On-line Learning for Very Large Datasets

Apprentissage Stochastique pour Très Grands Echantillons

Léon Bottou & Yann Le Cun
NEC Research Institute,
4 Independence Way,
Princeton NJ 08540, USA
Email: {leonb,yann}@nec-labs.com

Abstract: La conception de très grand systèmes d'apprentissage pose un grand nombre de problèmes non résolus. Savons nous, par exemple, construire un algorithme qui "regarde" la télévision pendant quelques semaines et apprend à énumérer les objets présents dans ces images. Les lois d'échelles de nos algorithmes ne nous permettent pas de traiter les quantités massives de données que cela implique. L'expérience suggère que les algorithmes les mieux adaptés sont les algorithmes stochastiques. Leur convergence est pourtant réputée beaucoup plus lente que celle des meilleurs algorithmes d'optimisation. Mais il s'agit de la convergence vers l'optimum empirique. Notre papier reformule la question en termes de convergence vers le point de meilleure généralisation et montre la superiorité d'un algorithme stochastique bien conçu.

Keywords: Learning, Convergence speed, Online learning, Stochastic optimisation.

1 Introduction

During the last decade, we have seen a considerable improvement in our theoretical understanding of learning systems as statistical machines. The Vapnik Chervonenkis theory (Vapnik, 1974) has spelled out the role of capacity and generalization in the design of learning algorithms. This understanding has influenced several recent learning algorithms such as support vector machines (Boser, Guyon and Vapnik, 1992) and boosting (Drucker, Schapire and Simard, 1993), (Freund and Schapire, 1996). These algorithms challenge the popular *curse of dimension-*

ality which says that statistical systems with large number of parameters require impracticably large data sets.

Bridges have been established between learning algorithms and both classical and Bayesian statistics. We have seen learning algorithms applied to problems usually associated with statistics. We also have seen a massive application of statistics to solve learning problems or improve learning algorithms.

Despite these advances, we have yet to see a spectacular increase in the size of both the data sets and the learning machines. The MNIST data set (Bottou et al., 1994), for instance, is still described as a relevant benchmark for more recent algorithms. Systems dealing with more than a few millions examples seldom compute more than simple counts or histograms. Do we know how to build a machine which can learn how to enumerate objects in arbitrary scenes using TV broadcasts as a data source?

The MNIST experiments were carried out on workstations featuring a 40MHz processor and 32MB of memory. The computer hardware was a clear bottleneck. Personal computers now feature fifty times that speed and memory. Hard disk technology has progressed even faster. Large data sources are now available, because audio and video capture hardware is now commonplace, and also because the development of on-line technologies has provided abundant transaction logs.

This discrepancy indicates that we have reached another bottleneck. Our learning algorithms do not scale well enough to take advantage of such large datasets and such large computing resources.

As datasets grow to practically infinite sizes, we argue that *on-line* algorithms (Bottou and Murata, 2002) outperform traditional learning algorithms that operate by repetitively sweeping over the entire training set. This paper shows that performing a single epoch of a suitable on-line algorithm converges to the "true" solution of the learning problem asymptotically as fast as any other algorithm.

The first part of the paper presents the problem and discusses the main results and their consequences. The second part of the paper provides proofs and mathematical details.

2 On-line Learning and Batch Learning

Many learning algorithms optimize an empirical cost function $C(\theta)$ that can be expressed as a very large sum of terms $L(z, \theta)$. Each term measures the cost associated with running a model with parameter vector θ on independently drawn examples[1] z_i.

$$C_n(\theta) \;\stackrel{\triangle}{=}\; \frac{1}{n}\sum_{i=1}^{n}\; L(z_i,\theta) \tag{1}$$

Two kinds of optimization procedures are often mentioned in connection with this problem:

- *Batch* gradient: Parameter updates are performed on the basis of the gradient and Hessian information accumulated over the entire training set:

$$\begin{aligned}\theta(k) &= \theta(k-1) - \Phi_k \frac{\partial C_n}{\partial\theta}(\theta(k-1)) \\ &= \theta(k-1) - \frac{1}{n}\,\Phi_k \sum_{i=1}^{n}\frac{\partial L}{\partial\theta}(z_i,\theta(k-1))\end{aligned} \tag{2}$$

  where Φ_k is an appropriately chosen positive definite symmetric matrix.

- *On-line* or *stochastic* gradient: Parameter updates are performed on the basis of a single sample z_t picked randomly at each iteration:

$$\theta(t) \;=\; \theta(t-1) - \frac{1}{t}\,\Phi_t \frac{\partial L}{\partial\theta}(z_t,\theta(t-1)) \tag{3}$$

  where Φ_t is again an appropriately chosen positive definite symmetric matrix. Very often the examples z_t are chosen by cycling over a randomly permuted training set. Each cycle is called an *epoch*. This paper however considers situations where the supply of training samples is practically unlimited. Each iteration of the online algorithm utilizes a fresh sample, unlikely to have been presented to the system before.

Simple batch algorithms converge linearly to the optimum θ_n^* of the empirical cost, that is to say $(\theta(k)-\theta_n^*)^2$ converges like e^{-t}). Careful choices of Φ_k make the convergence super-linear (e.g. like $1/e^{e^t}$ in favorable cases (Dennis and Schnabel, 1983).

By comparison, on-line algorithms appear to converge very slowly. This convergence has been studied extensively (Benveniste, Metivier and Priouret, 1990; Bottou, 1998). Under mild assumptions, they are shown to converge almost surely to a local minimum[2] of the cost. However, whereas on-line algorithms may converge to the general area of the optimum at least as fast as batch algorithms (Le Cun et al., 1998b), stochastic fluctuations due to the noisy gradient estimate make

[1] Each example z_t typically is an input/output pair (x_i, y_i).

[2] We assume in this paper that the parameters θ are confined in the neighborhood of a single minimum.

the parameter vector randomly wobble around the optimum in a region whose size decreases slowly. The analysis shows that $\mathbf{E}(\theta(t)-\theta_n^*)^2$ converges like $1/t$ at best.

At first glance, on-line algorithms seem hopelessly slow. However, the above discussion addresses the convergence toward the minimum of the *empirical cost* $C_n(\theta)$, whereas one should be much more interested in the convergence toward the minimum of the expected cost:

$$C_\infty(\theta) = \int L(z,\theta)\, p(z)\, \mathrm{d}z \tag{4}$$

where $p(z)$ is the unknown probability distribution from which the samples are drawn (Vapnik, 1974).

The main point of this paper is to show that, in situations where the supply of training samples is essentially unlimited, a well designed on-line algorithm converges toward the minimum of the *expected cost* just as fast as any batch algorithm. In those situations, the convergence speed is mainly limited by the fact that some informative examples have not yet been seen rather than by the fact that the examples already seen have not been fully exploited by the minimization process.

This point is very significant because on-line algorithms are considerably easier to implement. Each iteration of the batch algorithm (2) involves a large summation over all the available examples. Memory must be allocated to hold these examples, and computations must be performed on each of them. On the other hand, each iteration of the on-line algorithm (3) only involves one random example which can be discarded afterward.

3 Learning Speed

Assume we have immediate access to an infinite number of examples $(z_1 \dots z_t \dots)$ independently drawn from $p(z)$. We must decide how to use our computer cycles:

- We can run an on-line learning algorithm (3) and visit as many examples as possible during the allowed computer time. This procedure produces a sequence of parameter vectors $\theta(t)$ with $t = 1, \dots, T$ where T represent the total number of iterations achieved within the imparted time.

- We can run a batch super-linear algorithm (2) on a subset of examples examples $\{z_1, \dots, z_N\}$ where N is the size of the largest subset of examples that can be processed within the imparted time. This procedure accurately produces the parameter vector θ_N^* that minimizes the empirical cost $C_N(\theta)$.

The number of examples N processed by the batch algorithm cannot be as large as the number of examples T processed by the on-line algorithm. Comparing the complexity of equations (2) and (3) clearly shows that this would only allow for a couple iterations of the batch algorithm. Even if we assume that $N = T$, we show in this contribution that no batch algorithm can perform better than a well designed on-line algorithm.

3.1 On-line algorithm

The mathematics of on-line learning algorithm easily extend to the minimization of the expected cost. Examples z_t are drawn at each iteration of an on-line algorithm. When these examples are drawn from a set of n examples, the on-line algorithm minimizes the empirical error C_n. When these examples are drawn from the asymptotic distribution $p(z)$, it minimizes the expected cost C_∞.

Because the supply of training samples is practically unlimited, each iteration of the on-line update rule (3) utilizes a fresh example. These fresh examples then follow the asymptotic distribution. The parameter vectors $\theta(t)$ thus converge to the optimum θ^* of the expected cost C_∞. Furthermore, $\mathbf{E}(\theta(t) - \theta^*)^2$ converges like $1/t$ at best.

3.2 Batch algorithm

How fast does θ^*_N converge to the optimum θ^* of the expected cost $C_\infty(\theta)$?

A first hint is provided by the well known Cramer-Rao bound. In the Maximum Likelihood case[3], the bound suggests that $\mathbf{E}(\theta^*_n - \theta^*)^2$ converges no faster than $\mathcal{O}(1/t)$.

We consider now the sequence of solutions θ^*_n computed by a batch learning algorithm running on a set of n examples $(z_1, \ldots, z_n)$. Our first result (section A.3) provides the following recursive relation between θ^*_n and θ^*_{n-1}.

$$\theta^*_n \;=\; \theta^*_{n-1} - \frac{1}{n}\Psi_n \frac{\partial L}{\partial \theta}(z_n, \theta^*_{n-1}) + \mathcal{O}\left(\frac{1}{n^2}\right) \tag{5}$$

with

$$\Psi_n \;\triangleq\; \left(\frac{1}{n}\sum_{i=1}^{n} \frac{\partial^2}{\partial\theta\partial\theta} L(z_i, \theta^*_{n-1})\right)^{-1} \;\underset{t\to\infty}{\longrightarrow}\; \left(\frac{\partial^2}{\partial\theta\partial\theta} C_\infty(\theta^*)\right)^{-1} \tag{6}$$

This relation (5) describes the θ^*_n sequence as a recursive stochastic process that is essentially similar to the on-line learning algorithm (3). Each iteration of this "algorithm" consists in picking a fresh example z_n and updating the parameters according to (5). This is not a practical algorithm because we have no

[3] i.e. $L(z, \theta) = -\log \phi(z, \theta)$ with both conditions $\int \phi(z, \theta)\,\mathrm{d}z = 1$ and $\phi(z, \theta^*) = p(z)$.

analytical expression for the second order term. We can however apply the mathematics of on-line learning algorithms to this stochastic process. The similarity between (5) and (3) can be enhanced by an appropriate choice of the positive definite symmetric matrix Φ_t in the on-line algorithm (3).

3.3 Convergence speed result

Therefore, the convergence of the following stochastic process describes both the *convergence of an online learning algorithm* and the *behavior of the solutions of a batch learning algorithm.*

$$\theta_t \ = \ \theta_{t-1} - \frac{1}{t}\Phi_t \frac{\partial L}{\partial \theta}(z_t, \theta_{t-1}) + \mathcal{O}\left(\frac{1}{t^2}\right) \tag{7}$$

Because a same stochastic process describes both convergences, we can hope that they occur at identical speeds. It is therefore important to determine how the convergence speed of (7) depends on the unspecified second order terms and on the choice of Φ_t.

Our main result (section A.4) characterizes the convergence speed of this stochastic process under the following assumptions:

i) We only consider the final convergence phase (Bottou and Murata, 2002). More precisely we consider that the θ_t are confined in a bounded domain where $C_\infty(\theta)$ has a single minimum θ^*.

ii) We assume that Φ_t converges to the inverse of the hessian $\mathcal{H}$ of the expected risk at the optimum.

$$\Phi_t \longrightarrow \ \mathcal{H}^{-1} \quad \text{with} \quad \mathcal{H} = \frac{\partial^2}{\partial\theta\partial\theta} C_\infty(\theta^*)$$

iii) We first assume that Φ_t only depends on $z_1, \ldots, z_{t-1}$. The result however still holds when Φ_t depends mildly on z_t as in equation (6).

This convergence speed neither depends on the second order terms in our stochastic process nor depends on how fast Φ_t converges to $\mathcal{H}^{-1}$. More precisely we have:

$$\mathbf{E}\left((\theta_t - \theta^*)^2\right) = \frac{\mathbf{tr}\left(\mathcal{H}^{-1}\,\mathcal{G}\,\mathcal{H}^{-1}\right)}{t} + \mathrm{o}\left(\frac{1}{t}\right) \tag{8}$$

where $\mathbf{tr}\,(\cdot)$ represents the trace of a matrix, $\mathcal{H}$ is the hessian of the expected risk in θ^*, and $\mathcal{G}$ is a Gauss-Newton approximation of the hessian (Le Cun et al., 1998b):

$$\mathcal{G} = \mathbf{E}\left(\left[\frac{\partial L}{\partial \theta}(\mathbf{z}, \theta^*)\right]\left[\frac{\partial L}{\partial \theta}(\mathbf{z}, \theta^*)\right]'\right)$$

In the Maximum Likelihood case, it is well known that both $\mathcal{H}$ and $\mathcal{G}$ are equal to the Fisher information matrix on the optimum. Equation (8) then indicates that the convergence speed reaches the Cramer-Rao bound. Such a result was already reported in the case of the *Natural Gradient* algorithm (Murata and Amari, 1999). Our result extends Murata's result to vast classes of on-line learning algorithms beyond Natural Gradient.

3.4 Discussion

This result has implications for our initial dilemma. Should we visit as many examples as possible with a well designed on-line algorithm, or run a batch algorithm on the largest subset of examples we can afford ?

The surprisingly simple answer is to use the *algorithm that uses the most examples*. Learning is mainly limited by the fact that some informative examples have not yet been seen rather than by the fact that the examples already seen have not been fully exploited by the minimization process.

As discussed above, the higher complexity of the batch algorithm update (2) implies that the on-line algorithm can process more examples[4].

This result holds for any on-line algorithm where Φ_t converges to the inverse of the Hessian of the cost function. The speed of this convergence is not critical. It is however essential that Φ_t be a *full rank* approximation of the inverse hessian. Maintaining such a full rank approximation in a large system is very costly. This is probably the main justification for avoiding the otherwise elegant Natural Gradient algorithm.

It is therefore important to design new approximations of the inverse hessian that simultaneously are cost effective and still deliver near Cramer-Rao efficiency. We hope to achieve such a result using the new insights provided by the mathematical tools underlying the results presented in this paper.

4 Conclusion

We have shown that learning very large data sets is best achieved using a well designed on-line learning procedure. A well designed on-line learning algorithm learns just as fast as any batch algorithm *in terms of the number of examples*. Furthermore, on-line learning algorithms require less computing resources per example, and therefore are able to process more examples for a given amount of computing resources.

[4] Section A.5 shows that $T = \mathcal{O}(N \log \log N)$ where T is the number of examples visited by the on-line algorithm and N is the maximum set processed by a super-linear algorithm with the same computing resources.

A Mathematical discussion

A.1 Orders of magnitude

The main discussion uses the well known notations $\mathrm{o}\,(.)$ and $\mathcal{O}\,(.)$ without much analysis. In the case of stochastic sequences, these notations can have several distinct definitions. Let us first recall the definitions for non stochastic sequences f_t and g_t:

$$\begin{aligned} f_t = \mathrm{o}\,(g_t) \quad &\Leftrightarrow \quad \forall c,\ \exists t_0,\ \forall t > t_0,\ |f_t| < c\,|g_t| \\ f_t = \mathcal{O}\,(g_t) \quad &\Leftrightarrow \quad \exists c,\ \exists t_0,\ \forall t > t_0,\ |f_t| < c\,|g_t| \end{aligned}$$

Let $F_t(\omega)$ and $G_t(\omega)$ be two stochastic sequences. The parameter represents the elementary random variables. In the case of stochastic learning algorithms, for instance, ω represents the initial conditions and the sequence of observed examples $z_1, \ldots, z_t$. Although it is customary to simply write F_t or G_t, it is sometimes useful to make the ω parameter explicit.

We use the unmodified $\mathrm{o}\,(.)$ and $\mathcal{O}\,(.)$ notations to represent *pointwise orders of magnitude*. It means that the above definitions apply for each particular ω and that t_0 depends on ω.

Definition 1. *Pointwise orders of magnitude.*

$$\begin{aligned} F_t = \mathrm{o}\,(G_t) \quad &\Leftrightarrow \quad \forall \omega,\ \forall c,\ \exists t_0,\ \forall t > t_0,\ |F_t(\omega)| < c\,|G_t(\omega)| \\ F_t = \mathcal{O}\,(G_t) \quad &\Leftrightarrow \quad \forall \omega,\ \exists c,\ \exists t_0,\ \forall t > t_0,\ |F_t(\omega)| < c\,|G_t(\omega)| \end{aligned}$$

The above definitions is poorly adequate for deriving probabilistic results. We do not need the inequality to be true for absolutely all ω. Nothing bad happens if the inequality is violated on a set with zero probability. On the other hand it is often desirable to make t_0 independent from ω. This motivates the following definition.

Definition 2. *Almost uniform order of magnitudes.*

$$\begin{aligned} F_t = \mathrm{o}_u\,(G_t) \quad &\Leftrightarrow \quad \forall c,\ \exists t_0,\ \forall t > t_0,\ \mathbf{P}\left\{\,|F_t(\omega)| < c\,|G_t(\omega)|\,\right\} = 1 \\ F_t = \mathcal{O}_u\,(G_t) \quad &\Leftrightarrow \quad \exists c,\ \exists t_0,\ \forall t > t_0,\ \mathbf{P}\left\{\,|F_t(\omega)| < c\,|G_t(\omega)|\,\right\} = 1 \end{aligned}$$

It is more practical however to make a slight modification of the above definition, in the spirit of the concept of *convergence in probability*.

Definition 3. *Stochastic order of magnitudes (Mann and Wald, 1943).*

$$\begin{aligned} F_t = \mathrm{o}_p\,(G_t) \quad &\Leftrightarrow \quad \forall \varepsilon,\ \forall c,\ \exists t_0,\ \forall t > t_0,\ \mathbf{P}\left\{\,|F_t(\omega)| < c\,|G_t(\omega)|\,\right\} > 1 - \varepsilon \\ F_t = \mathcal{O}_p\,(G_t) \quad &\Leftrightarrow \quad \forall \varepsilon,\ \exists c,\ \exists t_0,\ \forall t > t_0,\ \mathbf{P}\left\{\,|F_t(\omega)| < c\,|G_t(\omega)|\,\right\} > 1 - \varepsilon \end{aligned}$$

Most of the properties of the usual orders of magnitude also apply to stochastic orders of magnitude. In particular it is known that $F_t = \mathrm{o}(G_t)$ implies $F_t = \mathrm{o}_p(G_t)$. On the other hand the relation $\mathbf{E}(\mathrm{o}_p(g_t)) = \mathrm{o}(g_t)$ is not true in general. This is why we introduce yet another concept:

Definition 4. *Almost uniformly bounded stochastic order of magnitude.*

$$F_t = \mathrm{o}_s(G_t) \;\Leftrightarrow\; F_t = \mathrm{o}_p(G_t) \text{ and } F_t = \mathcal{O}_u(G_t)$$

Theorem 1. *With the above definitions*

$$\mathbf{E}(\mathrm{o}_s(g_t)) = \mathrm{o}(g_t)$$

Proof. Let us write $F_t(\omega) = \mathrm{o}_s(g_t)$. Since $F_t(\omega) = \mathcal{O}_u(g_t)$, there is a constant M and a subscript t_1 such that

$$\forall t > t_1,\ \mathbf{P}\{\, |F_t(\omega)| < M\, |g_t|\, \} = 1$$

Let us choose an arbitrary positive number c. We define the event R_t as follows:

$$R_t \;\triangleq\; \left\{\omega \,:\, |F_t(\omega)| \geq \frac{c}{2}\, |g_t|\right\}$$

Since $F_t(\omega) = \mathrm{o}_p(g_t)$ there is a subscript t_2 such that

$$\forall t > t_2,\ \mathbf{P}(R_t) \leq \frac{c}{2M}.$$

Therefore, for all $t > \max(t_1, t_2)$, we can write

$$\mathbf{E}(F_t(\omega)) < (1 - \mathbf{P}(R_t))c\, |g_t| + \mathbf{P}(R_t)M\, |g_t| < \left(\frac{c}{2} + \frac{c}{2M}M\right)\, |g_t| = c\, |g_t|$$

This proves the theorem. □

A.2 Problem setup

Let the loss function $L(z, \theta)$ measure how much a learning system with parameter θ fails to handle example z. The unknown example distribution $p(\mathbf{z})$ represents the ground truth. Our goal is to minimize the expected risk $C_\infty(\theta)$.

$$C_\infty(\theta) \;\triangleq\; \mathbf{E}(L(\mathbf{z}, \theta)) \;\triangleq\; \int L(\mathbf{z}, \theta)\, \mathrm{d}p(\mathbf{z})$$

To achieve this goal, it is common to collect a finite training set $z_1, \ldots, z_n$ and to minimize the objective function $C_n(\theta)$.

$$C_n(\theta) \;\triangleq\; \frac{1}{n} \sum_{i=1}^{n} L(z_i, \theta)$$

Online learning algorithms provide another way to achieve this goal. Each iteration of a typical online algorithm consists of drawing a random example $\mathbf{z}$ and applying the following parameter update rule, where the Φ_t are well chosen positive definite symmetric matrices.

$$\theta_t \;=\; \theta_{t-1} - \frac{1}{t}\Phi_t \frac{\partial L}{\partial \theta}(z_t, \theta_{t-1})$$

We assume that functions $L(z,\theta)$, $C_n(\theta)$, and $C_\infty(\theta)$ are three times differentiable with continuous derivatives. We also assume that both the examples $\mathbf{z}$, the parameters θ, and the matrices Φ_t are uniformly bounded. These assumptions imply that many dependent variables are uniformly bounded because they are continuous functions of uniformly bounded variables. This rather strong assumption is supported by experience. Unbounded online algorithms tend to diverge and be useless.

Our discussion addresses the final convergence phase of online learning algorithms. Therefore we further assume that the parameters θ remain confined in a bounded domain $\mathcal{D}$ where the cost function $C_\infty(\theta)$ is convex and has a single non degenerate minimum $\theta^* \in \mathcal{D}$.

Notation $\mathcal{H}$ denotes the definite positive hessian of the expected cost in θ^*.

$$\mathcal{H} = \mathcal{H}(\theta^*) = \mathbf{E}\left(\frac{\partial^2}{\partial\theta\partial\theta} L(\mathbf{z}, \theta^*)\right)$$

Notation $\mathcal{G}$ denotes the expectation of the squared Jacobian of the loss function at the optimum. This matrix measures the noise introduced by the stochastic selection of the examples. It is also related to the Gauss-Newton approximation (Le Cun et al., 1998b). In the Maximum Likelihood case, it is equal to the well known Fisher Information matrix (see (Murata and Amari, 1999) for a definition).

$$\mathcal{G} = \mathbf{E}\left(\left[\frac{\partial L}{\partial \theta}(\mathbf{z}, \theta^*)\right]\left[\frac{\partial L}{\partial \theta}(\mathbf{z}, \theta^*)\right]'\right)$$

A.3 Recursive formulation of the batch algoritms

The result discussed in section 3.2 addresses the minima θ^*_n of the empirical objective functions $C_n(\theta)$. We must assume that the empirical costs $C_n(\theta)$ are convex and have a minimum on domain $\mathcal{D}$.

We define the empirical hessians

$$\hat{H}_n \;\triangleq\; \frac{1}{n}\sum_{i=1}^{n} \frac{\partial^2}{\partial\theta\partial\theta} L(z_i, \theta^*_{n-1}) \tag{9}$$

and further assume that the eigenvalues of the empirical hessians are bounded by some constants $\lambda_{\max} > \lambda_{\min} > 0$ with probability one[5]. This implies that the hessians and their inverses are all $\mathcal{O}_u(1)$.

Theorem 2. *With the above assumptions and notations,*

$$\theta^*_n \;=\; \theta^*_{n-1} - \frac{1}{n}\hat{H}_n^{-1}\frac{\partial L}{\partial \theta}(z_n, \theta^*_{n-1}) + \mathcal{O}_u\left(\frac{1}{n^2}\right)$$

Proof. Let us define

$$S_n(\theta) = \frac{1}{n}\sum_{i=1}^{n}\frac{\partial L}{\partial \theta}(z_i, \theta)$$

and write a first order expansion in point θ^*_{n-1}

$$S_n(\theta^*_n) - S_n(\theta^*_{n-1}) = \hat{H}_n\,\left(\theta^*_n - \theta^*_{n-1}\right) + \mathcal{O}_u\left((\theta^*_n - \theta^*_{n-1})^2\right).$$

where we use a uniform order of magnitude because the boundedness assumptions described in section A.2 mean that the second derivative of S_n is uniformly bounded. Since $S_n(\theta^*_n) = S_{n-1}(\theta^*_{n-1}) = 0$, we can then rewrite the left hand side of this equality.

$$-\frac{1}{n}\frac{\partial L}{\partial \theta}(z_n, \theta^*_{n-1}) = \hat{H}_n\,\left(\theta^*_n - \theta^*_{n-1}\right) + \mathcal{O}_u\left((\theta^*_n - \theta^*_{n-1})^2\right) \tag{10}$$

We can then transform the right hand side as

$$-\frac{1}{n}\frac{\partial L}{\partial \theta}(z_n, \theta^*_{n-1}) = \left(\hat{H}_n + \mathcal{O}_u\left(\theta^*_n - \theta^*_{n-1}\right)\right)\left(\theta^*_n - \theta^*_{n-1}\right)$$

and write

$$\theta^*_n - \theta^*_{n-1} = -\frac{1}{n}\left(\hat{H}_n^{-1} + \mathcal{O}\left(\theta^*_n - \theta^*_{n-1}\right)\right)\frac{\partial L}{\partial \theta}(z_n, \theta^*_{n-1})$$

Thanks to our pervasive boundedness assumptions, the above equality implies that $\theta^*_n - \theta^*_{n-1} = \mathcal{O}_u(1/n)$. We can then rewrite equation (10) as

$$-\frac{1}{n}\frac{\partial L}{\partial \theta}(z_n, \theta^*_{n-1}) = \hat{H}_n\,\left(\theta^*_n - \theta^*_{n-1}\right) + \mathcal{O}_u\left(\frac{1}{n^2}\right).$$

and derive the theorem. □

[5] This assumption is not very satisfying. We could consider that it is true only with some probability $1-\eta$. The results would then hold with probability $1-\eta$ as well.

A.4 Convergence speed

Section 3 defines a stochastic process that simultaneously describes (a) the dynamics of an online learning algorithm, and (b) the convergence of the solutions of a batch learning algorithm running on training sets of increasing sizes. The following theorem addresses the convergence speed of this stochastic process when the scaling matrices Φ_t converge to $\mathcal{H}^{-1}$ in probability.

Theorem 3. *We consider the following stochastic process*

$$\theta_t \;=\; \theta_{t-1} - \frac{1}{t}\Phi_t \frac{\partial L}{\partial \theta}(z_t, \theta_{t-1}) + \mathcal{O}_u\left(\frac{1}{t^2}\right) \tag{11}$$

with the assumptions described in section A.2 and where

i.) $\Phi_t = \mathcal{H}^{-1} + \mathrm{o}_p\,(1)$.

ii.) Φ_t *is a function of* $(\mathbf{z}_1, \ldots, \mathbf{z}_{t-1})$ *only.*

We have then

$$\mathbf{E}\left(|\theta_t - \theta^*|^2\right) \;=\; \frac{\mathbf{tr}\left(\mathcal{H}^{-1}\,\mathcal{G}\,\mathcal{H}^{-1}\right)}{t} \;+\; \mathrm{o}\left(\frac{1}{t}\right)$$

The proof relies on the following technical lemma.

Lemma 1. *: Let the positive sequence u_t verify*

$$u_t = \left(1 - \frac{\alpha}{t} + \mathrm{o}\left(\frac{1}{t}\right)\right) u_{t-1} + \frac{\beta}{t^2} + \mathrm{o}\left(\frac{1}{t^2}\right)$$

If $\alpha > 1$ and $\beta > 0$, then

$$t\,u_t \longrightarrow \frac{\beta}{\alpha - 1}$$

This result proves that u_t is asymptotically equivalent to $1/t$ and also provides the value of the constant factor. Amazingly enough, this constant does not depend on the unspecified low order terms of the recurrence.

Proof. Let us define $v_t = t u_t$ and observe that

$$\left(1 - \frac{\alpha}{t} + \mathrm{o}\left(\frac{1}{t}\right)\right) = \left(1 - \frac{1}{t}\right)\left(1 - \frac{\alpha - 1}{t} + \mathrm{o}\left(\frac{1}{t}\right)\right)$$

Multiplying the recurrence by t gives:

$$v_t = \left(1 - \frac{\alpha - 1}{t} + \mathrm{o}\left(\frac{1}{t}\right)\right) v_{t-1} + \frac{\beta}{t} + \mathrm{o}\left(\frac{1}{t}\right) \tag{12}$$

Let us define $v^* = \frac{\beta}{\alpha-1}$ and rewrite (12) as:

$$\begin{aligned}(v_t - v^*) &= \left(1 - \frac{\alpha-1}{t} + \mathrm{o}\left(\frac{1}{t}\right)\right)(v_{t-1} - v^*) + \mathrm{o}\left(\frac{1}{t}\right) \\ &= \left(1 - \frac{\alpha-1}{t} + \frac{A_t}{t}\right)(v_{t-1} - v^*) + \frac{B_t}{t} \qquad (13)\end{aligned}$$

with $A_t \to 0$ and $B_t \to 0$.

By repeated substitutions of (13), we obtain

$$(v_t - v^*) = h_t(v_0 - v^*) + \sum_{i=1}^{t} \frac{h_t}{h_i}\frac{B_i}{i} \qquad (14)$$

with

$$h_t = \prod_{j=1}^{t}\left(1 - \frac{\alpha - 1 + A_j}{j}\right)$$

To prove the lemma, we must prove that (14) converges to zero. There is an integer $i_0 > 0$ such that $1 - \frac{\alpha-1+A_i}{i}$ is positive for all $i > i_0$. For all $t > i_0$ we can then write:

$$\log\frac{h_t}{h_{i_0}} = \sum_{j=i_0+1}^{t} \log\left(1 - \frac{\alpha-1+A_j}{j}\right) \sim \sum_{i=j_0+1}^{t} -\frac{\alpha-1+A_j}{j} \to -\infty$$

Therefore $h_t \to 0$.

This result implies that the first term of (14) converges to zero. We now focus on the second term. Since $A_i \to 0$, there is a $i_1 > i_0$ such that $|A_i| < \frac{\alpha-1}{2}$. For all $i > i_1$ we can also write:

$$\log\frac{h_t}{h_i} = \sum_{j=i+1}^{t} \log\left(1 - \frac{\alpha-1+A_j}{j}\right) \;\leq\; -\sum_{j=i+1}^{t}\frac{\alpha-1+A_j}{j} \leq -\sum_{j=i+1}^{t}\frac{\alpha-1}{2j}$$

It is well known that

$$\int_i^t \frac{1}{x+1}\,dx \;\leq\; \sum_{j=i+1}^{t}\frac{1}{j} \;\leq\; \int_i^t \frac{1}{x}\,dx$$

and that

$$\forall i > 1,\; \int_i^t \frac{1}{x+1}\,dx > \int_i^t \frac{1}{x}\,dx \,-\, 1$$

Therefore we can write

$$\log\frac{h_t}{h_i} \leq \frac{\alpha-1}{2}\log\frac{i}{t} + \frac{\alpha-1}{2}$$

and

$$\forall\, t > i > i_1, \quad 0 \le \frac{h_t}{h_i} \le K_1 \left(\frac{i}{t}\right)^{\frac{\alpha-1}{2}} \qquad \text{with } K_1 = e^{\frac{\alpha-1}{2}}$$

Furthermore, for all $\varepsilon > 0$ there is a $i_2 > i_1$ such that $|B_i| < \varepsilon$ for all $i > i_2$. We can now bound the second term of (14) as follows:

$$\left|\sum_{i=1}^{t} \frac{h_t}{h_i} \frac{B_i}{i}\right| \le \left| h_t \sum_{i=1}^{i_2} \frac{B_i}{h_i\, i}\right| + \varepsilon\, K_1 \left|\sum_{i=i_2+1}^{t} \frac{1}{i}\left(\frac{i}{t}\right)^{\frac{\alpha-1}{2}}\right| \tag{15}$$

- The first term of this sum converges to zero because $h_t \to 0$.
- When t is large enough, the second term is smaller than $2\varepsilon\, K_1 K_2$ because

$$\sum_{i=i_2+1}^{t} \frac{1}{i}\left(\frac{i}{t}\right)^{\frac{\alpha-1}{2}} \le \frac{1}{t}\sum_{i=0}^{t}\left(\frac{i}{t}\right)^{\frac{\alpha-1}{2}-1} \longrightarrow \int_0^1 x^{\frac{\alpha-1}{2}-1}\, dx = K_2$$

We can now gather all the terms in (14) and (15). We can choose t large enough to ensure that the first term of (14) and the first term of (15) are each smaller than ε. We can also choose t large enough to ensure that the last term of (15) is smaller than $2\varepsilon\, K_1 K_2$.

Therefore we have just proven that for all $\varepsilon > 0$, we can choose t large enough to ensure that $|v_t - v^*| \le \varepsilon\,(2 + 2K_1K_2)$. Hence $v_t \to v^*$. □

We can now proceed with the proof of theorem 3.

Proof. To simplify the notation in the following proof, we assume that the optimum θ^* is located on the origin. This assumption entails no loss of generality since it only involves a translation of the coordinate system. We also use notation $J(\mathbf{z},\theta) = \frac{\partial L}{\partial \theta}(\mathbf{z},\theta)$.

The assumptions described in section A.2 are sufficient conditions for the general results discussed in section 4 of (Bottou, 1998). We know therefore that θ_t converges to $\theta^* = 0$ almost surely. This almost sure convergence implies that $\theta_t = \mathrm{o}_p(1)$. Since the θ_t are uniformly bounded, we have $\theta_t = \mathrm{o}_s(1)$.

We first derive an expression for $\theta_t\theta_t'$ by squaring the recursive update formula. This operation generates a number of high order terms that can be summarized as $\mathrm{o}_s(1/t^2)$. In particular the term $\theta_{t-1}\mathcal{O}(1/t^2)$ can be summarized as $\mathrm{o}_s(1/t^2)$ because we have established that $\theta_{t-1} = \mathrm{o}_s(1)$.

$$\begin{aligned} \theta_t\theta_t' \;=\; \theta_{t-1}\theta_{t-1}' &- \frac{\theta_{t-1} J'(\mathbf{z}_t,\theta_{t-1})\Phi}{t} - \frac{\Phi_t J(\mathbf{z}_t,\theta_{t-1})\theta_{t-1}'}{t} \\ &+ \frac{\Phi_t J(\mathbf{z}_t,\theta_{t-1}) J'(\mathbf{z}_t,\theta_{t-1})\Phi_t}{t^2} + \mathrm{o}_s\left(\frac{1}{t^2}\right) \end{aligned}$$

We shall now take compute the conditional expectation $\mathbf{E}\left(\theta_t\theta_t' \mid \mathcal{P}_t\right)$ where the notation $\mathcal{P}_t$ represents all variables known by time t, including the initial conditions θ_0 and the selected examples $\mathbf{z}_1, \ldots, \mathbf{z}_{t-1}$.

When the past $\mathcal{P}_t$ is known, variables Φ_t and θ_{t-1} are fixed and can be moved outside the expectation operator.

$$\mathbf{E}\left(\Phi_t J(\mathbf{z}_t,\theta_{t-1})J'(\mathbf{z}_t,\theta_{t-1})\Phi_t \mid \mathcal{P}_t\right) \;=\; \Phi_t\,\mathbf{E}_{\mathbf{z}}\left(J(\mathbf{z},\theta_{t-1})J'(\mathbf{z},\theta_{t-1})\right)\;\Phi_t \tag{16}$$

The following relation holds because $\theta_t = \mathrm{o}_s(1)$ and because function $J(\mathbf{z},\theta)$ is continuous and uniformly bounded.

$$\mathbf{E}_{\mathbf{z}}\left(J(\mathbf{z},\theta_{t-1})J'(\mathbf{z},\theta_{t-1})\right) \;=\; \mathcal{G} + \mathrm{o}_s(1)$$

We also remark that $\Phi_t = \mathcal{H}^{-1} + \mathrm{o}_s(1)$ because $\Phi_t = \mathcal{H}^{-1} + \mathrm{o}_p(1)$ and because both Φ_t and $\mathcal{H}^{-1}$ are uniformly bounded. Then

$$\mathbf{E}\left(\Phi_t J(\mathbf{z}_t,\theta_{t-1})J'(\mathbf{z}_t,\theta_{t-1})\Phi_t \mid \mathcal{P}_t\right) \;=\; \mathcal{H}^{-1}\,\mathcal{G}\,\mathcal{H}^{-1} + \mathrm{o}_s(1) \tag{17}$$

Using similar arguments we can also write the following equality.

$$\mathbf{E}\left(\Phi_t J(\mathbf{z}_t,\theta_{t-1})\theta_{t-1}' \mid \mathcal{P}_t\right) \;=\; \Phi_t\,\mathbf{E}_{\mathbf{z}}(J(\mathbf{z},\theta_{t-1}))\,\theta_{t-1}' \;=\; \Phi_t\,\frac{\partial C}{\partial\theta}(\theta_{t-1})\,\theta_{t-1}'$$
$$=\; \left(\mathcal{H}^{-1} + \mathrm{o}_s(1)\right)\left(\mathcal{H}\theta_{t-1} + \mathrm{o}_s(\theta_{t-1})\right)\theta_{t-1}' \;=\; \theta_{t-1}\theta_{t-1}' + \mathrm{o}_s\left(|\theta_{t-1}|^2\right)$$

We can now derive $\mathbf{E}\left(\theta_t\theta_t' \mid \mathcal{P}_t\right)$.

$$\mathbf{E}\left(\theta_t\theta_t' \mid \mathcal{P}_t\right) = \theta_{t-1}\theta_{t-1}' \;-\; \frac{2}{t}\,\theta_{t-1}\theta_{t-1}' \;+\; \mathrm{o}_s\left(\frac{|\theta_{t-1}|^2}{t}\right) \;+\; \frac{\mathcal{H}^{-1}\,\mathcal{G}\,\mathcal{H}^{-1}}{t^2} \;+\; \mathrm{o}_s\left(\frac{1}{t^2}\right)$$

Applying the trace operator gives the following expression.

$$\mathbf{E}\left(\,|\theta_t|^2 \mid \mathcal{P}_t\right) \;=\; \left(1-\frac{2}{t}\right)|\theta_{t-1}|^2 \;+\; \mathrm{o}_s\left(\frac{|\theta_{t-1}|^2}{t}\right) \;+\; \frac{\mathbf{tr}\left(\mathcal{H}^{-1}\,\mathcal{G}\,\mathcal{H}^{-1}\right)}{t^2} \;+\; \mathrm{o}_s\left(\frac{1}{t^2}\right)$$

We can then take the unconditional expectation, invoke theorem 1, and obtain

$$\mathbf{E}\left(\,|\theta_t|^2\right) \;=\; \left(1-\frac{2}{t}+\mathrm{o}\left(\frac{1}{t}\right)\right)\mathbf{E}\left(|\theta_{t-1}|^2\right) \;+\; \frac{\mathbf{tr}\left(\mathcal{H}^{-1}\,\mathcal{G}\,\mathcal{H}^{-1}\right)}{t^2} \;+\; \mathrm{o}\left(\frac{1}{t^2}\right)$$

Lemma 1 allows us to conclude.

$$\mathbf{E}\left(|\theta_t|^2\right) \;=\; \frac{\mathbf{tr}\left(\mathcal{H}^{-1}\,\mathcal{G}\,\mathcal{H}^{-1}\right)}{t} \;+\; \mathrm{o}\left(\frac{1}{t}\right)$$

□

Theorem 3 makes the assumption that Φ_t only depends on $z_1, \ldots, z_{t-1}$ and is therefore independent from z_t. This assumption is reasonable for online learning algorithms. However it is not verified by the empirical hessian (9) introduced in theorem 2. The following result relies on a weaker assumption which is verified by the empirical hessian (9).

Theorem 4. *The result from theorem 3 still holds when Φ_t can be written as*

$$\Phi_t = \mathbf{E}\left(\Phi_t \mid \mathcal{P}_t\right) + \phi_t(z_t) \ \text{ with } \ \phi_t(z_t) = \mathrm{o}_s\left(\frac{1}{t}\right)$$

The proof is essentially identical to the proof of theorem 3. The difference shows in equation (16). The scaling matrix Φ_t is no longer a constant when the past $\mathcal{P}_t$ is known. We cannot move Φ_t outside the expectation operator. We must instead use the following derivation where X is a uniformly bounded random variable.

$$\begin{aligned}
\mathbf{E}\left(\Phi_t X \mid \mathcal{P}_t\right) &= \mathbf{E}\left(\mathbf{E}\left(\Phi_t \mid \mathcal{P}_t\right) X \mid \mathcal{P}_t\right) + \mathbf{E}\left(\phi_t X \mid \mathcal{P}_t\right) \\
&= \mathbf{E}\left(\Phi_t \mid \mathcal{P}_t\right) \mathbf{E}\left(X \mid \mathcal{P}_t\right) + \mathrm{o}_s\left(\frac{1}{t}\right)
\end{aligned}$$

The last equality $\mathbf{E}\left(\phi_t X \mid \mathcal{P}_t\right) = \mathrm{o}_s\left(1/t\right)$ is a consequence of Schwartz's inequality:

$$\left|\mathbf{E}\left(\phi_t X \mid \mathcal{P}_t\right)\right| \leq \mathbf{E}\left(|\phi_t||X| \mid \mathcal{P}_t\right) \leq \sqrt{\mathbf{E}\left(|\phi_t|^2 \mid \mathcal{P}_t\right)}\sqrt{\mathbf{E}\left(|X|^2 \mid \mathcal{P}_t\right)}$$

We can then invoke the following relation

$$\mathbf{E}\left(\Phi_t \mid \mathcal{P}_t\right) = \mathbf{E}\left(\mathcal{H}^{-1} + \mathrm{o}_s\left(1\right) \mid \mathcal{P}_t\right) = \mathbf{E}\left(\mathcal{H}^{-1} \mid \mathcal{P}_t\right) + \mathrm{o}_s\left(1\right) = \mathcal{H}^{-1} + \mathrm{o}_s\left(1\right)$$

and obtain equation (17) without changes.

$$\mathbf{E}_{\mathbf{z}}\left(J(\mathbf{z}, \theta_{t-1}) J'(\mathbf{z}, \theta_{t-1})\right) \ = \ \mathcal{H}^{-1}\, \mathcal{G}\, \mathcal{H}^{-1} + \mathrm{o}_s\left(1\right)$$

The same argument yields

$$\mathbf{E}\left(\Phi_t J(\mathbf{z}_t, \theta_{t-1}) \theta'_{t-1} \mid \mathcal{P}_t\right) \ = \ \theta_{t-1}\theta'_{t-1} + |\theta_{t-1}|^2 \mathrm{o}_s\left(1\right) + \mathrm{o}_s\left(\frac{1}{t}\right)$$

The proof then proceeds without changes.

A.5 Complexity of Batch vs. Online Learning

Each iteration of a batch learning algorithm running on N training examples requires a time $K_1 N + K_2$. Constants K_1 and K_2 respectively represent the time to compute the gradient for each example, and the time to apply the update to the parameters. Theorems 2 and 4 indicate that

$$(\theta^*_N - \theta^*)^2 \;\sim\; \frac{\mathbf{tr}\left(\mathcal{H}^{-1}\,\mathcal{G}\,\mathcal{H}^{-1}\right)}{N}$$

We must perform enough iterations of the batch algorithm to approximate θ^*_N with at least the same accuracy $\mathcal{O}\left(1/N\right)$. A superlinear algorithm with quadratic convergence will achieve this in $\mathcal{O}\left(\log\log N\right)$ iterations.

Each iteration of an online learning algorithm requires a constant time. Processing T examples therefore requires time $K_3 T$. The number of examples processed by both algorithms with the same computing resources are therefore related by the following relation.

$$T = \mathcal{O}\left(N\ \log\log N\right)$$

We assume now that the online learning algorithm fulfils the conditions of theorem 3. Comparing the acuracies of both algorithms shows that the online algorithm asymptotically performs better.

$$(\theta_T - \theta^*)^2 \;\sim\; \frac{\mathbf{tr}\left(\mathcal{H}^{-1}\,\mathcal{G}\,\mathcal{H}^{-1}\right)}{T} \;<<\; \frac{\mathbf{tr}\left(\mathcal{H}^{-1}\,\mathcal{G}\,\mathcal{H}^{-1}\right)}{N} \;\sim\; (\theta^*_N - \theta^*)^2$$

The essential condition for such a fast online algorithm is the convergence of the scaling matrices Φ_t to $\mathcal{H}^{-1}$. This is not very difficuly in theory. The well known Natural Gradient algorithm, for instance, meets this condition and is known to perform optimally (Murata and Amari, 1999).

This means however that a full rank scaling matrix must be maintained. This is unfortunately not practical for large learning systems with many parameters. It is therefore important to find out whether reduced rank scaling matrices offer the same asymptotic properties.

References

Benveniste, A., Metivier, M., and Priouret, P. (1990). *Adaptive Algorithms and Stochastic Approximations*. Springer Verlag, Berlin, New York.

Boser, B., Guyon, I., and Vapnik, V. (1992). A Training Algorithm for Optimal Margin Classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, volume 5, pages 144–152.

Bottou, L. (1998). Online Algorithms and Stochastic Approximations. In Saad, D., editor, *Online Learning and Neural Networks*. Cambridge University Press, Cambridge, UK.

Bottou, L., Cortes, C., Denker, J. S., Drucker, H., Guyon, I., Jackel, L. D., Le Cun, Y., Muller, U. A., Säckinger, E., Simard, P., and Vapnik, V. N. (1994). Comparison of Classifier Methods: A Case Study in Handwritten Digit Recognition. In *Proceedings of the 13th International Conference on Pattern Recognition*, Jerusalem.

Bottou, L. and Murata, N. (2002). Stochastic Approximations and Efficient Learning. In Arbib, M. A., editor, *The Handbook of Brain Theory and Neural Networks, Second edition,*. The MIT Press, Cambridge, MA.

Dennis, J. and Schnabel, R. B. (1983). *Numerical Methods For Unconstrained Optimization and Nonlinear Equations*. Prentice-Hall, Inc., Englewood Cliffs, New Jersey.

Drucker, H., Schapire, R., and Simard, P. (1993). Improving performance in neural networks using a boosting algorithm. In Hanson, S. J., Cowan, J. D., and Giles, C. L., editors, *Advances in Neural Information Processing Systems 5*, pages 42–49, San Mateo, CA. Morgan Kaufmann.

Freund, Y. and Schapire, R. E. (1996). Experiments with a new boosting algorithm. In *Machine Learning: Proceedings of the Thirteenth International Conference*, pages 148–156.

Le Cun, Y., Bottou, L., Orr, G. B., and Müller, K.-R. (1998). Efficient Backprop. In *Neural Networks, Tricks of the Trade*, Lecture Notes in Computer Science 1524. Springer Verlag.

Mann, H. B. and Wald, A. (1943). On Stochastic Limit and Order Relationships. *Annal of mathematical Statistics*, 14:217–226

Murata, N. and Amari, S. (1999). Statistical analysis of learning dynamics. *Signal Processing*, 74(1):3–28.

Vapnik, V. N. and Chervonenkis, A. (1974). *Theory of Pattern Recognition* (in russian). Nauka. German translation: *Theorie der Zeichenerkennung*, Akademie Verlag, Berlin, 1979.