, we can see that there exist large differences between the distributions of the initial permeability among those models; the hypertonic and hypotonic zones are quite different. However, every model has reflected the basic reservoir characteristics: the direction of the hypertonic stripes and their approximate positions. By generating 100 stochastic models have reduced the construction error of the models because of their differences.

Fig 7 shows the real permeability field which generated from 100 random reservoir models. Because the parameters in those models are in accordance with Gaussian distributions, the real permeability (Fig 7) is also in accordance with Gaussian distributions. We use Algorithm II and III for the optimization of history matching objective function, and then compare the optimization results.

To obtain the minimum value of Eq 14, we use both Algorithm II and Algorithm III. In Algorithm II: We set the initial step to 1 in the linear search process and obtain the stochastic gradients, which requires 5 random perturbations at each step; then, we iterate with their average gradient.

In Algorithm III: The initial setting and the process of generating the stochastic gradient are consistent with Algorithm II. The difference between the two algorithms is that the gradients for the iterations are not the same. In Algorithm II, the calculated stochastic gradient is used directly for optimization, while in Algorithm III, a new gradient must be generated from the previous stochastic gradient with quantitative finite differences.

From Fig 8, we can see that several hypertonic stripes appear in the matching permeability field, and the trends of the stripes are similar to those in the reference permeability field. In Fig 8(A), the distributions of the hypertonic stripes are clearly in the matching permeability field but are very different from those in the reference permeability field because of the randomness of the gradient obtained by using Algorithm II. In Fig 8(B), the distribution of the hypertonic stripes not only is clear in the matching permeability field but also is much closer to those in the reference permeability field. With this result, we can realize that by using Algorithm III, we can describe the reference permeability field more clearly than by using Algorithm II.

Now, we analyze the optimization result of the production data by using both algorithms. Fig 9 shows the rate of water production of well Pro-4. The observed data varies greatly with the increase of time. We can obtain the simulation result that the stratigraphic parameters can be inversed by Eq 14. From Fig 10, we can realize that the matching curve from using Algorithm III is much closer to the observed data than that obtained by using Algorithm II.

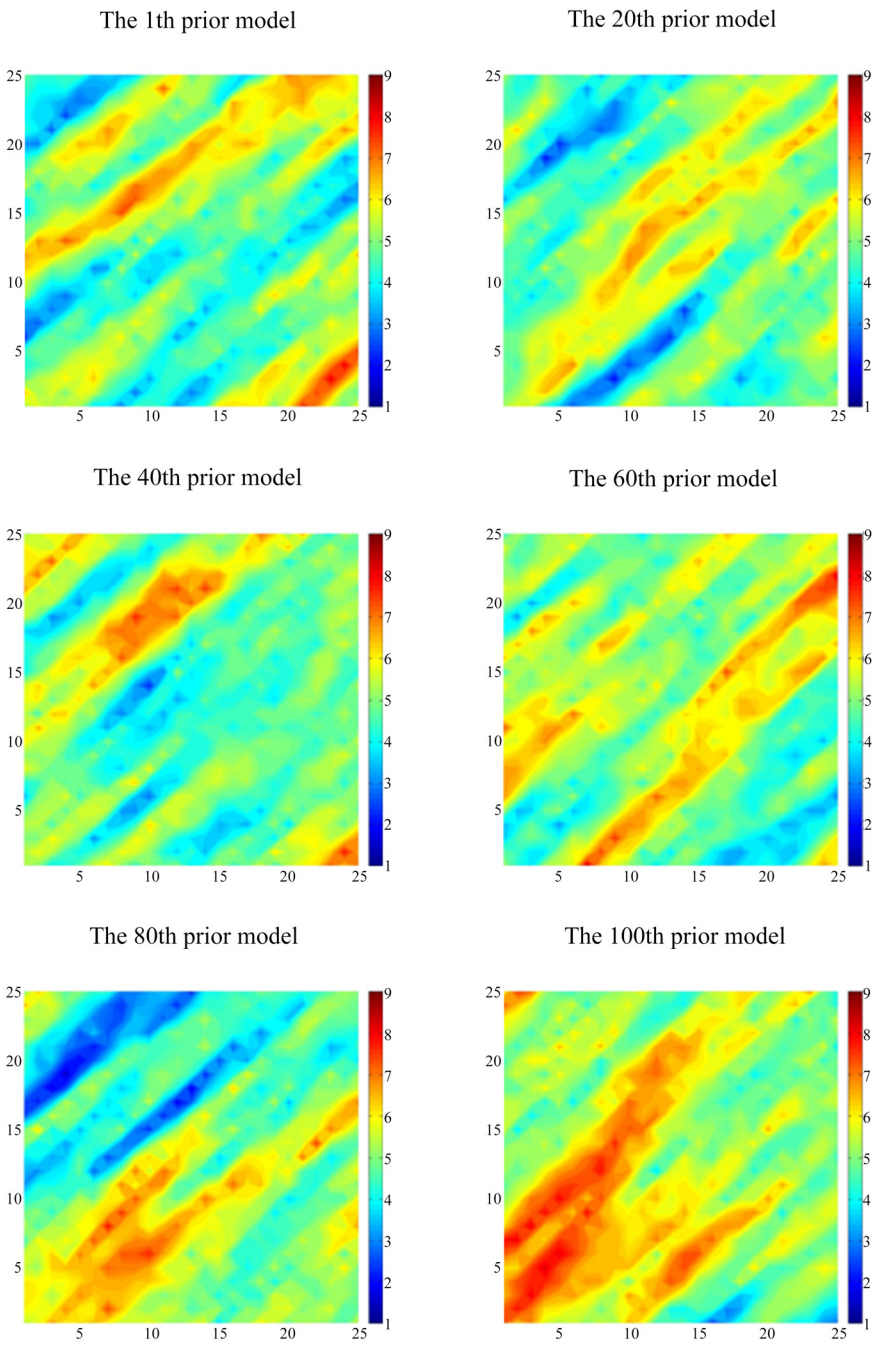

Fig 6. The log-permeability distribution of the prior models. These models are generated by the professional geological modeling software: Petrel.

doi:10.1371/journal.pone.0132418.g006

Case 2: Three-phase reservoir model

This reservoir simulation model is a 20×30 two-dimensional model, and it has a total of 6 production wells. In this model, the period of history matching is 9000 days, and every control step is 30 days. In other words, the well control parameters will be adjusted per every 30 days. In the process of history matching, the variables that must be resolved include the permeability

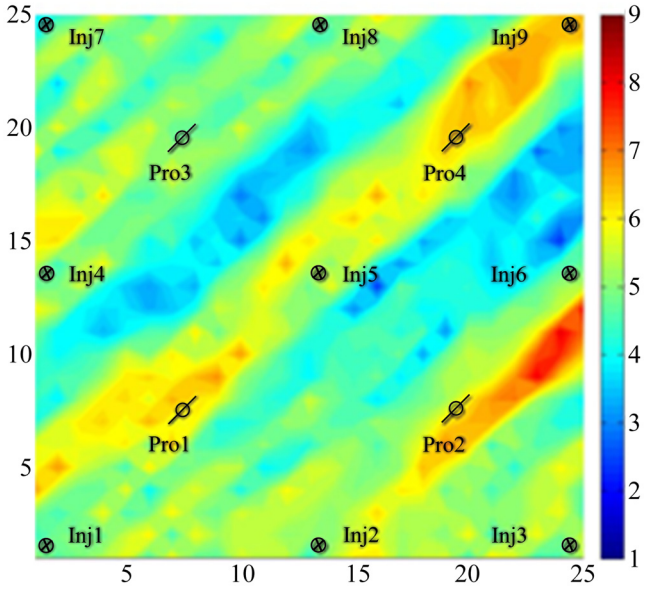

Fig 7. The real log-permeability distribution. This reservoir uses five spot pattern for numerical simulation.

doi:10.1371/journal.pone.0132418.g007

at each cell, and the number of cells is 600. The production data that need matching include the bottom-hole flowing pressure, oil, gas and water production of wells.

In the production process of this reservoir model, the 6 wells began constant production, one-at-a-time in a sequence per 90 days. The first one is Pro-1, and the last one is Pro-6. After 1630 days, Pro-1 was converted into a water injection well while other production wells retained the same quota.

Before the start of the numerical simulation, we set the initial permeability field, which is based on reservoir information, as in Fig 10(A). Fig 10(B) shows the real permeability field. Then, we used two algorithms for optimization.

Fig 11 shows the estimate of the log permeability that was obtained by history matching of the observed bottom pressure and the WOR (water oil ratio) data with the two algorithms. The

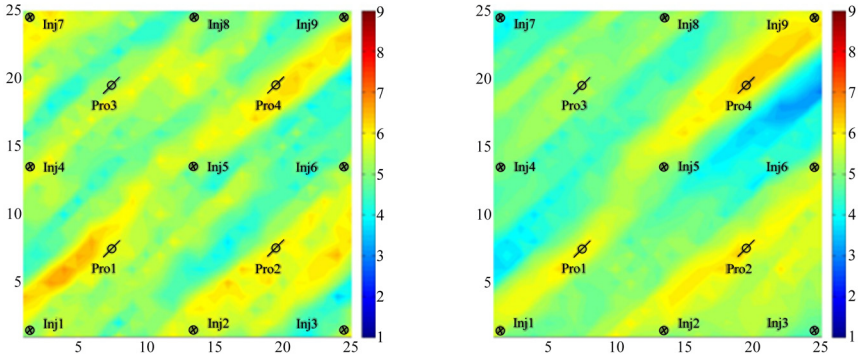

Fig 8. The matching result of the two algorithms. A:The simulation result obtained by Algorithm II, B: The simulation result obtained by Algorithm III.

doi:10.1371/journal.pone.0132418.g008

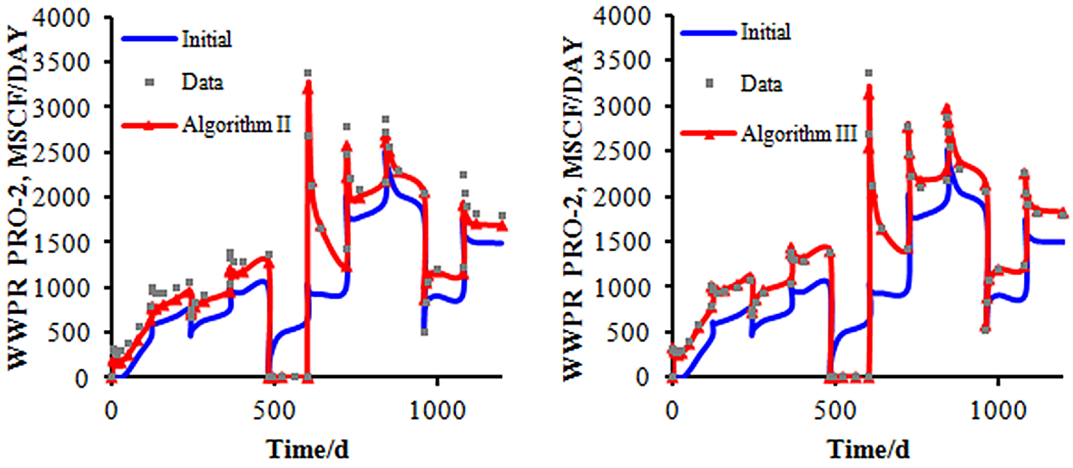

Fig 9. The matching result of the production data of the two algorithms. The bule line repesents the initial model's simulation result; the gray points represent the observed data; the red line represents the final matching model's simulation result.

doi:10.1371/journal.pone.0132418.g009

matching result obtained by Algorithm II, as shown in Fig 11(A), shows the basic permeability channel structure without exactly matching the truth.

In Fig 11(B), the matching result shows that the matching result by Algorithm III has correctly reflected the permeability channel features, especially near the production wells where the reservoir characteristics can be exactly described.

From Fig 12, we can see that there emerges a very large gap between the initial and observed data. After the history matching by using the two algorithms respectively, the results agree well with the observed data.

The matching results of the cumulative gas production are shown in Fig 12. In the early stages of oil field development, the gas production data of both the initial and real model are consistent. Those data begin to change at the 6000th day. After producing for 9000 days, the cumulative production of gas in the real model is $1.0203\times10^{7}m^{3}$. The matching result by Algorithm II is $9.3057\times10^{6}m^{3}$, while the matching result by Algorithm III is $1.0191\times10^{7}m^{3}$. Algorithm III has a better matching effect than Algorithm II.

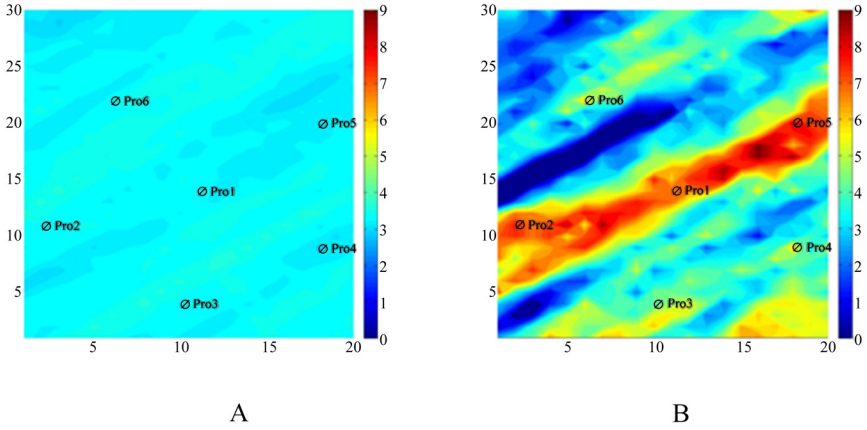

Fig 10. The different log-permeability distribution. A:The initial permeability field; B:the real permeability field.

doi:10.1371/journal.pone.0132418.g010

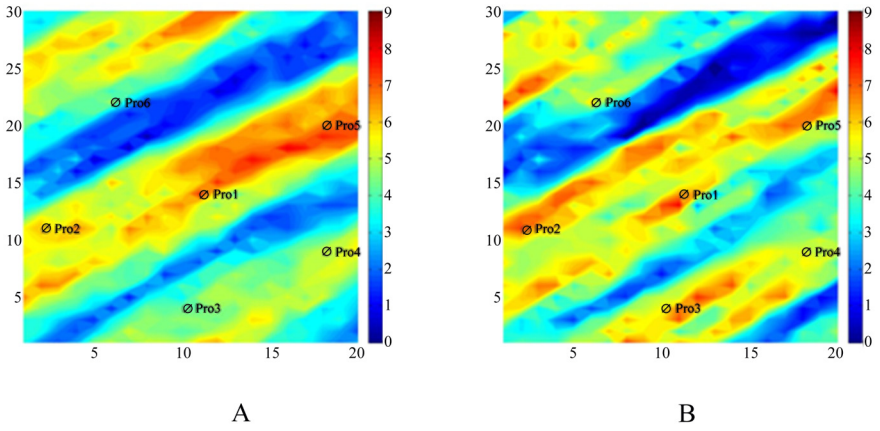

Fig 11. The matching results of the two algorithms. A:The simulation result obtained by Algorithm II, B: The simulation result obtained by Algorithm III.

doi:10.1371/journal.pone.0132418.g011

Both Algorithms II and III have the same matching result for the other production data. Fig 13 shows the matching result of the other production data by Algorithm III. We can realize that through Algorithm III, we can obtain a satisfied optimization result, and Algorithm III can resolve the inverse problem of optimization.

One of the criteria for comparing two algorithms is the rate of convergence of the objective function. The initial objective function value is 4.16×10^5 (Fig 14). After a sufficient number of simulations, both of the algorithms could minimize the objective function value to near zero.

From Figs 12 and 13, we can require that both algorithms can match the production data very well. In Fig 14, we can see that the objective function optimized by Algorithm III has a higher speed of decline than that obtained from Algorithm II initially, but after 100 iterations, the objective function values of the two algorithms are nearly the same.

Through the above comparison of several aspects, we can see that Algorithm III has a higher rate of convergence and reflects the geological information more clearly than Algorithm II, but the variance matrix constituted by the formation parameters must be reduced dimensionally by decomposition before the optimization (in Problem Formulation part), which greatly reduces the number of parameters (it reduces to 100 in this paper). This method increases the

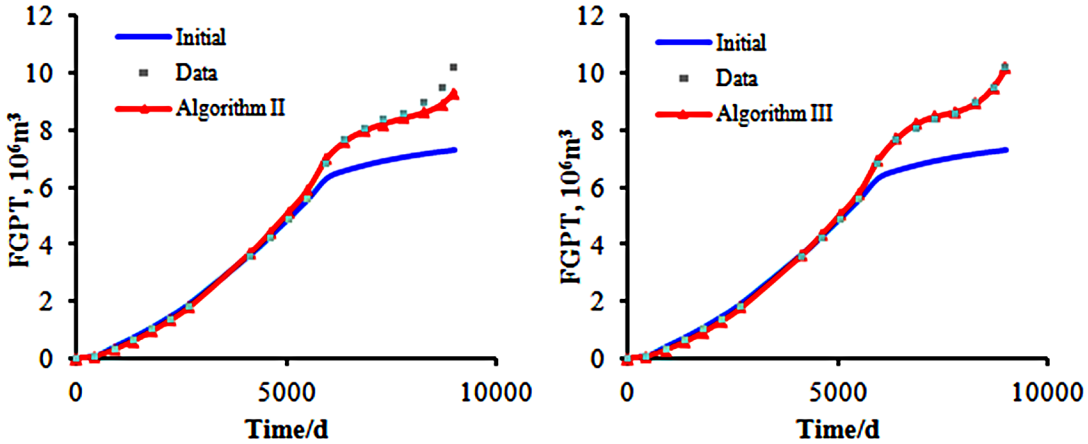

Fig 12. The total gas production of the two algorithms. The bule line repesents the initial model's simulation result; the gray points represent the observed data; the red line represents the final matching model's simulation result.

doi:10.1371/journal.pone.0132418.g012

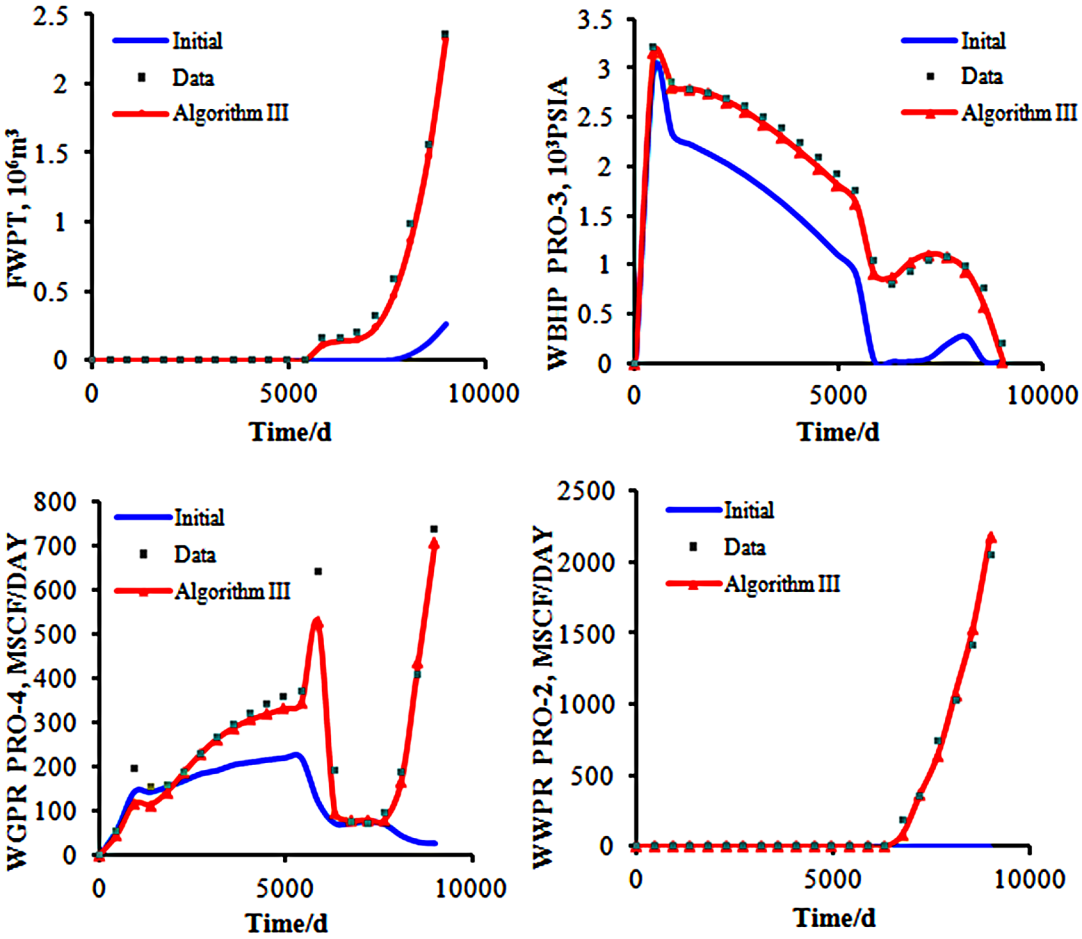

Fig 13. The matching result of the production data by Algorithm III. The bule line repesents the initial model's simulation result; the gray points represent the observed data; the red line represents the final matching model's simulation result.

doi:10.1371/journal.pone.0132418.g013

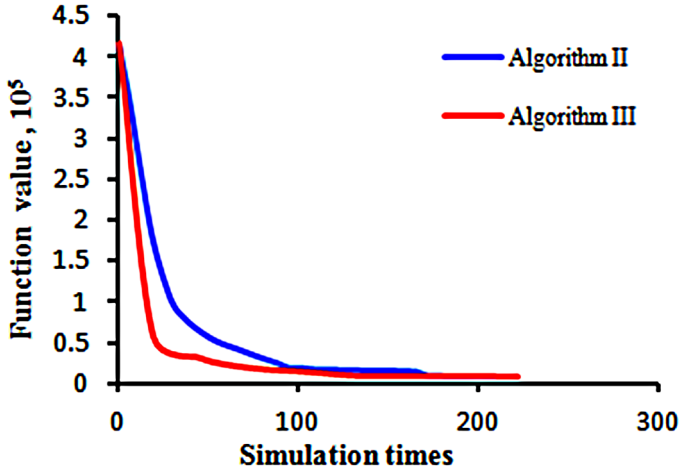

Fig 14. The convergence of the objective function. The blue line represents the objective function value versus the simulation runs by using Algorithm II. The red line represents the objective function value versus the simulation runs by using Algorithm III.

doi:10.1371/journal.pone.0132418.g014

errors between the optimization parameters and the observed data. Thus, when there are too many formation parameters, tthe advantages of Algorithm III over Algorithm II are not very obvious. Therefore, there exists potential for further research in this field.

Conclusions

The present paper has investigated the application of finite differences with a stochastic algorithm for history matching in reservoir models. We use this method to optimize and resolve the Bayesian inverse problem, which depends on posterior distributions. Stochastic algorithms (such as SPSA, directional derivatives) determine the direction of the gradient with a random perturbation. The disadvantages are that the direction of the approximate gradient deviates from the direction of the true gradient; there are a large number of iterations; it has a slow rate of convergence, and it is vulnerable to local loops. The Hybrid algorithm obtains an approximate gradient that is much closer to true gradient by partial finite difference. It increases the accuracy of the search direction and improves the rate of convergence. This paper has verified (by the mathematical model and reservoir examples) that the Hybrid algorithm has the following advantages:

- Hybrid algorithm introduced with finite difference on the basis of a stochastic algorithm has greatly improved the accuracy of the approximate gradient, and this gradient is closer to the true gradient as the iteration steps increase;
- The approximate cosine formula to determine the accuracy of the approximate gradient has a high degree of accuracy, which provides a criterion to judge the accuracy of the approximate gradient for actual reservoirs;
- Compared with the stochastic algorithm based on a directional derivative, the Hybrid algorithm has a faster rate of convergence and can better describe geologicalinformation.

Acknowledgments

This work is supported by “China Important National Science & Technology Specific Projects” under Grant 2011ZX05024-002-008, "The National Natural Science Foundation of China" under Grant 61004095, “the Specialized Research Fund for the Doctoral Program of Higher Education of China” under Grant 20100133120003, “the Fundamental Research Funds for the Central Universities” under Grant13CX02053A, and Changjiang Scholars and Innovative Research Team in University (IRT1294), and "863 Important Project" under Grant 2013AA09A215 and “the Fundamental Research Funds for the Central Universities” under Grant 15CX05035A. PetroChina Coalbed Methane Company Limited provided support in the form of salaryfor author Xia Yan, but did not have any additional role in the study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Author Contributions

Provided the theoretical instructions for how to solve the practical problems: XY. Improved of the models and written the program and the manuscript: KZ ZW. Tested the actual models: JY. Tested the theoretical algorithm: LZ.

References

1. Oliver DS, He N, Reynolds A. Conditioning Permeability Fields to Pressure Data. Proceedings of the 5th European Conference on the Mathematical Oil Recovery-Leoben Austria. 1996.
2. Gao G, Zafari M, Reynolds A. Quantifying uncertainty for the PUNQ-S3 problem in a Bayesian setting with RML and EnKF. Society of Petroleum Engineers. 2006; 11(4):506–15.

3. Reynolds AC, He N, Oliver D. Reducing uncertainty in geostatistical description with well testing psure data. Reservoir Characterization—Recent Advances American Association of Petroleum Geresologists. 1999:149–62.
4. Yang PH, Wilson A. Automatic History Matching with Variable-Metic Methods. Society of Petroleum Engineers. 1988:995–1001.
5. Emanuel AS, Milliken W. History matching finite difference models with 3D streamlines. SPE Annual Technical Conference and Exhibition; 27–30 September; New Orleans;1998.
6. Vasco DW, Yoon S, Datta-Gupta A. Integrating dynamic data into high-resolution reservoir models using streamline-based analytic sensitivity coefficients Society of Petroleum Engineers. 1999; 4 (4):389–99.
7. Wang Y, Kovscek A. Streamline approach for history matching production data. Society of Petroleum Engineers. 2000; 5 (4):353–62.
8. Soleng H. Oil reservoir production forecasting with uncertainty estimation using genetic algorithms. Congress of Evolutionary Computing. 1999; 2.
9. Li L, Peng H, Kurths J, Yang Y, Schellnhuber H. Chaos-order transition in foraging behavior of ants. Proceedings of the National Academy of Sciences. 2014; 111(23):8392–7.
10. Cai J, Ma X, Li L, Yang Y, Peng H, Wang X. Chaotic ant swarm optimization to economic dispatch. Electric Power Systems Research. 2006; 77(10):1373–80.
11. Evensen G. Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics. Journal of Geophysical Research:Oceans. 1994; 99(C5):10143–62.
12. Evensen G. The ensemble Kalman filter: theoretical formulation and practical implementation. Ocean Dynamics. 2003; 53(4):343–67.
13. Spall J. Multivariate Stochastic Approximation Using a Simultaneous Perturbation Gradient Approximation. Transactions on Automatic Control. 1992; 37(3):332–41.
14. Spall JC, Cristion J. Nonlinear adaptive control using neural networks: estimation with a smoothed form of simultaneous perturbation gradient approximation. American Control Conference. 1994; 3:2560–4.
15. Klie H, Rodriguez A, Thomas S. Assessing the Value of Sensor Information In 4-D Seismic History Matching. Seg Technical Program Expanded Abstracts. 2006; 26(1).
16. Gao G, Li G, Reynolds A. A Stochastic Optimization Algorithm for Automatic History Matching. Society of Petroleum Engineers. 2007; 12(2):196–208.
17. Li G, Reynolds A. Uncertainty Quantification of Reservoir Performance Predictions using a Stochastic Optimization Algorithm. Computational Geosciences. 2011; 15(3):451–62.
18. Zhao H, Chen C, Do S, Oliveira D, Li G, Reynolds A. Maximization of a Dynamic Quadratic Interpolation Model for Production Optimization Society of Petroleum Engineers. 2011; 18(6):1012–25.
19. Yan X, Reynolds A. Optimization Algorithms Based on Combining FD Approximations and Stochastic Gradients Compared With Methods Based Only on a Stochastic Gradient. Society of Petroleum Engineers. 2014; 19(5):873–90.
20. Wang J, Zabaras N. A Bayesian inference approach to the inverse heat conduction problem. International Journal of Heat and Mass Transfer. 2004; 47(17–18):3927–41.
21. Marzouk Y, Xiu D. A stochastic collocation approach to Bayesian inference in inverse problems. Communications in Computational Physics. 2009; 6(4):826–47.
22. Ma X, Zabaras N. An efficient Bayesian inference approach to inverse problems based on an adaptive sparse grid collocation method. Inverse Problems. 2009; 25(3).
23. Galbally D, Fidkowski K, Willcox K, Ghattas O. Non-linear model reduction for uncertainty quantification in large-scale inverse problems. International Journal for Numerical Methods in Engineering. 2010; 81 (12):1581–608.
24. Lieberman C, Willcox K, Ghattas O. Parameter and state model reduction for large-scale statistical inverse problems. SIAM Journal on Scientific Computing 2010; 32(5):2523–42.
25. Jensen HA, Millas E, Kusanovic D, Papadimitriou C. Model-reduction techniques for Bayesian finite element model updating using dynamic response data. Computer Methods in Applied Mechanics and Engineering. 2014; 279:301–24.
26. Soize C. A computational inverse method for identification of non-Gaussian random fields using the Bayesian approach in very high dimension. Computer Methods in Applied Mechanics and Engineering. 2011; 200(45–46):3083–99.
27. Pradlwarter H. Relative Importance of Untertain Structural Parameters.Part I:Algorithm. Computational Mechanics. 2007; 40(4):627–35.