Averaged Stochastic Gradient Descent with Feedback: An Accurate, Robust, and Fast Training Method

Xu Sun*, Hisashi Kashima*, Takuya Matsuzaki†, and Naonori Ueda‡
*Department of Mathematical Informatics, The University of Tokyo
†Department of Computer Science, The University of Tokyo
‡NTT Communication Science Laboratories, Kyoto, Japan
{xusun, kashima}@mist.i.u-tokyo.ac.jp matuzaki@is.s.u-tokyo.ac.jp ueda@cslab.kecl.ntt.co.jp

***Abstract*—On large datasets, the popular training approach has been stochastic gradient descent (SGD). This paper proposes a modification of SGD, called averaged SGD with feedback (ASF), that significantly improves the performance (robustness, accuracy, and training speed) over the traditional SGD. The proposal is based on three simple ideas: averaging the weight vectors across SGD iterations, feeding the averaged weights back into the SGD update process, and deciding when to perform the feedback (linearly slowing down feedback). Theoretically, we demonstrate the reasonable convergence properties of the ASF. Empirically, the ASF outperforms several strong baselines in terms of accuracy, robustness over the noise, and the training speed. To our knowledge, this is the first study of "feedback" in stochastic gradient learning. Although we choose latent conditional models for verifying the ASF in this paper, the ASF is a general purpose technique just like SGD, and can be directly applied to other models.**

I. Introduction

Structured classification is a general task that encompasses many problems in data mining and other areas. Real-world problems may contain hidden structures that are difficult to be captured by conventional structured classification models without latent variables. For example, in the syntactic parsing task for natural language, the hidden structures can be refined grammars which are unobservable in the supervised training data [1]. In the gesture recognition task of the computational vision area, there are also hidden structures which are crucial for successful gesture recognition [2]. There are also plenty of hidden structure examples in other tasks among different areas [3], [4], [5]. In such cases, models that exploit latent variables are advantageous in learning.

Training latent conditional models (more formally, discriminative probabilistic latent variable models, DPLVMs) is quite challenging. Standard gradient descent methods are normally batch training methods, in which the true gradient is used to update the parameters of the model, for example, the quasi-Newton methods like Limited-memory BFGS (L-BFGS) [6]. The true gradient is usually the sum of the gradients from each individual training instance. Therefore, batch gradient descent requires the training method to go through the entire training set before updating the parameters. For this reason, the batch training methods are intolerably slow on training DPLVMs [1], [4].

A promising fast probabilistic training method is the stochastic gradient method, for example, the SGD [7], [8], [9]. In the stochastic gradient method, the true gradient is approximated by the gradient of a small set, or more aggressively, a single training example. The parameter vector is updated based on the approximate gradients. The parameters of the model are updated much more frequently, and much fewer iterations are needed before the convergence. For large scale data sets, the SGD can be faster than batch gradient based training methods.

However, there are problems on the current SGD literature: 1) The SGD is sensitive to noise. The accuracy of the SGD training is limited when the data is noisy. 2) The SGD is not robust. It contains many hyper-parameters (not only regularization, but also decaying rate) and it is quite sensitive to them. Tuning the hyper-parameters for SGD is not a easy task. 3) The speed is still not fast enough. It is time consuming to perform regularization with an online setting.

To deal with the problems of the traditional stochastic methods, we present a new stochastic gradient learning method. The proposal can significantly improve the accuracy of stochastic training by smoothening out noise. In addition, according to the experiments, the proposal is quite robust and fast for structured classification tasks in data mining.

II. Background

A. Latent Conditional Model

Given the training data, the task is to learn a mapping between a sequence of observations $\mathbf{x} = x_1, x_2, \ldots, x_m$ and a sequence of labels $\mathbf{y} = y_1, y_2, \ldots, y_m$. Each y_j is a class label for the j'th token of a word sequence, and is a member of a set $\mathbf{Y}$ of possible class labels. For each sequence, the model also assumes a sequence of latent variables $\mathbf{h} = h_1, h_2, \ldots, h_m$, which is unobservable in training examples.

1550-4786/10 $26.00 © 2010 IEEE
DOI 10.1109/ICDM.2010.26

The DPLVM model[1] is defined as follows [10]:

$$P(\mathbf{y}|\mathbf{x},\mathbf{\Theta}) \triangleq \sum_{\mathbf{h}} P(\mathbf{y}|\mathbf{h},\mathbf{x},\mathbf{\Theta})P(\mathbf{h}|\mathbf{x},\mathbf{\Theta}),$$

where $\mathbf{\Theta}$ represents the parameter vector of the model. DPLVM models can be seen as a natural extension of CRF models, and CRF models can be seen as a special case of DPLVMs that employ only one latent variable for each label.

To make the training and inference efficient, the model is restricted to have disjointed sets of latent variables associated with each class label. Each h_j is a member in a set $\mathbf{H}_{y_j}$ of possible latent variables for the class label y_j. $\mathbf{H}$ is defined as the set of all possible latent variables, i.e., the union of all $\mathbf{H}_{y_j}$ sets. Since sequences which have any $h_j \notin \mathbf{H}_{y_j}$ will by definition have $P(\mathbf{y}|h_j,\mathbf{x},\mathbf{\Theta}) = 0$, the model can be further defined as:

$$P(\mathbf{y}|\mathbf{x},\mathbf{\Theta}) \triangleq \sum_{\mathbf{h}\in\mathbf{H}_{y_1}\times\ldots\times\mathbf{H}_{y_m}} P(\mathbf{h}|\mathbf{x},\mathbf{\Theta}),$$

where $P(\mathbf{h}|\mathbf{x},\mathbf{\Theta})$ is defined by the usual conditional random field formulation:

$$P(\mathbf{h}|\mathbf{x},\mathbf{\Theta}) = \frac{\exp \mathbf{\Theta}\cdot\mathbf{f}(\mathbf{h},\mathbf{x})}{\sum_{\forall\mathbf{h}} \exp \mathbf{\Theta}\cdot\mathbf{f}(\mathbf{h},\mathbf{x})},$$

in which $\mathbf{f}(\mathbf{h},\mathbf{x})$ is a feature vector. Given a training set consisting of n labeled sequences, $(\mathbf{x}_i,\mathbf{y}_i)$, for $i = 1\ldots n$, parameter estimation is performed by optimizing the objective function. In what follows, we denote the conditional log-likelihood of each sample $\log P(\mathbf{y}_i|\mathbf{x}_i,\mathbf{\Theta})$ as $L_s(i,\mathbf{\Theta})$. Then, the objective function is (if use L_2 regularization):

$$L(\mathbf{\Theta}) = \sum_{i=1}^{n} L_s(i,\mathbf{\Theta}) - \frac{||\mathbf{\Theta}||^2}{2\sigma^2}. \qquad (1)$$

B. Stochastic Gradient Descent

The SGD uses a small randomly-selected subset of the training samples to approximate the gradient of the objective function given by Equation 1. The extreme case is a batch size of 1, and it gives the maximum frequency of updates, which we adopt in this work. The model parameters are updated in such a way:

$$\mathbf{\Theta}^{k+1} = \mathbf{\Theta}^{k} + \gamma_k \frac{\partial}{\partial\mathbf{\Theta}}(L_s(i,\mathbf{\Theta}) - \frac{||\mathbf{\Theta}||^2}{2n\sigma^2}),$$

where k is the update counter and γ_k is the learning decaying rate. A typical convergent choice of learning rate can be found in [11]:

$$\gamma_k = \frac{\gamma_0}{1+k/n},$$

where γ_0 is a constant. This scheduling guarantees ultimate convergence [9]. In this paper we adopt this decaying schedule for the SGD.

```
Notes $m$ is the number of periods when the ASF reaches the convergence;
$b$ is the current number of period;
$c$ is the current number of iteration;
$n$ is the number of training samples;
The decaying rate, $\gamma \longleftarrow \frac{\gamma_0}{1+b/Z}$, is only for theoretical analysis. In practice we can simply set $\gamma \leftarrow 1$, i.e., remove the decaying rate.

Procedure ASF-train
    Initialize $\mathbf{\Theta}$ with random values
    $c \longleftarrow 0$
    for $b \longleftarrow 1$ to $m$
    .   $\gamma \longleftarrow \frac{\gamma_0}{1+b/Z}$ with $Z \gg n$, or simply $\gamma \leftarrow 1$
    .   for 1 to $b$
    .       $\mathbf{\Theta} \longleftarrow$ SGD-update($\mathbf{\Theta}$)
    .   $c \longleftarrow c + b$
    .   $\mathbf{\Theta} \longleftarrow \overline{\mathbf{\Theta}}^{iter(c)}$ in Eq. 2
    Return $\mathbf{\Theta}$

Procedure SGD-update($\mathbf{\Theta}$)
    for 1 to $n$
    .   select a sample $j$ randomly
    .   $\mathbf{\Theta} \longleftarrow \mathbf{\Theta} + \gamma\frac{\partial}{\partial\mathbf{\Theta}}L_s(j,\mathbf{\Theta})$
    Return $\mathbf{\Theta}$
```

Figure 1. The major steps of the ASF training.

III. Proposal: Averaged SGD with Feedback

We will present the averaged SGD, and more importantly, we will show that a reasonable feedback schedule is the key to make the averaged SGD being robust and accurate.

The naive version of averaged SGD is inspired by the averaged perceptron technique [12]. Let $\mathbf{\Theta}^{iter(c),sample(d)}$ be the parameters after the d'th training example has been processed in the c'th iteration over the training data. We define the *averaged parameters* at the end of the iteration c' as:

$$\overline{\mathbf{\Theta}}^{iter(c')} \triangleq \frac{\sum_{c=1\ldots c',\ d=1\ldots n} \mathbf{\Theta}^{iter(c),sample(d)}}{nc'}. \qquad (2)$$

However, a straightforward application of parameter averaging is not adequate. A potential problem of traditional parameter averaging is that the model parameters $\mathbf{\Theta}$ receive no information from the averaged parameters: the model parameters $\mathbf{\Theta}$ are trained exactly the same like before (SGD without averaging). $\mathbf{\Theta}$ could be misleading as the training goes on. To solve this problem, a natural idea is to reset $\mathbf{\Theta}$ by using the averaged parameters, which are more reliable. We propose a refined version of averaged SGD by further applying a *"periodic feedback"*.

We periodically reset the parameters $\mathbf{\Theta}$ by using the averaged parameters $\overline{\mathbf{\Theta}}$. The interval between a feedback

[1] The implementation source code of DPLVMs and CRFs is available at http://www.ibis.t.u-tokyo.ac.jp/XuSun

operation and its previous operation is called a *training period* or simply a *period*. It is important to decide when to do the feedback, i.e., the length of each period should be adjusted reasonably as the training goes on. For example, at the early stage of the training, the $\mathbf{\Theta}$ is highly noisy, so that the feedback operation to $\mathbf{\Theta}$ should be performed more frequently. As the training goes on, less frequent feedback operation would be better in order to adequately optimize the parameters. In practice, we adopt a schedule of *linearly slowing-down feedback*, and we will show the reasonable convergence properties of this scheduling in Section III-A.

In what follows, we call the proposal as *averaged SGD with feedback (ASF)*[2]. Figure 1 shows the steps of the ASF. Now, we analyze the averaged parameters produced by each period. We denote $\mathbf{\Theta}^{b,c,d}$ as the model parameters after the d'th sample is processed in the c'th iteration of the b'th period. Without making any difference, we denote $\mathbf{\Theta}^{b,c,d}$ more simply as $\mathbf{\Theta}^{b,cn+d}$ where n is the number of samples in a training data. Similarly, we use $g^{b,cn+d}$ to denote $\frac{\partial}{\partial\mathbf{\Theta}}L_s(d,\mathbf{\Theta})$ in the c'th iteration of the b'th period. Let $\gamma^{(b)}$ be the decaying rate in the b'th period. Let $\overline{\mathbf{\Theta}}^{(b)}$ be the averaged parameters produced by the b'th period. We can induce the explicit form of $\overline{\mathbf{\Theta}}^{(1)}$:

$$\overline{\mathbf{\Theta}}^{(1)} = \mathbf{\Theta}^{1,0} + \gamma^{(1)} \sum_{d=1\ldots n} \frac{n-d+1}{n} g^{1,d}. \quad (3)$$

When the 2nd period ends, the parameters are again averaged over all previous model parameters, $\mathbf{\Theta}^{1,0},\ldots,\mathbf{\Theta}^{1,n},\mathbf{\Theta}^{2,0},\ldots,\mathbf{\Theta}^{2,2n}$, and it can be expressed as:

$$\begin{aligned}\overline{\mathbf{\Theta}}^{(2)} = \mathbf{\Theta}^{1,0} &+ \gamma^{(1)} \sum_{d=1\ldots n} \frac{n-d+1}{n} g^{1,d} \\ &+ \gamma^{(2)} \sum_{d=1\ldots 2n} \frac{2n-d+1}{3n} g^{2,d}.\end{aligned} \quad (4)$$

Similarly, the averaged parameters produced by the b'th period can be expressed as follows:

$$\overline{\mathbf{\Theta}}^{(b)} = \mathbf{\Theta}^{1,0} + \sum_{i=1\ldots b} (\gamma^{(i)} \sum_{d=1\ldots in} \frac{in-d+1}{ni(i+1)/2} g^{i,d}). \quad (5)$$

The procedure of deriving Eq.5 is sketched in Appendix.

While it is strongly recommended for LBFGS and SGD to perform regularization, the ASF does not strongly rely on regularization. A possible reason is that the averaging performs as an "implicit regularization" for improving the generality. For this reason, we currently only perform simple lazy-regularization on ASF, i.e., do the regularization after the ASF training is done. We find the removing the lazy-regularization do not undermine the performance of ASF: the ASF without regularization still outperforms the SGD with tuned regularizer.

A. Convergence Analysis

The best possible convergence result for stochastic learning is the *"almost sure convergence"*: to prove that the stochastic algorithm converges towards the solution with probability 1 [9]. We will show that the proposed method guarantees to achieve *almost sure convergence*. We first introduce the general convexity assumption: everywhere in the parameter space, the opposite of the gradient must point toward a unique minimum $\mathbf{\Theta}^*$. Such a strong assumption is only valid for a few simple learning algorithms. Nevertheless, the assumption usually holds within the final convergence region because the cost function is locally convex in many practical applications.

If a stochastic update is convergent, it means that either the gradients or the learning rates must vanish near the optimum [13]. According to [13], it is reasonable to assume that the variance of the stochastic gradient does not grow faster than the norm of the real gradient itself. Also, it is reasonable to assume that $||\frac{\partial}{\partial\mathbf{\Theta}}L(\mathbf{\Theta})||^2$ behaves quadratically within the final convergence region. Both assumptions are conveniently expressed as follows:

$$\mathbf{E}_i[\frac{\partial}{\partial\mathbf{\Theta}}L_s(i,\mathbf{\Theta})^2] < A + B(\mathbf{\Theta}-\mathbf{\Theta}^*)^2, \quad (6)$$

where $A \geq 0$ and $B \geq 0$. Based on the assumptions, the *convergence theorem* has been given [13]: two conditions on the learning rate are sufficient conditions for the *almost sure convergence* of the SGD to the optimum $\mathbf{\Theta}^*$. The two conditions on the learning rate are as follows [13]:

$$\sum \gamma_k = \infty \quad and \quad \sum \gamma_k^2 < \infty. \quad (7)$$

Too fast decaying rate may make the SGD fail to reach the optimum if it is far away, while too slow decaying rate may make the SGD keep oscillating around.

With those preparations, we have the convergence theorem for the ASF:

Theorem 1: *Let the optimization procedure be defined in Figure 1. Given the convex assumption and the assumption Eq. 6, the averaged parameters produced at the end of each period of the optimization procedure are "almost surely convergent" towards the optimum* $\mathbf{\Theta}^*$.

The proof of Theorem 1 is sketched in Appendix. As we discussed before, although the convex assumption is a strong one, in practice the assumption usually holds within the final convergence region for non-convex cost functions. From another point of view, for non-convex cost functions, the convergence is probably not a big issue for practitioners because normally the training has to be terminated at a certain number of iterations in practice [7].

IV. Experiments and Discussion

We choose two real-world structured classification tasks for our experiments: biomedical named entity recognition, and sensor-based action recognition.

[2] The implementation source code will be available online at http://www.ibis.t.u-tokyo.ac.jp/XuSun

Table IV

RESULTS IN THE ACTION RECOGNITION TASK (AVERAGED OVER THREE REPEATED EXPERIMENTS).

Methods	Accuracy (%)	Iterations	Time
DPLVM, ASF (**)	**70.9 (0.1)**	**20**	1 hour
DPLVM, Averag. SGD (*)	69.6 (0.1)	60	4 hours
DPLVM, SGD	61.3 (1.1)	35	2 hours
DPLVM, LBFGS	68.9 (3.7)	400	14 hours
CRF, SGD	63.2 (0.9)	35	1 hour
CRF, LBFGS	66.6 (0.4)	400	6 hours

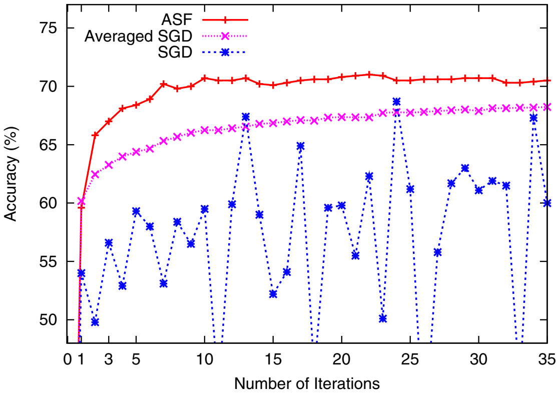

Figure 3. Curves of token accuracies while varying the number of iterations on the sensor-based action recognition task: the proposal method vs. the SGD. This figure comes from one of the repeated experiments, which have similar tendencies.

quence). A time-window contains 4 values: {*time (the seconds past from the beginning of a session), x-axis-acceleration, y-axis-acceleration, z-axis-acceleration*}, for example, {539.266(s), 0.091(g), -0.145(g), -1.051(g)}[3]. There are three kinds of action labels: `label-1` means *"Walking/Running"*, `label-2` means *"On Vehicle (Taking Train/Bus/Car/Bicycle)"* and `label-3` means *"Others"*. We split this data into a training data (85%), a development data for hyper-parameters (5%), and the final evaluation data (10%). The evaluation metric are token-accuracy (%). The features are listed in Table III. The remaining experimental settings are very similar to the biomedical named entity recognition task, therefore we will not repeat the description for the space.

The experimental results are listed in Table IV. As we can see, similar to the previous task, the DPLVM-ASF significantly outperforms its latent conditional baselines and traditional baselines, on both the accuracy and the speed.

In Figure 3, we show the curves of token accuracies on varying the number of training iterations of the ASF and the SGD. As can be seen, the ASF training is much more stable/robust than the SGD training. The fluctuation of the SGD is more drastic than the previous task, probably due to the more noisy data. The robustness of the ASF method is closely related to the stable nature of averaging with feedback.

V. CONCLUSIONS, DISCUSSION, AND FUTURE WORK

The ASF significantly improves the performance (robustness, accuracy, and training speed) over the traditional SGD. The ASF is based on three simple ideas: averaging the weights, feedback, and a heuristic on deciding when to perform the feedback. It is important to linearly slowing down the feedback, because it performs much better than other alternative settings (not reported for space). Further study on this direction is interesting.

The design of decaying rate and lazy regularization of the ASF is quite different from traditional SGD. A study on them can be a future work. For faster speed or simplicity, the decaying rate and the lazy regularization can be removed from ASF. The ASF without decaying rate and regularization will still work well enough in general. On the other hand, the SGD without decaying rate or regularization will suffer performance loss considerably.

If the objective function of the ASF and the SGD is the same and the function is convex (e.g., for CRFs), it is then supposed that they will arrive the same optimum. However, our extra experiments show that the ASF outperforms the SGD considerably. The reason is unknown, and further analysis is a future work. This is also not very surprising. Similar phenomenon happens between the SGD and the LBFGS with the same objective function: they are supposed to achieve the same optimum after the convergence, but in fact their performance can be quite different.

In practice, it is important to choose a good weight vector by using development data. If the evaluation on development data was performed after each period of ASF, it may neglect good weight vector during the iterations of a period, because there could be many iterations in a period. Therefore, it is recommended to evaluate the averaged weights after each iteration rather than after each period.

ACKNOWLEDGMENT

X.S., H.K., and N.U. were supported by the FIRST Program of Japan Society for Promotion of Science.

REFERENCES

[1] S. Petrov and D. Klein, "Discriminative log-linear grammars with latent variables," in *Advances in Neural Information Processing Systems 20 (NIPS)*, 2008, pp. 1153–1160.

[2] S. B. Wang, A. Quattoni, L.-P. Morency, D. Demirdjian, and T. Darrell, "Hidden conditional random fields for gesture recognition," in *Proceedings of CVPR'06*. IEEE Computer Society, 2006, pp. 1521–1527.

[3] X. Sun, N. Okazaki, and J. Tsujii, "Robust approach to abbreviating terms: A discriminative latent variable model with global information," in *Proceedings of the ACL'09*, Suntec, Singapore, August 2009, pp. 905–913.

[3] In the example, 'g' is the *acceleration of gravity*.

[4] X. Sun, T. Matsuzaki, D. Okanohara, and J. Tsujii, "Latent variable perceptron algorithm for structured classification," in *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI 2009)*, 2009, pp. 1236–1242.

[5] S. Petrov, "Products of random latent variable grammars," in *Proceedings of NAACL'10*, 2010, to appear.

[6] J. Nocedal and S. J. Wright, "Numerical optimization," *Springer*, 1999.

[7] Y. Tsuruoka, J. Tsujii, and S. Ananiadou, "Stochastic gradient descent training for l1-regularized log-linear models with cumulative penalty," in *Proceedings of ACL'09*, Suntec, Singapore, August 2009, pp. 477–485.

[8] S. Shalev-Shwartz, Y. Singer, and N. Srebro, "Pegasos: Primal estimated sub-gradient solver for svm," in *Proceedings of ICML'07*, 2007.

[9] L. Bottou, "Online algorithms and stochastic approximations," *Online Learning and Neural Networks. Saad, David. Cambridge University Press*, 1998.

[10] L.-P. Morency, A. Quattoni, and T. Darrell, "Latent-dynamic discriminative models for continuous gesture recognition," in *Proceedings of CVPR'07*, 2007, pp. 1–8.

[11] M. Collins, A. Globerson, T. Koo, X. Carreras, and P. L. Bartlett, "Exponentiated gradient algorithms for conditional random fields and max-margin markov networks," *J. Mach. Learn. Res. (JMLR)*, vol. 9, pp. 1775–1822, 2008.

[12] M. Collins, "Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms," in *Proceedings of EMNLP'02*, 2002, pp. 1–8.

[13] L. Bottou, "Stochastic learning," *Advanced Lectures on Machine Learning*, pp. 146–168, 2004.

[14] J.-D. Kim, T. Ohta, Y. Tsuruoka, and Y. Tateisi, "Introduction to the bio-entity recognition task at JNLPBA," in *Proceedings of BioNLP'04*, 2004, pp. 70–75.

[15] T. Huynh, M. Fritz, and B. Schiele, "Discovery of activity patterns using topic models," in *Proceedings of the 10th international conference on Ubiquitous computing*. ACM, 2008, pp. 10–19.

[16] Y. Hattori, M. Takemori, S. Inoue, G. Hirakawa, and O. Sudo, "Operation and baseline assessment of large scale activity gathering system by mobile device," in *Proceedings of DICOMO'10*, 2010.

Appendix

Derivation of Eq. 5:

We follow the denotations of $\boldsymbol{\Theta}^{b,cn+d}$, $g^{b,cn+d}$, $\gamma^{(b)}$ and $\overline{\boldsymbol{\Theta}}^{(b)}$ defined in Section III. For the 1st period, the parameters after each update is as follows:

$$\begin{aligned}
&\boldsymbol{\Theta}^{1,1} = \boldsymbol{\Theta}^{1,0} + \gamma^{(1)}g^{1,1},\\
&\boldsymbol{\Theta}^{1,2} = \boldsymbol{\Theta}^{1,1} + \gamma^{(1)}g^{1,2} = \boldsymbol{\Theta}^{1,0} + \gamma^{(1)}g^{1,1} + \gamma^{(1)}g^{1,2},\\
&\ldots\\
&\boldsymbol{\Theta}^{1,n} = \boldsymbol{\Theta}^{1,n-1} + \gamma^{(1)}g^{1,n}\\
&\qquad = \boldsymbol{\Theta}^{1,0} + \gamma^{(1)}g^{1,1} + \gamma^{(1)}g^{1,2} + \cdots + \gamma^{(1)}g^{1,n}.
\end{aligned}$$

At the end of the 1st period, the parameters are averaged as follows:

$$\overline{\boldsymbol{\Theta}}^{(1)} = \frac{\sum_{d=1\ldots n}\boldsymbol{\Theta}^{1,d}}{n} = \boldsymbol{\Theta}^{1,0} + \gamma^{(1)}\sum_{d=1\ldots n}\frac{n-d+1}{n}g^{1,d}.$$

At the beginning of the 2nd period, we do the feedback: $\boldsymbol{\Theta}^{2,0} \longleftarrow \overline{\boldsymbol{\Theta}}^{(1)}$. When the 2nd period ends, the parameters are again averaged over all previous model parameters, $\boldsymbol{\Theta}^{1,0}, \ldots, \boldsymbol{\Theta}^{1,n}, \boldsymbol{\Theta}^{2,0}, \ldots, \boldsymbol{\Theta}^{2,2n}$:

$$\begin{aligned}
\overline{\boldsymbol{\Theta}}^{(2)} &= \frac{\sum_{d=1\ldots n}\boldsymbol{\Theta}^{1,d} + \sum_{d=1\ldots 2n}\boldsymbol{\Theta}^{2,d}}{n+2n}\\
&= \frac{n\overline{\boldsymbol{\Theta}}^{(1)} + \sum_{d=1\ldots 2n}\boldsymbol{\Theta}^{2,d}}{3n}\\
&= \frac{n\overline{\boldsymbol{\Theta}}^{(1)} + [2n\overline{\boldsymbol{\Theta}}^{(1)} + \gamma^{(2)}\sum_{d=1\ldots 2n}(2n-d+1)g^{2,d}]}{3n}\\
&= \boldsymbol{\Theta}^{1,0} + \gamma^{(1)}\sum_{d=1\ldots n}\frac{n-d+1}{n}g^{1,d}\\
&\quad + \gamma^{(2)}\sum_{d=1\ldots 2n}\frac{2n-d+1}{3n}g^{2,d}.
\end{aligned}$$

In a similar way, it is straightforward to conclude that:

$$\overline{\boldsymbol{\Theta}}^{(b)} = \boldsymbol{\Theta}^{1,0} + \sum_{i=1\ldots b}(\gamma^{(i)}\sum_{d=1\ldots ni}\frac{ni-d+1}{ni(i+1)/2}g^{i,d}). \qquad (8)$$

Proof of theorem 1:

In Eq. 8, notice that the $\overline{\boldsymbol{\Theta}}^{(b)}$ can be explicitly expressed as the form $\overline{\boldsymbol{\Theta}}^{(b)} = \boldsymbol{\Theta}^{1,0} + \overline{\gamma}_1 g_1 + \overline{\gamma}_2 g_2 + \cdots + \overline{\gamma}_i g_i$, where $g_1 \ldots g_i$ represents the respective gradients in the right side of Eq. 8 and $\overline{\gamma}_1 \ldots \overline{\gamma}_i$ are the corresponding factors of those gradients.

We then prove that the decaying rates $\overline{\gamma}_1 \ldots \overline{\gamma}_i$ satisfy the two conditions: $\sum\overline{\gamma}_i = \infty$ and $\sum\overline{\gamma}_i^2 < \infty$ (see Eq. 7).

$$\begin{aligned}
\lim_{b\to\infty}\sum\overline{\gamma}_i &= \lim_{b\to\infty}\sum_{i=1\ldots b}(\gamma^{(i)}\sum_{d=1\ldots ni}\frac{ni-d+1}{ni(i+1)/2})\\
&= \lim_{b\to\infty}\sum_{i=1\ldots b}(\frac{\gamma_0}{1+i/Z}\sum_{d=1\ldots ni}\frac{ni-d+1}{ni(i+1)/2})\\
&= \lim_{b\to\infty}\sum_{i=1\ldots b}\frac{\sum_{d=1\ldots ni}(ni-d+1)}{ni^2(i+1)/2}\\
&= \lim_{b\to\infty}\sum_{i=1\ldots b}\frac{1}{i} = \infty
\end{aligned}$$

Similarly,

$$\begin{aligned}
\lim_{b\to\infty}\sum\overline{\gamma}_i^2 &= \lim_{b\to\infty}\sum_{i=1\ldots b}[(\frac{\gamma_0}{1+i/Z})^2\sum_{d=1\ldots ni}(\frac{ni-d+1}{ni(i+1)/2})^2]\\
&= \lim_{b\to\infty}\sum_{i=1\ldots b}\frac{1}{i^3} < \infty
\end{aligned}$$

Then, applying those two conditions into the *convergence theorem* (see Section III-A) completes the proof.

□