NUMERICAL MATHEMATICS: Theory, Methods and Applications
Numer. Math. Theor. Meth. Appl., Vol. **1**, No. 4, pp. 460-474 (2008)

A Stochastic Approximation Frame Algorithm with Adaptive Directions

Zi Xu[1] and Yu-Hong Dai[2]*

[1] *Department of Mathematics, College of Sciences, Shanghai University, Shanghai, 200444, P.R.China.*
[2] *State Key Laboratory of Scientific and Engineering Computing, Institute of Computational Mathematics and Scientific/Engineering Computing, The Academy of Mathematics and Systems Science, Chinese Academy of Sciences, P.O.Box 2719, Beijing, 100190, P.R.China.*

Received 24 October, 2007; Accepted (in revised version) 17 April, 2008

Abstract. Stochastic approximation problem is to find some root or extremum of a nonlinear function for which only noisy measurements of the function are available. The classical algorithm for stochastic approximation problem is the Robbins-Monro (RM) algorithm, which uses the noisy evaluation of the negative gradient direction as the iterative direction. In order to accelerate the RM algorithm, this paper gives a frame algorithm using adaptive iterative directions. At each iteration, the new algorithm goes towards either the noisy evaluation of the negative gradient direction or some other directions under some switch criterions. Two feasible choices of the criterions are proposed and two corresponding frame algorithms are formed. Different choices of the directions under the same given switch criterion in the frame can also form different algorithms. We also proposed the simultanous perturbation difference forms for the two frame algorithms. The almost surely convergence of the new algorithms are all established. The numerical experiments show that the new algorithms are promising.

AMS subject classifications: O242.1
Key words: Stochastic approximation, conjugate gradient, adaptive directions.

1. Introduction

Stochastic approximation algorithm provides a simple and effective approach for finding root or minimum of function whose evaluations are contaminated with noise. Consider a n-dimensional loss function $f : \mathscr{R}^n \to \mathscr{R}$, with gradient $g : \mathscr{R}^n \to \mathscr{R}^n$. We have that $g(x) = 0$ if and only if $x = x^*$ when f has a unique local minimizer $x^* \in \mathscr{R}^n$. If the direct noisy estimate of the gradient function $\tilde{g}_k$ is available, the Robbins-Monro(RM) algorithm [1](extended by Blum [2] to multidimensional systems) estimates a root of g with the following recursion:

$$x_{k+1} = x_k - \alpha_k \tilde{g}_k, \tag{1.1}$$

*Corresponding author. *Email addresses:* `xuzi@lsec.cc.ac.cn` (Z. Xu), `dyh@lsec.cc.ac.cn` (Y.-H. Dai)

$$\tilde{g}_k = g(x_k) + \varepsilon_k, \tag{1.2}$$

where ε_k is the noise and α_k is a sequence that satisfies

$$\alpha_k > 0, \quad \sum_{k\geq 1} \alpha_k = \infty, \quad \sum_{k\geq 1} \alpha_k^2 < \infty. \tag{1.3}$$

Since the direct noisy measurements $\tilde{g}_k$ are sometimes not available, Kiefer and Wolfowitz [3] introduced the finite difference form of the RM algorithm, which employs an estimator for the gradient denoted by $\hat{g}(x_k)$, whose ith component is given by

$$\hat{g}_i(x_k) = \frac{\tilde{f}(x_k + c_k e_i) - \tilde{f}(x_k - c_k e_i)}{2c_k}, \tag{1.4}$$

where e_i is the unit vector along the ith axis and $\tilde{f}$ is a noisy measurement of the function value f. We can call this algorithm finite difference stochastic approximation (FDSA) algorithm or KW algorithm simply. The almost surely convergence of the KW algorithm is also given by Kiefer and Wolfowitz [3]. The major disadvantage of the KW algorithm is that it requires $2n$ measurements of the function value per iteration. By contrast, the random direction stochastic approximation (RDSA) algorithm first given by Kushner and Clark [4], needs only two measurements per iteration. It has the following recursion:

$$x_{k+1} = x_k - \alpha_k \left[\frac{\tilde{f}(x_k + c_k \xi_k) - \tilde{f}(x_k - c_k \xi_k)}{2c_k} \right] \xi_k. \tag{1.5}$$

A special case of the RDSA algorithm is the simultaneous perturbation stochastic approximation (SPSA) algorithm introduced by Spall [5] which employs the estimator:

$$\hat{g}(x_k) = \left[\frac{\tilde{f}(x_k + c_k \xi_k) - \tilde{f}(x_k - c_k \xi_k)}{2c_k} \right] \zeta_k, \tag{1.6}$$

where ξ_k is chosen from a distribution that has to satisfy some particular constraints, and the ith component of ζ_k are given by

$$\zeta_k^{(i)} = 1/\xi_k^{(i)}. \tag{1.7}$$

Since in fact the Bernoulli distribution is the only choice that has ever been advocated for SPSA, SPSA is a special case of RDSA, though it does bear remarking that the use of a Bernoulli distribution with RDSA had not been suggested until after SPSA had been introduced. The FDSA, RDSA and SPSA algorithm exhibit similar convergence properties.

The RM algorithm is a classical stochastic approximation algorithm and exhibits the property that it converges to a stationary point almost surely. The major disadvantage of RM algorithm and its difference forms including the FDSA, RDSA and SPSA algorithms are their slow speed of convergence. There have been many efforts to accelerate the RM algorithm. Most of them consist in the choice of the step size α_k, such as the Kesten algorithm (see Kesten [6], Delyon and Juditsky [7]). Spall [8] has given an extensive review on the choice of α_k in stochastic approximation algorithms when the iterative direction is $-\tilde{g}_k$.

Note that, till now, few people have done research in the choice of iterative direction. Because of the influence of noise, $-\tilde{g}_k$ may not always be the best iterative direction, especially when the function have narrow valleys. In fact, in practice, we find that the choice of some faster iterative directions in the initial iterations can yield better finite-sample behavior. So, in this paper, we shall introduce a frame algorithm which employs some other iterations $\tilde{d}_k$ in the initial iterations and change to $-\tilde{g}_k$ in the latter iterations. Here we should note in [13] that a conjugate direction method is given if some information about the Hessian need be known. In practice, it is difficult to obtain second-order information and hence the method is restrictive in application.

The rest of this paper is organized as follows. In the next section, we present a general frame algorithm and two different switch criterions for our algorithm, and then two new algorithms according to the two criterions are proposed. In Section 3, we give the convergence analysis of the new frame algorithm. The difference form of these algorithms are given in Section 4. Numerical results and some implementations are reported in Section 5. The last section is some concluding remarks.

2. Our frame algorithm

In this section, we consider the situation that the direct noisy estimator $\tilde{g}_k$ is available and give a new frame algorithm. Our basic idea is as follows. We keep the step size α_k the same with the RM algorithm and let $\|g_k\|$ decrease faster than in the RM algorithm in initial iterations through using some other faster degressive directions such as the noisy conjugate gradient direction. We change the iterative direction to the negative noisy gradient direction in the later iterations to keep stabilization. Thus, the step size of the new algorithm can be expected to be larger than the RM algorithm when $\|g_k\|$ attains the same accuracy. By doing this, $\|g_k\|$ in the new algorithm can be expected to decrease more than in the RM algorithm with the same number of measurements being used. In other words, the new algorithm will have better finite-sample performance than the RM algorithm. It has the following recursion:

$$x_{k+1} = x_k + \alpha_k d_k, \tag{2.1}$$

$$d_k = \begin{cases} \tilde{d}_k, & \text{If some criterion satisfied;} \\ -\tilde{g}_k, & \text{otherwise.} \end{cases} \tag{2.2}$$

We can also call it as adaptive direction stochastic approximation frame algorithm. Though $\tilde{d}_k$ can be chosen as any directions that can make $\|g_k\|$ has a descent in one or several iterations, directions that can let $\|g_k\|$ descend faster than the noisy negative gradient direction are preferred. When the iterations near the solution, in order to guarantee the convergence of the frame algorithm, we change to use the noisy negative gradient directions $-\tilde{g}_k$. In this procedure, the criterion is an important factor. Bad criterion may not guarantee the convergence. In the following, we give two available criterions for this frame algorithm.

2.1. Two criterions for new frame algorithm

Our purpose is to let $||g_k||$ have a certain descent during several iterations. However, under the influence of noise we can not exactly get the value of $||g_k||$. Fortunately, we find that in most cases, with a large probability, the minimal value of $||\tilde{g}_k||$ over every finite iterations still has a certain descent if $||g_k||$ is degressive during this process. So, instead of $||g_k||$, our new direction $\tilde{d}_k$ should let the minimal value of $||\tilde{g}_k||$ have a definite descent during several iterations though there is noise. Based on this idea, we can give the first switch criterion as follows.

Given positive constants β and δ and an integer r. During the initial $2r$ iterations, d_k equals $\tilde{d}_k$ if $||\tilde{g}_k|| < \beta$. This is to avoid the overflow of $||\tilde{g}_k||$ as a result of noise. When $k > 2r$, if the norm of noisy gradient has a descent in the foregoing r iterations and if $||\tilde{g}_k|| < \beta$, we let d_k equal $\tilde{d}_k$. Otherwise, we think that the direction $\tilde{d}_k$ can not work well and hence change to the noisy gradient direction $-\tilde{g}_k$ from then on. We can formulate this criterion as follows. Define the one dimensional function:

$$\mathscr{I}(t) = \begin{cases} 1, & \text{if } t \geq 0; \\ 0, & \text{otherwise,} \end{cases} \tag{2.3}$$

and let

$$t_k = \mathscr{I}(||\tilde{g}_k|| - \beta), \tag{2.4}$$
$$q_k = \mathscr{I}[||\tilde{g}_{u_2(k)}|| - (||\tilde{g}_{u_1(k)}|| - \delta)], \tag{2.5}$$

where

$$u_1(k) = \underset{1 \leq i \leq k-r}{arg \min} ||\tilde{g}_i||$$

and $u_2(k) = \underset{k-r+1 \leq i \leq k}{arg \min} ||\tilde{g}_i||$. Our first switch criterion computes d_k as follows:

$$d_k = \begin{cases} (1-t_k)\tilde{d}_k - t_k\tilde{g}_k, & \text{if } k < 2r; \\ \varphi_k(1-t_k)\tilde{d}_k - [1-\varphi_k(1-t_k)]\tilde{g}_k, & \text{otherwise.} \end{cases} \tag{2.6}$$

where φ_k is a sequence that satisfies $\varphi_0 = 1$ and

$$\varphi_k = (1-q_k)\varphi_{k-1}. \tag{2.7}$$

We call this criterion (2.6) by *Criterion I* in this paper. Under *Criterion I*, we expect new direction $\tilde{d}_k$ can make the the minimal value of $||\tilde{g}_k||$ has a descent of at least δ every r iterations. We can also see from *Criterion I* that, we can guarantee this because once $\varphi_k = 0$ for some k, φ_l will be zero and d_l will equal $-\tilde{g}_l$ for all $l \geq k$ according to (2.6) and (2.7).

Influenced by noise, sometimes the condition $||\tilde{g}_{u_2(k)}|| \leq ||\tilde{g}_{u_1(k)}|| - \delta$ in *Criterion I* seems too strict. Considering this, we give the second criterion by using an expanding trust region for $||\tilde{g}_k||$ instead of the condition in (2.5). We think that sometimes $||\tilde{g}_k||$ does not decrease through several iterations is because of the influence of noise. Under this criterion, though sometimes $\|\tilde{g}_{u_2(k)}\| \geq \|\tilde{g}_{u_1(k)}\| - \delta$, it still has chance to go towards $\tilde{d}_k$ till for some k_0 that satisfies $\|\tilde{g}_{k_0}\| > k_0 s$, where s is a very small positive constant. This criterion give more possibility for the frame algorithm to go towards the direction $\tilde{d}_k$. By doing this, it can also avoid the too large alteration which may sometimes go out of the neighborhood of the solution. It computes d_k as follows:

$$d_k = \begin{cases} -t_k \tilde{g}_k + (1 - t_k)\tilde{d}_k, & \text{if } k < 2r; \\ -[1 - \tilde{\varphi}_k(1 - t_k q_k)]\tilde{g}_k + \tilde{\varphi}_k(1 - t_k q_k)\tilde{d}_k, & \text{otherwise.} \end{cases} \tag{2.8}$$

where $\tilde{\varphi}_k$ is a sequence that satisfies $\tilde{\varphi}_0 = 1$ and

$$\tilde{\varphi}_k = \mathscr{I}(\|\tilde{g}_k\| - ks)\tilde{\varphi}_{k-1}. \tag{2.9}$$

Here, t_k and q_k are also chosen as in (2.4) and (2.5) respectively. We call criterion (2.8) by *Criterion II* in this paper. The frame algorithm (2.1)-(2.2) under *Criterion I* and *Criterion II* is called by *Frame I* algorithm and *Frame II* algorithm respectively.

2.2. Choice of directions $\tilde{d}_k$

The only problem remained for *Frame I* algorithm and *Frame II* algorithm is how to choose a suitable direction $\tilde{d}_k$. Though $\tilde{d}_k$ can be chosen as any direction that can make $\|g_k\|$ has a descent in one or several iterations, directions that can let $\|g_k\|$ descend faster than the noisy negative gradient direction are preferred. We all know, conjugate gradient method (see [9]) converges much faster than the steepest descent method when there is no noise. The conjugate gradient method is based on the idea that the convergence to the solution could be accelerated if we minimize the function over the hyperplane that contains all previous search directions, instead of minimizing it over just the line that points down gradient. So we expect that the noisy conjugate gradient directions can still have a good performance in the initial iterations. It may work worse than the noisy steepest descent direction $-\tilde{g}_k$ at each iteration separately, however, we expect it can work better through several iterations. This is also the cause that conjugate gradient method works better than the steepest descent method in deterministic optimization. Now, we give the choice of $\tilde{d}_k$ as follows:

$$\tilde{d}_k = -\tilde{g}_k + \gamma_k d_{k-1}, \tag{2.10}$$

which is a combination of $-\tilde{g}_k$ and d_{k-1}. Here, γ_k can be chosen similarly as in conjugate gradient method (see [9])in deterministic optimization as follows:

$$\gamma_k = \frac{\|\tilde{g}_k\|^2}{\|\tilde{g}_{k-1}\|^2}, \tag{2.11}$$

$$\gamma_k = \frac{\tilde{g}_k^T(\tilde{g}_k - \tilde{g}_{k-1})}{\|\tilde{g}_{k-1}\|^2}, \tag{2.12}$$

$$\gamma_k = \frac{\tilde{g}_k^T(\tilde{g}_k - \tilde{g}_{k-1})}{d_{k-1}^T(\tilde{g}_k - \tilde{g}_{k-1})}, \tag{2.13}$$

$$\gamma_k = \frac{\|\tilde{g}_k\|^2}{d_{k-1}^T(\tilde{g}_k - \tilde{g}_{k-1})}, \tag{2.14}$$

which are respectively the FR, PRP, HS and DY conjugate gradient methods in deterministic optimization. If we just regard γ_k as the weighted coefficient to combine the two directions d_{k-1} and $-\tilde{g}_k$, we can choose γ_k in some other ways for example:

$$\gamma_k = \frac{\|\tilde{g}_k\|}{\|\tilde{g}_{k-1}\|}. \tag{2.15}$$

In practice, for different functions, the different choices of γ_k will have different numerical performances.

Now, we can give *Frame I* algorithm and *Frame II* algorithm with $\tilde{d}_k$ chosen in (2.10) as follows, called by *Algorithm 2.1* and *Algorithm 2.2* respectively in this paper:

Algorithm 2.1.
step 0. Give r, δ, $\varphi_0 = 1$; choose x_1, measure $\tilde{g}_1$, give β(some proportion of $\|\tilde{g}_1\|$), $k = 1$, $d_1 = -\tilde{g}_1$.

step 1. If some given stopping criteria is satisfied or some given maximum iteration has been arrived, then stop and print some results, otherwise go to step 2;

step 2. Choose α_k from some given sequence; If $k > 1$, compute d_k according to (2.6), goto step 3;

step 3. Compute $x_{k+1} = x_k + \alpha_k d_k$; k=k+1; goto step 1.

Algorithm 2.2. In step 2 of *Algorithm 2.1*, d_k is calculated by (2.8) instead of (2.6).

3. Convergence analysis

In this section, we study the convergence property of *Frame I* algorithm and *Frame II* algorithm including *Algorithm 2.1* and *Algorithm 2.2*. We consider the problem:

$$arg \min_{x\in\Re^n} f(\cdot) \tag{3.1}$$

where $\mathfrak{R}^n$ denotes the n-dimensional Euclidean space and $f\colon \mathfrak{R}^n \to \mathfrak{R}$ is a continuously differentiable function, such that for some constant L we have

$$\|\nabla f(x) - \nabla f(\bar{x})\| \le L\|x - \bar{x}\|, \quad \forall x,\ \bar{x} \in \mathfrak{R}^n \tag{3.2}$$

where $\|\cdot\|$ denotes the norm of a vector.

Let x_k be a sequence generated by the new frame algorithm, $\mathscr{F}_k$ be an increasing sequence of σ-fields which should be interpreted as the history of the algorithm up to time k, just before ε_k is generated. x_k and d_k in (2.1) are $\mathscr{F}_k$-measurable. First, we give the following assumptions:

Assumption 3.1. *The stepsize α_k in* (2.1) *satisfies that the condition* (1.3).

Assumption 3.2. *For all k and with probability 1, ε_k satisfies that*

$$E[\varepsilon_k|\mathscr{F}_k] = 0, \tag{3.3}$$

$$E[\|\varepsilon_k\|^2|\mathscr{F}_k] \le A(1 + \|g_k\|^2), \tag{3.4}$$

where A is a positive deterministic constant.

Lemma 3.1. *There exists a $K_0 \in \mathbb{Z}$, such that $d_k \equiv -\tilde{g}_k$ for all $k \ge K_0$ in* Frame I *algorithm.*

Proof: Assume that there is no such $K_0 \in \mathbb{Z}$ exists. We can conclude that $\varphi_k > 0$, $\forall k > 0$. That is we have $\|\tilde{g}_{u_2(k)}\| < \|\tilde{g}_{u_1(k)}\| - \delta$ for all $k \ge 2r$. Let $u_3(k) = \underset{k-2r+1\le i\le k-r}{arg\min} \|\tilde{g}_i\|$, according to the definition of $u_1(k)$, we have

$$\|\tilde{g}_{u_1(k)}\| \le \|\tilde{g}_{u_3(k)}\| \tag{3.5}$$

and

$$\|\tilde{g}_{u_3(k)}\| \le \|\tilde{g}_{u_1(k-r)}\| - \delta. \tag{3.6}$$

Then, $\forall k = mr,\ m > 1$, we can have the following inequality:

$$\begin{aligned} 0 \le &\|\tilde{g}_{u_2(k)}\| < \|\tilde{g}_{u_3(k)}\| - \delta \le \|\tilde{g}_{u_1(k-r)}\| - 2\delta \\ &< \dots < \|\tilde{g}_{u_1(k-(m-2)r)}\| - (m-1)\delta. \end{aligned} \tag{3.7}$$

Take limits for the above inequality, we have that

$$\lim_{k\to\infty} \|\tilde{g}_{u_2(k)}\| = -\infty. \tag{3.8}$$

This contradicts $\|\tilde{g}_{u_2(k)}\| \ge 0$. So, there must exist some $K_0 \in \mathbb{Z}$, such that $d_k \equiv -\tilde{g}_k$ for all $k \ge K_0$. The lemma is proved. □

Theorem 3.1. *Under Assumptions 3.1 and 3.2, for* Frame I *algorithm, we have either $f(x_k) \to -\infty$ or $f(x_k)$ converges to a finite value and $lim_{k\to\infty} g_k = 0$ with probability 1.*

Proof: Under the Assumptions 3.1 and 3.2, according to the Proposition 3 in section 4 given by Bertsekas and Tsitsiklis [10], and by Lemma 3.1, we can easily conclude that theorem is right. The cause is that when $k \geq K_0$, the *Frame I* algorithm has the recursion:

$$x_{k+1} = x_k - \alpha_k \tilde{g}_k, \quad \forall k \geq K_0. \tag{3.9}$$

Obviously, it satisfies all the conditions given in [10]. So, the theorem is proved. □

Lemma 3.2. *There exists a $K_0 \in \mathbb{Z}$, such that $d_k \equiv -\tilde{g}_k$ for all $k \geq K_0$ in* Frame II *algorithm.*

Proof: If there does not exist $K_0 \in \mathbb{Z}$, then we have that $\tilde{\varphi}_k > 0$, $\forall k$; that is $\forall k$ we have

$$\|\tilde{g}_k\| \geq ks. \tag{3.10}$$

If there exists a sequence $k^{(1)}(\geq 2r)$, $k^{(2)}$, ..., $k^{(l)}(k^{(l)} \geq k^{(l-1)} + r)$, ... , such that

$$\|\tilde{g}_{u_2(k^{(l)})}\| < \|\tilde{g}_{u_1(k^{(l)})}\| - \delta, \ \forall l \geq 1. \tag{3.11}$$

Then according to $k^{(2)} \geq k^{(1)} + r$, we can get

$$\|\tilde{g}_{u_2(k^{(2)})}\| < \|\tilde{g}_{u_1(k^{(2)})}\| - \delta < \|\tilde{g}_{u_2(k^{(1)})}\| - \delta < \|\tilde{g}_{u_1(k^{(1)})}\| - 2\delta. \tag{3.12}$$

Similarly, we can also get that

$$\|\tilde{g}_{u_2(k^{(l)})}\| < \|\tilde{g}_{u_1(k^{(1)})}\| - l\delta. \tag{3.13}$$

According to (3.10), we have

$$\|\tilde{g}_{u_2(k^{(l)})}\| \geq (k^{(l)} - r + 1)s \geq (k^{(1)} - 2r + 1 + lr)s. \tag{3.14}$$

By (3.13) and (3.14), we get

$$(k^{(1)} - 2r + 1 + lr)s < \|\tilde{g}_{u_1(k^{(1)})}\| - l\delta. \tag{3.15}$$

Let $l \to +\infty$ in (3.15), we can see a contradiction. So, there exist a K_1 such that $\forall k \geq K_1$, we have

$$\|\tilde{g}_{u_2(k)}\| \geq \|\tilde{g}_{u_1(k)}\| - \delta. \tag{3.16}$$

That is to say $\forall k \geq K_1$ we have

$$\|\tilde{g}_k\| < \beta. \tag{3.17}$$

This contradicts (3.10). So, there must exist some $K_0 \in \mathbb{Z}$, such that $d_k \equiv -\tilde{g}_k$ for all $k \geq K_0$ in *Frame II* algorithm. □

Theorem 3.2. *Under Assumptions 3.1 and 3.2, for* Frame II *algorithm, we have either $f(x_k) \to -\infty$ or $f(x_k)$ converges to a finite value and $lim_{k\to\infty} g_k = 0$ with probability 1.*

Proof:The statement follows from Assumptions 3.1 and 3.2, according to Proposition 3 in section 4 in [10] and Lemma 3.2. □

Obviously, we can also have the following corollary from Theorems 3.1 and 3.2.

Corollary 3.1. *Under Assumptions 3.1 and 3.2, for* Algorithm 2.1 *and* Algorithm 2.2*, we both have that either $f(x_k) \to -\infty$ or else $f(x_k)$ converges to a finite value and $lim_{k\to\infty} g_k = 0$ with probability 1.*

4. The difference form of new frame algorithm

When we can only use the noisy function value $\tilde{f}_k$ at any point x_k, our new frame algorithm under two criterions can be also extended to the difference form respectively. First, we can use the simultanous perturbation approximation of gradient in the SPSA method [5] instead of $\tilde{g}_k$ in our two new frame algorithms. That is to say we can use $\hat{g}_k$ in (1.6) instead of $\tilde{g}_k$. Our frame algorithm can be changed to the difference form:

$$d_k = \begin{cases} \hat{d}_k, & \text{if some criterion is satisfied;} \\ -\hat{g}_k, & \text{otherwise,} \end{cases} \tag{4.1}$$

where $\hat{d}_k$ can be chosen as any descent directions only using the noisy function value. Now, two criterions given in Section 2 can also be changed to the ones only use the noisy function value. Before giving them, we need to give definitions for some marks. Let

$$\bar{f}_k = \frac{\tilde{f}(x_k + c_k\xi_k) + \tilde{f}(x_k - c_k\xi_k)}{2}, \tag{4.2}$$

$$\hat{t}_k = \mathscr{I}\,(\tilde{f}_k - \beta), \tag{4.3}$$

$$\hat{q}_k = \mathscr{I}\,[\bar{f}_{u_2(k)} - (\bar{f}_{u_1(k)} - \delta)], \tag{4.4}$$

where $u_1(k) = \underset{1\le i\le k-r}{arg\min} \bar{f}_i$, $u_2(k) = \underset{k-r+1\le i\le k}{arg\min}\ \bar{f}_i$, and $\mathscr{I}$ is defined as in (2.3).

The difference form of *Criterion I*, called by *Criterion III*, computes d_k as follows:

$$d_k = \begin{cases} (1-\hat{t}_k)\hat{d}_k - \hat{t}_k\hat{g}_k, & \text{if } k < 2r; \\ \hat{\varphi}_k(1-\hat{t}_k)\hat{d}_k - [1-\hat{\varphi}_k(1-\hat{t}_k)]\hat{g}_k, & \text{otherwise.} \end{cases} \tag{4.5}$$

where $\hat{\varphi}_k$ is a sequence that satisfies $\hat{\varphi}_0 = 1$ and

$$\hat{\varphi}_k = (1-\hat{q}_k)\hat{\varphi}_{k-1}. \tag{4.6}$$

The parameters β, δ and r can be chosen similarly as in Criterion I. So, (2.1) and (4.5) compose the difference form of *Frame I* algorithm, called by *Frame III* algorithm here. When $\hat{d}_k$ is computed by

$$\hat{d}_k = -\hat{g}_k + \gamma_k d_{k-1}, \tag{4.7}$$

where γ_k can be regarded as some weight coefficient, we get the difference form of the *Algorithm 2.1* called by *Algorithm 2.3*.

Similarly, we give the difference form of the *Criterion II*, it computes d_k as follows:

$$d_k = \begin{cases} -\hat{t}_k\hat{g}_k + (1-\hat{t}_k)\hat{d}_k, & \text{if } k < 2r; \\ -[1-\phi_k(1-\hat{t}_k\hat{q}_k)]\hat{g}_k + \phi_k(1-\hat{t}_k\hat{q}_k)\hat{d}_k, & \text{otherwise,} \end{cases} \tag{4.8}$$

where ϕ_k is a sequence satisfies that $\phi_0 = 1$ and

$$\phi_k = \mathscr{I}(\bar{f}_k - ks)\phi_{k-1}. \tag{4.9}$$

The algorithms (2.1) and (4.8) compose the difference form of *Frame II* algorithm, called by *Frame IV* algorithm in this paper. Also, (2.1), (4.7) and (4.8) compose the difference form of *Algorithm 2.2*, called by *Algorithm 2.4* in this paper.

Frame III and *Frame IV* algorithms, the simultaneous perturbation forms of our new frame algorithms, exhibit convergence properties similar to the SPSA algorithm. Similarly to Lemmas 3.1 and 3.2, we can also prove that there exists K_0 such that when $k > K_0$, almost surely we have $d_k \equiv -\hat{g}_k$ for *Frame III* and *Frame IV* algorithms.

5. Numerical experiment and some analysis

5.1. Case I

First, we compare the RM algorithm, *Algorithm 2.1* and *Algorithm 2.2*. Here, in *Algorithm 2.1* and *Algorithm 2.2*, the γ_k in $\tilde{d}_k$ is chosen as in (2.15). This is because in practice, our finite experiments show that this choice of γ_k works better. Five functions were chosen from the collection of unconstrained minimization test problems in Moré et al.[11]. They are listed as follows.

P1. The Helical valley function

$$f(x) = 100[x_3 - 10\theta(x_1, x_2)]^2 + 100[\sqrt{x_1^2 + x_2^2} - 1]^2, \tag{5.1}$$

where

$$\theta(x_1, x_2) = \begin{cases} \frac{1}{2\pi}\arctan(\frac{x_2}{x_1}), & \text{if } x_1 > 0; \\ \frac{1}{2\pi}\arctan(\frac{x_2}{x_1}) + 0.5 & \text{if } x_1 < 0. \end{cases}$$

P2. The Penalty function I

$$f(x) = \sum_{i=1}^{10} 10^{-5}(x_i - 1)^2 + (\sum_{j=1}^{10} x_j^2 - \frac{1}{4})^2. \tag{5.2}$$

P3. The extended Rosenbrock function ($m = 4$)

$$f(x) = \sum_{i=1}^{2}[100(x_{2i} - x_{2i-1}^2)^2 + (1 - x_{2i-1})^2]. \tag{5.3}$$

P4. The Extended Powell singular function

$$\begin{aligned} f(x) = \sum_{i=1}^{4} & (x_{4i-3} + 10x_{4i-2})^2 + 5(x_{4i-1} - x_{4i})^2 \\ & + (x_{4i-2} - 2x_{4i-1})^4 + 10(x_{4i-3} - x_{4i})^4. \end{aligned} \tag{5.4}$$

P5. The extended Rosenbrock function ($m = 10$)

$$f(x) = \sum_{i=1}^{5}[100(x_{2i} - x_{2i-1}^2)^2 + (1 - x_{2i-1})^2]. \tag{5.5}$$

Table 1: The initialization of the parameters and x_0 of five algorithms for 5 functions

P	a	A	s	x_0	f^*
1	1	0	0.001	$[-1;0;0]$	0
2	0.1	50	0.001	$[1;2;...;10]$	7.087e-5
3	1	100	0.01	$[0;0;0;0]$	0
4	0.1	100	0.01	$[1;3;-1;0;...1;3;-1;0]$	0
5	0.1	100	0.01	$[-0.5;...;-0.5]$	0

Table 2: The mean function values at the terminal 2000 evaluations of gradient of four algorithms over 50 independent runs. Approximate 90% confidence intervals are also shown below.

P	RM	Algorithm 2.1	Algorithm 2.2
1	4.13	4.25	1.59
f($\times 10^{-3}$)	[3.84, 4.42]	[4.00, 4.51]	[1.01, 2.18]
2	3.78	2.65	1.05
f($\times 10^{-2}$)	[3.73, 3.84]	[2.24, 3.06]	[0.68, 1.42]
3	2.96	2.72	2.68
f($\times 10^{-2}$)	[2.82, 3.10]	[1.07, 4.37]	[1.65, 3.71]
4	496	4.55	3.52
f($\times 10^{-2}$)	[495, 497]	[3.03, 6.06]	[0.96, 6.09]
5	22.9	2.56	0.77
f($\times 10^{-1}$)	[22.8, 23.0]	[0.68, 4.45]	[5.46, 9.91]

The initial points for the test problems and the optimal function value of each problem are given in the last two columns of Table 1. Normal distribution noise is added to the gradient evaluations of the above five functions, namely, $\xi_k \sim N(0,\ \sigma^2 I)$, I is the identity matrix. In our experiments, we choose $\sigma = 0.5$. For all algorithms, we choose α_k with the form

$$\alpha_k = \frac{a}{k+1+A}. \tag{5.6}$$

For each test problem, we use the same stability constants a and A for all the algorithms given in the first two columns in Table 1. The parameters in *Algorithm 2.1* and *Algorithm 2.2* are the same $\delta = 0.001$, $\beta = \min(\|\tilde{g}_0\|, 500)$ and $r = 20$. Besides, s in *Algorithm 2.2* is given in the third column in Table 1. MATLAB software was used to carry out this study.

For each test problem, we ran each algorithm for 50 times and observed the average function values after 2000 iterations. The results are listed in Table 2. Under each average function value, the approximate 90% confidence interval is also taken down. To avoid the table is too broad, the power of the end points of each interval is not listed, which is the same as that of the average function value.

From Table 2, we can see that both *Algorithm 2.1* and *Algorithm 2.2* perform better than the RM algorithm especially for problem 4. We can see more clearly of the performance

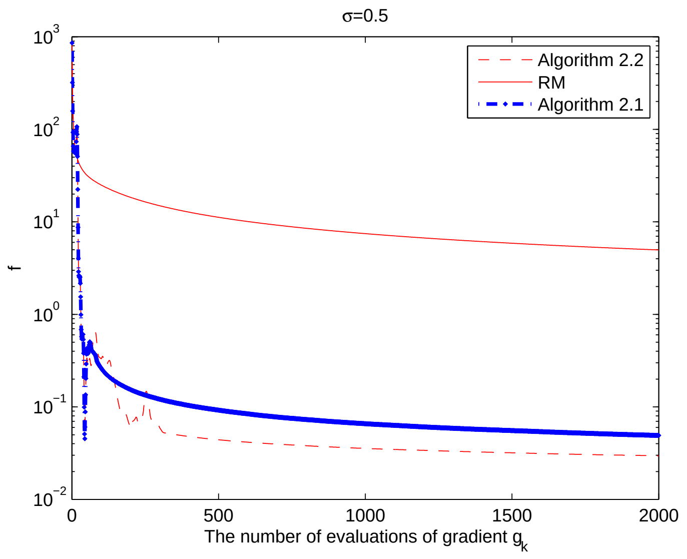

Figure 1: The mean function values of three algorithms during 2000 iterations over 50 independent runs with $\sigma = 0.5$ for problem 4.

of the three algorithms for problem 4 in Figure 1. *Algorithm 2.2* performs best for all 5 problems. Although the gain for each problem is not much, we think that *Algorithm 2.1* and *Algorithm 2.2* are promising alternatives of the RM algorithm.

5.2. Case II

In this part, we mainly consider the case that only noisy measurements of the function value can be used. We compare the SPSA algorithm, *Algorithm 2.3* and *Algorithm 2.4*. Here, in *Algorithm 2.3* and *Algorithm 2.4*, the γ_k in $\hat{d}_k$ is chosen as in (2.15). We consider here the skewed quadratic function given by Spall ([8],chap.6) as follows

$$f(x) = x^T B^T Bx + 0.1 \sum_{i=1}^{n} (Bx)_i^3 + 0.01 \sum_{i=1}^{n} (Bx)_i^4, \tag{5.7}$$

with $n = 10$, where $(\cdot)_i$ represents the ith component of the argument vector Bx and nB is an upper triangular matrix of 1's. The minimum occurs at $x^* = 0$ with $f(x^*) = 0$, and all runs are initialized at $x_0 = [1, 1, ..., 1]^T$. Independent identical distributed noise

Table 3: The mean function values at the terminal 3000 iterations of three algorithms over 50 independent runs. Approximate 90% confidence intervals are also shown below.

SPSA	Algorithm 2.3	Algorithm 2.4
0.98	0.87	0.62
$[0.97, 0.99]$	$[0.82, 0.93]$	$[0.57, 0.68]$

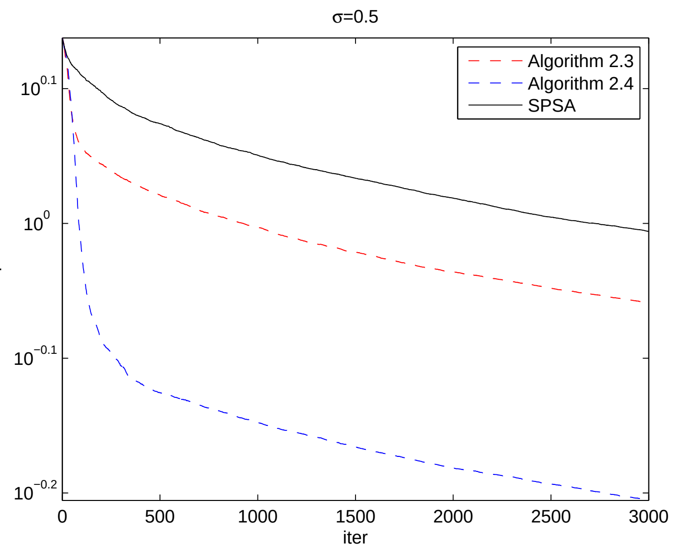

Figure 2: The mean function values of three algorithms at 3000 iterations over 50 independent runs.

ε_k is added to the measurements of function values and the distribution of the noise is $N(0, 0.5^2)$. The parameters in the SPSA algorithm is chosen as $a = 0.5$, $A = 1$, $c = 0.1$, $\alpha = 0.602$ and $\gamma = 0.101$ which are recommended by Spall when α_k and c_k has the following forms

$$\alpha_k = \frac{a}{(k+1+A)^{\alpha}}, \tag{5.8}$$

$$c_k = \frac{c}{(k+1)^{\gamma}}. \tag{5.9}$$

The parameter $\beta = \min(\bar{f}_1, 500)$ and $\delta = 0.001$ in *Algorithm 2.3* and *Algorithm 2.4*, $s = 0.001$ in *Algorithm 2.4*. We ran each algorithm for 50 times and observed the average function values after 3000 iterations. The results are listed in Table 3. Under each average function value, the approximate 90% confidence interval is also taken down. We can see more clearly the performance of the three algorithms in Figure 2. From the results, we think that *Algorithm 2.3* and *Algorithm 2.4* are also promising alternatives of the SPSA algorithm.

5.3. Implementations

Though our new frame algorithms have good numerical performance, as is typical in all stochastic algorithms, the specific implementation details are also important.

Firstly, at each iteration, block 'bad' steps if the new estimate for x fails a certain criterion(i.e., set $x_{k+1} = x_k$ in going from k to k+1) such as $\|\tilde{g}(x_{k+1})\| > \|\tilde{g}(x_k)\| + m$ or $|\tilde{f}(x_{k+1})| > |\tilde{f}(x_k)| + m$, m might be set about a large positive constant such as 10, 20, 50 etc. Experiments have shown that it can improve our algorithms' stability (This method is referred to Spall [12]).

Secondly, note the best point at every r iteration which is the point that satisfies

$$x_k^* = arg\min_x\{\|\tilde{g}_{k-r+1}\|, \|\tilde{g}_{k-r+2}\|, \ldots \|\tilde{g}_k\|\}, \tag{5.10}$$

when we use *Frame I* algorithm. When $\varphi_k = 0$, we can go back to the best point x_k^*, and then go towards the direction $-\tilde{g}_{x_k^*}$ instead of going back to x_k and towards the direction $-\tilde{g}_{x_k}$. Experiments have shown that it can improve the performance sometimes.

Thirdly, in a high noise environment, it may be desirable to compute and average several $\tilde{g}_k$ or $\tilde{f}_k$.

At last, different choice of $\tilde{d}_k$ may result in very different performance, so we should choose suitable $\tilde{d}_k$ according to different problems. At the same time, if $\tilde{d}_k$ is chosen in (2.10), different choice of γ_k can have different behavior.

6. Conclusions

In this paper, in order to accelerate the convergence of stochastic approximation algorithm, two useful frame algorithms with adaptive directions are proposed. Either of the two frame algorithms can produce different algorithms by different choice of $\tilde{d}_k$. We give suitable choice of $\tilde{d}_k$ in (2.10). We also prove the almost surely convergence property under some assumptions for the two frame algorithms. Limited numerical experiments show that they can perform better than the RM algorithm. We also propose two difference forms frame algorithms when there are only noisy function values can be used. The numerical experiments also show that for the test problem the two difference forms frame algorithms outperform SPSA algorithm. They are promising. We expect that these algorithms are useful in some other circumstances.

Acknowledgments This research was supported by the Chinese NSF grants 10571171 and 40233029. The authors are very grateful to the two referees for many useful suggestions that improve this paper.

References

[1] H. Robbins and S. Monro, *A stochastic approximation method*, Annals of Mathematical Statistics, 22 (1951), pp. 400–407.

[2] J. R. Blum, *Multidimensional stochastic approximation methods*, Annals of Mathematical Statistics, 25 (1954), pp. 737–744.

[3] J. Kiefer and J. Wolfowitz, *Stochastic estimation of the modulus of a regression function*, Annals of Mathematical Statistics, 23 (1952), pp. 462–466.

[4] H. J. Kushner and D. S. Clark, *Stochastic Approximation for Constrained and Unconstrained Systems*, Springer-Verleg, 1978.

[5] J. C. Spall, *Multivariate stochastic approximation using a simultanoues perturbation gradient approximation*, IEEE Transactions on Automatic Control, 37 (1992), pp.332–341.

[6] H. Kesten, *Accelerated stochastic approximation*, Annals of Mathematical Statistics, 29 (1958), pp.41–59.

[7] B. Delyon and A. Juditsky, *Accelerated stochastic approximation*, SIAM Journal on Optimization, 3 (1993), pp.868–881.

[8] J. C. Spall, *Introduction to Stochastic Search and Optimization*, John Wiley & sons, 2003.

[9] Y. H. Dai and Y. Yuan, *Nonlinear Conjugate Gradient Methods*, Shanghai Scientific and Technical Publishers, 2000. (in Chinese)

[10] D. P. Bertsekas and J. N. Tsitsiklis, *Gradient convergence in gradient methods with errors*, SIAM Journal on Optimization, 10 (2003), pp. 627–642.

[11] J. J. Moré, B. S. Garbow, and K. E. Hillstrom, *Testing unconstrained optimization software*, ACM Transactions on Mathematical Software, 7 (1981), pp. 17–41.

[12] J. C. Spall, *Adaptive stochastic approximation by the simultaneous perturbation method*, IEEE Transactions on Automatic Control, 45 (2000), pp.1839–1853.

[13] N. N. Schraudolph and T. Graepel, *Combining conjugate direction methods with stochastic approximation of gradients*, Proceedings of the 9th International Workshop on Artificial Intelligence and Statistics, Key West, FL, (2002), pp.7–13.