

The Early Restart Algorithm

Malik Magdon-Ismail

Amir F. Atiya

Learning Systems Group, Electrical Engineering Department, California Institute of Technology, Pasadena, CA 91125, U.S.A.

Consider an algorithm whose time to convergence is unknown (because of some random element in the algorithm, such as a random initial weight choice for neural network training). Consider the following strategy. Run the algorithm for a specific time T . If it has not converged by time T , cut the run short and rerun it from the start (repeat the same strategy for every run). This so-called restart mechanism has been proposed by Fahlman (1988) in the context of backpropagation training. It is advantageous in problems that are prone to local minima or when there is a large variability in convergence time from run to run, and may lead to a speed-up in such cases. In this article, we analyze theoretically the restart mechanism, and obtain conditions on the probability density of the convergence time for which restart will improve the expected convergence time. We also derive the optimal restart time. We apply the derived formulas to several cases, including steepest-descent algorithms.

1 Introduction

One of the difficulties encountered when training multilayer networks using the backpropagation method, or any other training method, is the slow convergence and the uncertainty in the length of the training time. There is a large variability in the convergence time from run to run, since this time depends strongly on the initial conditions, which are generated randomly. To avoid such an uncertainty, Fahlman (1988) proposed the restart mechanism. In this mechanism, a training algorithm is cut short if it has not converged after a certain number of iterations and is restarted with new random weights; the hope is that this "fresh start" will find its way faster to the solution. The restart mechanism has also been applied in a number of other optimization applications (e.g., Ghannadian & Alford, 1996; Hu, Shonkwiler, & Spruill, 1997).

The restart algorithm can reduce the overall average and standard deviation of the training time, since it avoids waiting for excessively long runs. In addition, the restart algorithm will become even more indispensable if there are local minima. If for such a situation we do not restart, the training algorithm could spend forever approaching an unacceptable local minimum

and never reach the global minimum. In this article we present a theoretical analysis of the restart algorithm and derive conditions under which restart becomes advantageous. The results we develop indicate that whenever an algorithm displays fat-tailed behavior, multimodality, or potential lack of convergence, restart can be useful and, in fact, absolutely necessary when there is a nonzero probability of not converging. Our goal is to gain some insight into the value of restart by providing such conditions and a general framework for understanding when restart can be helpful. The analysis developed applies to any type of algorithm; it is not restricted to training algorithms in the neural network field.

2 Analysis of the Restart Algorithm

Because the weights are generated randomly, the training time to convergence, t , will obey a certain probability density, $p(t)$. In our case, converging to a local minimum is not considered convergence; thus, the density $p(t)$ might have a delta function at ∞ , which represents the probability that the algorithm does not converge at all. Let $\mathcal{T} = E[t]$ be the expected convergence time for the algorithm. Let T be the restart time, and let $\mathcal{T}_{RES}(T) = E_{RES}[t]$ be the expected convergence time for the algorithm when applying restart. We will denote the probability that we restart by α_T :

$$\alpha_T = \text{probability of restart} = 1 - \int_0^T p(t)dt. \tag{2.1}$$

The optimal restart time, T^* , can be obtained by finding the value of T that minimizes $\mathcal{T}_{RES}(T)$. The following theorem gives $\mathcal{T}_{RES}(T)$ in terms of $p(t)$ and T .

Theorem 1. *The optimal time to restart is the time T^* that minimizes the expression*

$$\mathcal{T}_{RES}(T) = \frac{\alpha_T}{1 - \alpha_T} T + \int_0^T t \frac{p(t)}{1 - \alpha_T} dt \tag{2.2}$$

with respect to T .

Proof. The expectation of the convergence time can be derived as follows (let T be the restart time):

$$\mathcal{T}_{RES}(T) = \int_0^T tp(t)dt + \alpha_T (\mathcal{T}_{RES}(T) + T). \tag{2.3}$$

The first term represents the contribution due to the case that the algorithm converged before the restart time. The second term represents the case that

restart occurred. Note that the term in parentheses is due to the fact that if restart occurs, the expected convergence time equals the time already spent until restart ($\equiv T$), plus the expectation of another application of the restart algorithm ($\equiv \mathcal{T}_{RES}(T)$). Of course, the term in parentheses is weighted by the probability that restart occurs (α_T).

Solving for $\mathcal{T}_{RES}(T)$ in equation 2.3, we obtain

$$\mathcal{T}_{RES}(T) = \frac{\alpha_T}{1 - \alpha_T} T + \int_0^T t \frac{p(t)}{1 - \alpha_T} dt. \tag{2.4}$$

To obtain the optimal restart time, we need to obtain the minimum of equation 2.4 with respect to T . This completes the proof.

The optimal restart time can be obtained by taking the derivative of equation 2.2 and equating to zero. We get

$$\frac{1 - \alpha_{T^*}}{p(T^*)} = T^* + \frac{\int_0^{T^*} tp(t)dt}{\alpha_{T^*}} \tag{2.5}$$

as the condition that has to be satisfied by the optimal restart time.

Note. Equation 2.2 for $\mathcal{T}_{RES}(T)$ can also be viewed from a different light, which will give additional insight. The term $\frac{\alpha_T}{1 - \alpha_T}$ equals the infinite series $\alpha_T + \alpha_T^2 + \dots$ (α_T^i represents restarting i times). Hence this series represents the expected number of restarts (which is evaluated as $\sum_{i=1}^{\infty} i(\alpha_T^i - \alpha_T^{i+1})$ and hence equal $\alpha_T + \alpha_T^2 + \dots$). This term is multiplied by the length of the restart time, to obtain the mean amount of time spent restarting. The second term represents the mean time until convergence given that no restart will occur. The division of $p(t)$ by $1 - \alpha_T$ is a normalization factor, so that the probability density in the expectation (in the second term) represents the density for the training time given that no restart will occur.

Note. Several studies have proposed methods for estimating $p(t)$ for back-propagation networks (e.g., Leen & Moody, 1993; Orr & Leen, 1993; Heskes, Slijpen & Kappen, 1992). These estimates of $p(t)$ can be used to estimate T according to equation 2.2.

The real value of restart is when the algorithm in question is prone to local minima. Then, restarting can be interpreted as repeated attempts to reach the global minimum. For such a case, $p(t)$ will have a delta function at ∞ . Let

$$p(t) = p_0(t) + P_{\infty}\delta(t - \infty). \tag{2.6}$$

We obtain the following theorem.

Theorem 2. For the problem considered above with $0 < P_\infty < 1$, it is always advantageous to restart, i.e. $\exists T$ such that $T_{RES}(T) < T$.

Proof. It is easy to see that $T = \infty$, because of the delta function at ∞ . Now consider $T_{RES}(T)$. It is always possible to choose T such that it is finite and such that $\int_0^T p(t)dt \neq 0$, because of the condition $P_\infty \neq 1$. From equation 2.2, since the numerator is finite and the denominator is nonzero, $T_{RES}(T)$ is finite and hence strictly less than T .

It is interesting to note that P_∞ does not appear explicitly in the expression for the optimal restart time. However, it affects the restart in an indirect way, by affecting the integrals of $p(t)$ and $tp(t)$, because the density integrates to 1.

3 Application of the Theory

To illustrate the analysis, we have applied the theory to a number of examples.

3.1 Steepest-Descent Algorithms. Here, we consider steepest- and gradient-descent algorithms and investigate the effect of restart on their expected convergence time. We first derive an expression for the density of the convergence time $p(t)$ for general steepest-descent algorithms. Consider for simplicity the error function associated with a linear hypothesis function,

$$E = \sum_{i=1}^N \left(w^T x_i - y_i \right)^2, \tag{3.1}$$

where x_i 's are the input examples, y_i 's are the target outputs, and w is the weight vector. For nonlinear hypothesis functions, the error surface is approximately quadratic close to the minimum. Thus, the analysis derived will be approximately valid for the nonlinear case if we assume we start relatively close to the minimum.

For simplicity consider the case of a three-dimensional weight vector. The more general N -dimensional case, though algebraically more cumbersome, can be treated using identical techniques. Suitably normalizing the x_i 's, the probability $P[k]$ of stopping at iteration k can be derived (see the appendix). For the case where the weight initialization is around the true weight vector, $P[k]$ is given by equation A.9. Figure 1a shows the probability distribution $P[k]$ as given in equation A.9, together with the probability distribution as estimated from a numerical simulation of the steepest-descent algorithm. The figure shows the accurate agreement between theory and simulation (in this linear case). Figure 1b shows $T_{RES}(k)$ as a function of k , calculated from equation 2.2. For this case, we can see that the optimal restart time is $T = 2$.

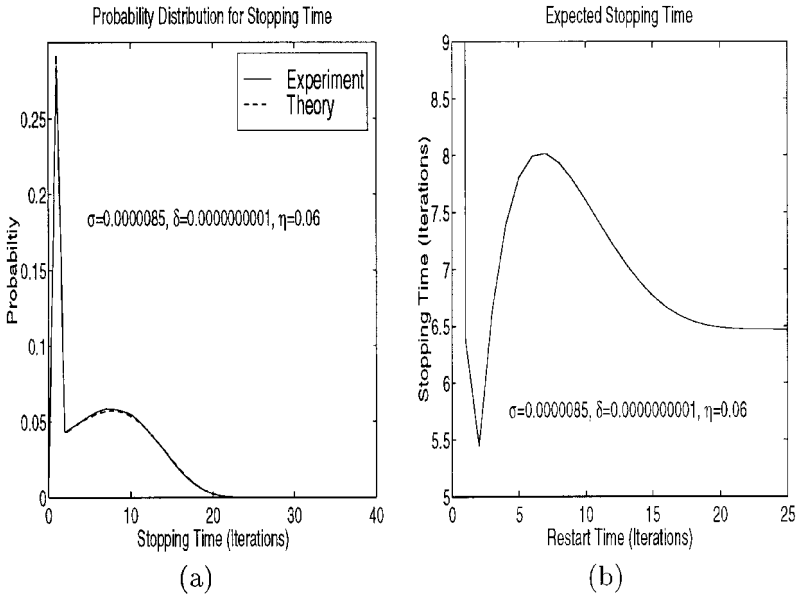


Figure 1: Steepest descent. (a) The experimental and theoretical probability of stopping at iteration k for the specific choices of the parameters σ, δ, η as shown. (b) $T_{RES}(k)$ as a function of k obtained using the theoretical $P(k)$.

Example: XOR problem. We have applied the restart method to the back-propagation algorithm. We chose the XOR problem using a 2-2-1 network, a widely studied benchmark. It is well known that the XOR problem with the 2-2-1 network possesses no local minima (see Sprinkhuizen-Kuyper & Boers, 1996, 1998; Hamey, 1995, 1998). First, we have estimated the probability density of the convergence time $p(t)$, by generating the initial weights according to a gaussian density with $\Sigma = \sigma^2 I$, and repeatedly training the network (with learning rate η) until the prespecified tolerance for the error function (δ) is achieved. We have implemented over 150,000 training runs. The resulting estimated density is as shown in Figure 2a. As is well known, steepest descent exhibits a large tail, so in spite of the fact that convergence is expected at 1000 iterations, many training runs did not converge until after 3000 iterations. Applying the restart formula, equation 2.2, we obtain the expected convergence time as a function of the restart time T . This relationship is shown in Figure 2b. One can see that restart improves convergence speed. In particular, the optimal restart time is at about 950 iterations.

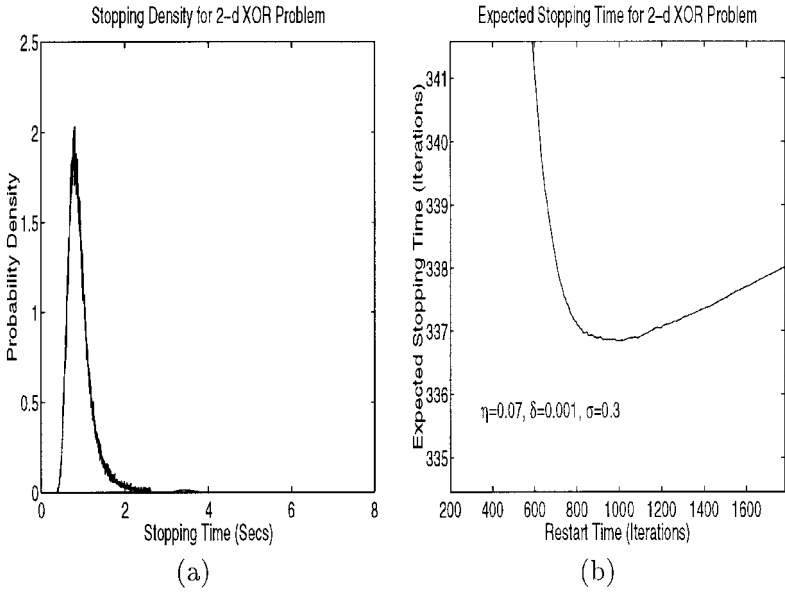


Figure 2: Two-dimensional XOR problem. (a) The experimental probability of stopping at time t for the specific choices of the parameters σ, δ, η as shown. (b) $T_{RES}(k)$ as a function of k obtained using the density in a.

3.2 A Sum of Gamma Functions Density. As another example, we consider a more sophisticated density: the sum of two gamma functions (see Figure 3a). Many algorithms exhibit several peaks in their convergence time density. In multilayer training, this can happen in particular if the minimum is between a steep and a flat surface. Also, in other optimization applications, such as maximization of likelihood functions, this same situation is frequently encountered because of the nature of the problem (see Magdon-Ismail & Abu-Mostafa, 1998). Figure 3b shows the expected convergence time as a function of the restart time. The best restart time is $T^* \approx 6$ in this example. We can see the importance of choosing the right restart time for such a case; choosing T slightly smaller or larger would be detrimental.

4 Conclusion

In this study we have analyzed the restart mechanism. We have shown that it is often faster on the average to stop a running algorithm and restart it with new initial conditions. The restart algorithm is essential when there are local minima, because it reduces the expected convergence time from ∞ to

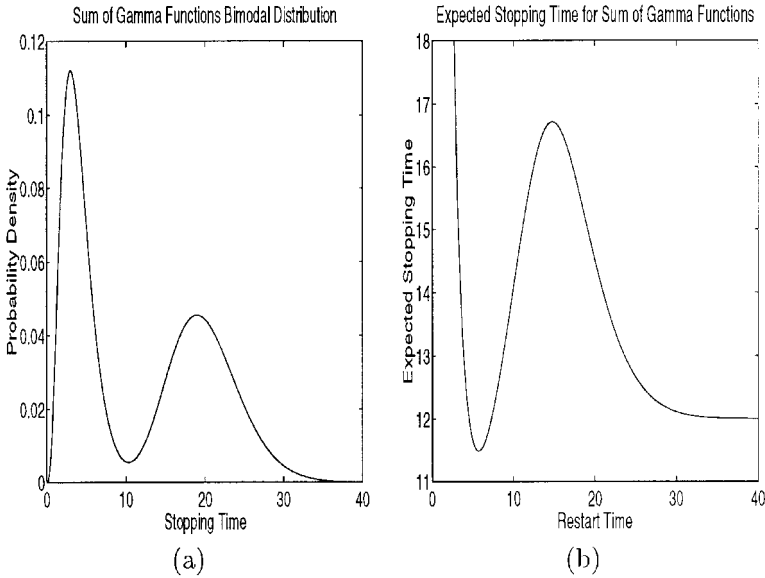


Figure 3: Sum of two gamma functions. (a) The probability of stopping at time t . (b) $T_{RES}(T)$ as a function of T .

a finite number. However, we have shown in the simulations that restart is also advantageous in some applications where no local minima exist, partly because of the large tail behavior or the multimodality for the density of the convergence time. We have also derived the optimal restart time in an attempt to understand the phenomenon. Although the convergence time density is often unknown (except for algorithm developers, who can afford to run the algorithm many times), the formula derived will give guidance as to how to choose a good estimate of the restart time. As an example, if it is known that the minimum is between a steep and a flat surface, then the density is multimodal, and an estimate of the modes will lead to a better estimate of the restart time.

Appendix A: Distribution of the Number of Iterations for Gradient Descent on the Squared Error with Linear Models

Suppose the input vectors are $\{x_i\}_{i=1}^N$. Corresponding to each input is the output y_i , the i th element in the column vector y . Let the input matrix X be

given by $\mathbf{X} = [\mathbf{x}_1 \ \mathbf{x}_2 \ \cdots \ \mathbf{x}_N]$. The error E can be put in the following form:

$$E = \sum_{i=1}^N (w^T \mathbf{x}_i - y_i)^2 = w^T \mathbf{X} \mathbf{X}^T w - 2w^T \mathbf{X} \mathbf{y} + \mathbf{y}^T \mathbf{y}. \tag{A.1}$$

We use the gradient-descent algorithm:

$$w(k+1) = w(k) - \frac{1}{2} \eta \frac{\partial E}{\partial w(k)}, \tag{A.2}$$

with $\partial E / \partial w(k) = 2(\mathbf{X} \mathbf{X}^T w - \mathbf{X} \mathbf{y})$. Assuming without loss of generality that $\mathbf{X} \mathbf{X}^T = \mathbf{I}$, we find that E depends on the iteration number as

$$E(k) = \|w - w^*\|^2 = (1 - \eta)^{2k} \|w(0) - w^*\|^2. \tag{A.3}$$

We suppose the condition for stopping is that $E \leq \delta$. Letting $P[k]$ denote the probability of stopping at iteration k , we have that

$$P[k] = \mathcal{P} \left[\frac{\sqrt{\delta}}{(1 - \eta)^{k-1}} \leq \|w(0) - w^*\| < \frac{\sqrt{\delta}}{(1 - \eta)^k} \right]. \tag{A.4}$$

Thus, we need the distribution of $\|w(0) - w^*\|$.

We will restrict ourselves to a three-dimensional weight space. The more general case, though algebraically more complex, is treated with exactly the same techniques. Suppose that $w(0)$ has a normal distribution with mean 0 and variance σ^2 . Let $F(\rho) = \mathcal{P}[\|w(0) - w^*\| \leq \rho]$:

$$F(\rho) = \frac{1}{(2\pi \sigma^2)^{3/2}} \int_{\|w-w^*\|^2 \leq \rho^2} d^3w e^{-\frac{w^2}{2\sigma^2}}. \tag{A.5}$$

Transforming to spherical coordinates with the orientation such that the z -axis is aligned with w^* , doing the ϕ integration, and making the substitution $v = \epsilon \cos \theta$, we have

$$F(\rho) = \frac{2\pi}{(2\pi \sigma^2)^{3/2}} \int_0^\infty dw \int_{-1}^1 dv w^2 e^{-\frac{w^2}{2\sigma^2}}. \tag{A.6}$$

$w^2 - 2ww^*v \leq \rho^2 - w^{*2}$

One needs to consider separately the cases $w^* \geq \rho$ and $w^* \leq \rho$. Then, differentiating the expression for $F(\rho)$, one obtains the density. After considerable calculations, one then finds that

$$f(\rho) = \frac{2}{w^*} \rho \sinh \left(\frac{\rho}{\sigma} \right) \frac{e^{-\frac{w^{*2} + \rho^2}{2\sigma^2}}}{\sqrt{2\pi \sigma^2}}. \tag{A.7}$$

Let $\lambda_1(k) = \sqrt{\delta}/(1 - \eta)^{k-1}$ for $k > 0$ and $\lambda_1 = 0$ for $k = 0$. Let $\lambda_2(k) = \sqrt{\delta}/(1 - \eta)^k$. Then, from equation A.4 we see that

$$P[k] = \int_{\lambda_1(k)}^{\lambda_2(k)} d\rho f(\rho). \tag{A.8}$$

When w^* is close to 0, this expression reduces to

$$P[k] = \left[\operatorname{erf} \left(\frac{\lambda_2}{\sqrt{2}\sigma} \right) - \operatorname{erf} \left(\frac{\lambda_1}{\sqrt{2}\sigma} \right) \right] + \frac{2}{\sqrt{2\pi}\sigma^2} \left[\lambda_1 e^{-\frac{\lambda_1^2}{2\sigma^2}} - \lambda_2 e^{-\frac{\lambda_2^2}{2\sigma^2}} \right], \tag{A.9}$$

where $\operatorname{erf}(x) = 2/\sqrt{\pi} \int_0^x dt \exp(-t^2)$. The case where $w(0)$ is distributed normally about the true weight vector is also equivalent to the case $w^* \rightarrow 0$.

Acknowledgments

We thank Yaser Abu-Mostafa and the Caltech Learning Systems Group for their useful input. We also acknowledge the support of NSF’s Engineering Research Center at Caltech.

References

Fahlman, S. (1988). *An empirical study of learning speed in back-propagation networks* (CMU Tech. Rep. No. CMU-CS-88-162). Pittsburgh: Carnegie Mellon University.

Ghannadian, F., & Alford, C. (1996). Application of random restart to genetic algorithms. *Intelligent Systems*, 95, 81–102.

Hamey, L. (1995). *Analysis of the error surface of the XOR network with two hidden nodes* (Computing Rep. No. 96/167C). Department of Computing, Macquarie University.

Hamey, L. (1998). XOR has no local minima: A case study in neural network error surface analysis. *Neural Networks*, 11, 669–681.

Heskes, T., Slijpen, E., & Kappen, B. (1992). Learning in neural networks with local minima. *Physical Review A*, 46(8): 5221–5231.

Hu, X., Shonkwiler, R., & Spruill, M. (1997). Random restart in global optimization. Preprint, 1192-015, School of Mathematics, Georgia Institute of Technology.

Leen, T., & Moody, J. (1993). Weight space probability densities in stochastic learning: I. Dynamics and equilibria. In C. L. Giles, S. J. Hanson, & J. D. Cowan (Eds.), *Advances in neural information processing systems*, (pp. 451–458). San Mateo, CA: Morgan Kaufmann.

Magdon-Ismail, M., & Abu-Mostafa, Y. (1998). Validation of volatility models. *Journal of Forecasting*, 17, 349–368.

- Orr, G., & Leen, T. (1993). Weight space probability densities in stochastic learning: II. Transients and basin hopping times. In C. L. Giles, S. J. Hanson, & J. D. Cowan (Eds.), *Advances in neural information processing systems*, (pp. 507–514). San Mateo, CA: Morgan Kaufmann.
- Sprinkhuizen-Kuyper, I., & Boers, E. (1996). The error surface of the simplest XOR network has only global minima. *Neural Computation*, *8*, 1301–1320.
- Sprinkhuizen-Kuyper, I., & Boers, E. (1998). The error surface of the 2-2-1 XOR: The finite stationary points. *Neural Networks*, *11*, 683–690.

Received October 2, 1998; accepted April 13, 1999.