SPSA with Momentum

Chtiba Reda

2021.12.15

What is the FDSA

The Finite Difference Stochastic Approximation (FDSA) is an algorithm for optimizing systems that lack gradient information and the accessible input-output data generally depends on some noise. In this algorithm, we update the unknown parameter θ ($\theta \in \mathcal{R}^p$) of the objective(loss) function $L(\theta)$, in each iteration, by adding information from the gradient estimate $\hat{g}(\theta)$. The procedure used to estimate the gradient is the Finite-Difference Method, thus requiring 2.p function evaluations per iterations.

Recursion procedure for the algorithm

Let $\hat{\theta}_k$ be the estimate of the θ at the k-th iteration,a_k the gain sequence with positive scalar output and $\hat{g}_k(\hat{\theta}_k)$ the gradient approximation at the k-th iteration as well.

Recursion procedure for the algorithm

Let $\hat{\theta}_k$ be the estimate of the θ at the k-th iteration, a_k the gain sequence with positive scalar output and $\hat{g}_k(\hat{\theta}_k)$ the gradient approximation at the k-th iteration as well.

$$\hat{\theta}_{k+1} = \hat{\theta}_k - a_k.\hat{g}_k(\hat{\theta}_k)$$

Recursion procedure for the algorithm

Let $\hat{\theta}_k$ be the estimate of the θ at the k-th iteration, a_k the gain sequence with positive scalar output and $\hat{g}_k(\hat{\theta}_k)$ the gradient approximation at the k-th iteration as well.

$$\hat{\theta}_{k+1} = \hat{\theta}_k - a_k.\hat{g}_k(\hat{\theta}_k)$$

The gradient estimate formula using the F-D Method is the following:

Recursion procedure for the algorithm

Let $\hat{\theta}_k$ be the estimate of the θ at the k-th iteration, a_k the gain sequence with positive scalar output and $\hat{g}_k(\hat{\theta}_k)$ the gradient approximation at the k-th iteration as well.

$$\hat{\theta}_{k+1} = \hat{\theta}_k - a_k.\hat{g}_k(\hat{\theta}_k)$$

The gradient estimate formula using the F-D Method is the following:

$$\hat{g}_k(\hat{\theta}_k) = \begin{bmatrix} \dfrac{y_k(\hat{\theta}_k + c_k.\xi_1) - y_k(\hat{\theta}_k - c_k.\xi_1)}{2c_k} \\ \vdots \\ \dfrac{y_k(\hat{\theta}_k + c_k.\xi_p) - y_k(\hat{\theta}_k - c_k.\xi_p)}{2c_k} \end{bmatrix}$$

Recursion procedure for the algorithm

Let $\hat{\theta}_k$ be the estimate of the θ at the k-th iteration,a_k the gain sequence with positive scalar output and $\hat{g}_k(\hat{\theta}_k)$ the gradient approximation at the k-th iteration as well.

$$\hat{\theta}_{k+1} = \hat{\theta}_k - a_k.\hat{g}_k(\hat{\theta}_k)$$

The gradient estimate formula using the F-D Method is the following:

$$\hat{g}_k(\hat{\theta}_k) = \begin{bmatrix} \dfrac{y_k(\hat{\theta}_k + c_k.\xi_1) - y_k(\hat{\theta}_k - c_k.\xi_1)}{2c_k} \\ \vdots \\ \dfrac{y_k(\hat{\theta}_k + c_k.\xi_p) - y_k(\hat{\theta}_k - c_k.\xi_p)}{2c_k} \end{bmatrix}$$

Where y_k the noisy representation of the loss function, ξ_i is a column vector with p components, 1 in it's i-th row and o everywhere else and c_k is a gain coefficient.

What is SPSA

Similar to the FDSA, the Simultaneous Perturbation Stochastic Approximation (SPSA) is also an algorithm for optimizing systems without information on the gradient, the difference lies in the method to approximate the gradient,which is the Simultaneous Perturbation Method, and the main feature of this technique is that it only requires two measurements of the loss function, regardless of the dimension of θ

Recursion Formula for the SPSA

The SPSA has similar recursion procedure as the FDSA:

Recursion Formula for the SPSA

The SPSA has similar recursion procedure as the FDSA:

$$\hat{\theta}_{k+1} = \hat{\theta}_k - a_k.\hat{g}_k(\hat{\theta}_k)$$

Recursion Formula for the SPSA

The SPSA has similar recursion procedure as the FDSA:

$$\hat{\theta}_{k+1} = \hat{\theta}_k - a_k.\hat{g}_k(\hat{\theta}_k)$$

Where this time the gradient $\hat{g}_k(\hat{\theta}_k)$ is approximated using the SP Method, thus having the following form:

Recursion Formula for the SPSA

The SPSA has similar recursion procedure as the FDSA:

$$\hat{\theta}_{k+1} = \hat{\theta}_k - a_k.\hat{g}_k(\hat{\theta}_k)$$

Where this time the gradient $\hat{g}_k(\hat{\theta}_k)$ is approximated using the SP Method, thus having the following form:

$$\hat{g}_k(\hat{\theta}_k) = \begin{bmatrix} \dfrac{y_k(\hat{\theta}_k + c_k.\Delta_k) - y_k(\hat{\theta}_k - c_k.\Delta_k)}{2c_k.\Delta_{k_1}} \\ \vdots \\ \dfrac{y_k(\hat{\theta}_k + c_k k.\Delta_k) - y_k(\hat{\theta}_k - c_k k.\Delta_k)}{2c_k.\Delta_{k_p}} \end{bmatrix}$$

Recursion Formula for the SPSA

The SPSA has similar recursion procedure as the FDSA:

$$\hat{\theta}_{k+1} = \hat{\theta}_k - a_k.\hat{g}_k(\hat{\theta}_k)$$

Where this time the gradient $\hat{g}_k(\hat{\theta}_k)$ is approximated using the SP Method, thus having the following form:

$$\hat{g}_k(\hat{\theta}_k) = \begin{bmatrix} \dfrac{y_k(\hat{\theta}_k + c_k.\Delta_k) - y_k(\hat{\theta}_k - c_k.\Delta_k)}{2c_k.\Delta_{k_1}} \\ \vdots \\ \dfrac{y_k(\hat{\theta}_k + c_k k.\Delta_k) - y_k(\hat{\theta}_k - c_k k.\Delta_k)}{2c_k.\Delta_{k_p}} \end{bmatrix}$$

Where $\Delta_k \in \mathcal{R}^p$ is the random perturbation vector and $E(\Delta_k) = 0$ for every k

What is SPSA with Momentum

The SPSA with Momentum is an extension to the Basic SPSA, where we include the Momentum Method in the recursion form of the SPSA, in hope that additional information of the history of the algorithm, will accelerate the convergence of this enhanced SPSA.

Recursion Formula

$$\hat{\theta}_{k+1} = \hat{\theta}_k - a_k.\hat{g}_k(\hat{\theta}_k) + b.(\hat{\theta}_k - \hat{\theta}_{k-1})$$

Recursion Formula

$$\hat{\theta}_{k+1} = \hat{\theta}_k - a_k.\hat{g}_k(\hat{\theta}_k) + b.(\hat{\theta}_k - \hat{\theta}_{k-1})$$

Where $\hat{g}_k$ is the same Estimate of the gradient using the S-P Method, and b is the Momentum coefficient.

Numerical proof of convergence

In the sequel, I will present three plots, that were simulated to show convergence of the parameter of the loss function to the Optimum.

Numerical proof of convergence

In the sequel, I will present three plots, that were simulated to show convergence of the parameter of the loss function to the Optimum.First, let's give the setting that have been used in the simulation.

Numerical proof of convergence

In the sequel, I will present three plots, that were simulated to show convergence of the parameter of the loss function to the Optimum.First, let's give the setting that have been used in the simulation.We consider, a loss function $L(\theta_1, \theta_2) = \theta_1^2 + \theta_2^2$, where $\theta = [\theta_1, \theta_2]^T$,

Numerical proof of convergence

In the sequel, I will present three plots, that were simulated to show convergence of the parameter of the loss function to the Optimum.First, let's give the setting that have been used in the simulation.We consider, a loss function $L(\theta_1, \theta_2) = \theta_1^2 + \theta_2^2$, where $\theta = [\theta_1, \theta_2]^T$,the optimum is $\theta^\star = [0, 0]^T$.

Numerical proof of convergence

In the sequel, I will present three plots, that were simulated to show convergence of the parameter of the loss function to the Optimum.First, let's give the setting that have been used in the simulation.We consider, a loss function $L(\theta_1, \theta_2) = \theta_1^2 + \theta_2^2$, where $\theta = [\theta_1, \theta_2]^T$,the optimum is $\theta^\star = [0, 0]^T$.We consider the loss measurements are taken with i.i.d noise having distribution $\mathcal{N}(0, 1)$.

Numerical proof of convergence

In the sequel, I will present three plots, that were simulated to show convergence of the parameter of the loss function to the Optimum.First, let's give the setting that have been used in the simulation.We consider, a loss function $L(\theta_1, \theta_2) = \theta_1^2 + \theta_2^2$, where $\theta = [\theta_1, \theta_2]^T$,the optimum is $\theta^\star = [0, 0]^T$.We consider the loss measurements are taken with i.i.d noise having distribution $\mathcal{N}(0, 1)$.We let, $\hat{\theta_0}$=$\hat{\theta_1}$ (initial values of the parameter) to be generated randomly.

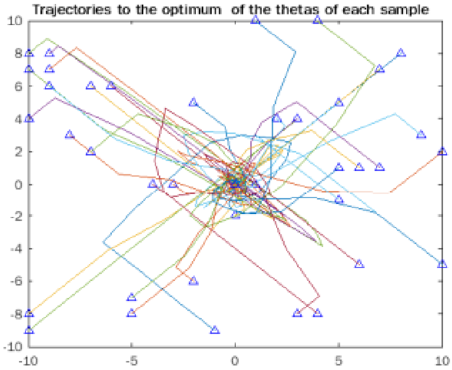
Numerical proof of convergence

In the sequel, I will present three plots, that were simulated to show convergence of the parameter of the loss function to the Optimum.First, let's give the setting that have been used in the simulation.We consider, a loss function $L(\theta_1, \theta_2) = \theta_1^2 + \theta_2^2$, where $\theta = [\theta_1, \theta_2]^T$,the optimum is $\theta^\star = [0, 0]^T$.We consider the loss measurements are taken with i.i.d noise having distribution $\mathcal{N}(0, 1)$.We let, $\hat{\theta_0}$=$\hat{\theta_1}$ (initial values of the parameter) to be generated randomly. we also choose the coefficient to be in the procedure as : A=10, c=0.05, a=0.5, $\alpha = 0.602$ and $\gamma = 0.101$.

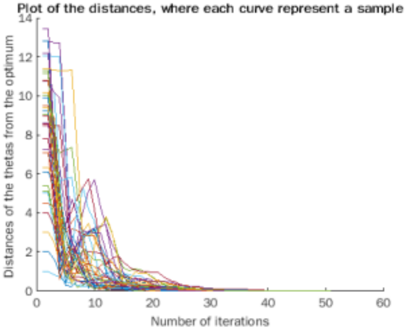
Numerical proof of convergence

In the sequel, I will present three plots, that were simulated to show convergence of the parameter of the loss function to the Optimum.First, let's give the setting that have been used in the simulation.We consider, a loss function $L(\theta_1, \theta_2) = \theta_1^2 + \theta_2^2$, where $\theta = [\theta_1, \theta_2]^T$,the optimum is $\theta^\star = [0, 0]^T$.We consider the loss measurements are taken with i.i.d noise having distribution $\mathcal{N}(0, 1)$.We let, $\hat{\theta_0}$=$\hat{\theta_1}$ (initial values of the parameter) to be generated randomly. we also choose the coefficient to be in the procedure as : A=10, c=0.05, a=0.5, $\alpha = 0.602$ and $\gamma = 0.101$.Then, we proceed to make 500 experiments each ruining 1000 iterations

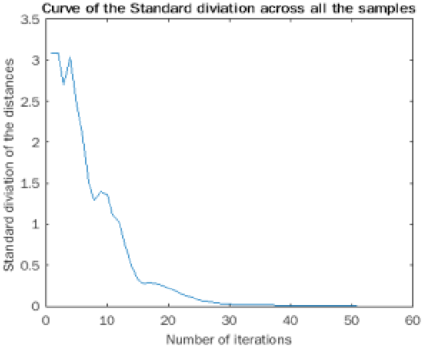
First figure



Second figure



Third figure



Numerical comparison between FDSA and SPSAM

Let's consider a similar framework of the previous implementation, only this time, we will take

Numerical comparison between FDSA and SPSAM

Let's consider a similar framework of the previous implementation, only this time, we will take $\hat{\theta}_0 = \hat{\theta}_1 = [0.1, -0.6]^T$ and

Numerical comparison between FDSA and SPSAM

Let's consider a similar framework of the previous implementation, only this time, we will take $\hat{\theta}_0 = \hat{\theta}_1 = [0.1, -0.6]^T$ and the coefficient $a = 0.3$.

Numerical comparison between FDSA and SPSAM

Let's consider a similar framework of the previous implementation, only this time, we will take $\hat{\theta}_0=\hat{\theta}_1=[0.1, -0.6]^T$ and the coefficient $a = 0.3$.We define the normalized loss $L_{\text{norm}}\,(\hat{\theta}_k)=\frac{L(\hat{\theta}_k)-L(\theta^\star)}{L(\hat{\theta}_0)-L(\theta^\star)}$, where $\hat{\theta}_k$ will represent the terminal of the iterations in each experiment.

Numerical comparison between FDSA and SPSAM

Let's consider a similar framework of the previous implementation, only this time, we will take $\hat{\theta}_0 = \hat{\theta}_1 = [0.1, -0.6]^T$ and the coefficient $a = 0.3$.We define the normalized loss $L_{\text{norm}}\,(\hat{\theta}_k) = \frac{L(\hat{\theta}_k) - L(\theta^\star)}{L(\hat{\theta}_0) - L(\theta^\star)}$, where $\hat{\theta}_k$ will represent the terminal of the iterations in each experiment. and we will present in the sequel, a table that shows, the contrast in efficiency between the SPSAM and FDSA

Table 3.1. Sample means of normalized loss $L_{\text{norm}} = L_{\text{norm}} \left(\hat{\boldsymbol{\theta}}_k\right)$ at terminal $\hat{\boldsymbol{\theta}}_k$ for FDSA and SPSAM over 50 independent replications. Number of loss measurements $y(\theta)$ is such that FDSA and SPSAM take the same number of iterations in each comparison.

Number of $y(\theta)$ values [number of iterations]	Mean L_{norm} for FDSA	Mean L_{norm} for SPSAM
200-FDSA; 100-SPSAM [50 iterations]	3.73×10^{-4}	7.95×10^{-9}
4000-FDSA; 2000-SPSAM [1000 iterations]	4.07×10^{-18}	1.4×10^{-33}

Conclusion

The analysis given previously, in the case of a quadratic function, indicates that the SPSA with Momentum is potentially more efficient than the FDSA when using the same number of iterations

THANK YOU