Continuous And Discrete-Time Nonlinear Gradient Descent: Relative Loss Bounds and Convergence

M. K. Warmuth and A. K. Jagota
Department of Computer Science
University of California at Santa Cruz
{manfred,jagota}@cs.ucsc.edu

September 30, 1997

Abstract

We introduce a general algorithm for continuous- and discrete-time nonlinear gradient-descent. The nonlinearity is captured by the choice of a link function. The discrete-time algorithm yields, for various choices of link function, the conventional gradient-descent algorithm as well as several exponentiated gradient ones. We obtain relative loss bounds for the general algorithm in an on-line setting for both the continuous- and discrete-time versions. These bounds reveal the dependence on the link function and show that an additional term is present in the discrete-time case which disappears in the continuous-time case. This additional term is responsible for the pair of dual norms that appear in the relative loss bounds for linear and logistic regression. The continuous-time version is also shown to have a simple proof of convergence in the batch setting. Convergence of Hopfield recurrent neural networks is seen as a special case.

1 Introduction

This paper introduces continuous- and discrete-time versions of an algorithm for non-linear gradient-descent. The main focus of this paper is the application of this algorithm to on-line learning. We also briefly discuss the implications for the batch case.

The model of on-line learning adopted in this paper is the following. An on-line learning algorithm is assumed to evolve in discrete-time $t = 0, h, 2h, \ldots$, parametrized by a fixed $h \in (0, 1]$. The continuous-time case is seen as the limit $h \to 0$. The algorithm's hypothesis at time t is represented by a parameter vector $\boldsymbol{\omega}_t \in \mathbf{R}^n$. At time t the algorithm receives an example $(\boldsymbol{x}_t, y_t)$ consisting of an instance $\boldsymbol{x}_t$ and a label y_t and incurs a loss describing how well the parameter vector $\boldsymbol{\omega}_t$ "performed" on the example. This loss is some non-negative function $L(\boldsymbol{\omega}_t, (\boldsymbol{x}_t, y_t))$ and we use $L_t(\boldsymbol{\omega}_t)$ as short-hand.

For example assume we have some fixed neural network architecture. The hypothesis at time t is the weight-vector $\boldsymbol{\omega}_t$. From the example $(\boldsymbol{x}_t, y_t)$ that arrives at time t, the input $\boldsymbol{x}_t$ is fed to the network, which

outputs $N(\boldsymbol{\omega}_t, \boldsymbol{x}_t)$. The loss that the network incurs on this example under its current hypothesis $\boldsymbol{\omega}_t$ is some function of y_t and $N(\boldsymbol{\omega}_t, \boldsymbol{x}_t)$, for example the square loss $L_t(\boldsymbol{\omega}_t) = (y_t - N(\boldsymbol{\omega}_t, \boldsymbol{x}_t))^2$.

The key piece that completes the description of the on-line model is the fact that $\boldsymbol{\omega}_t$ is updated to $\boldsymbol{\omega}_{t+1}$ at the end of the trial. The new parameter $\boldsymbol{\omega}_{t+1}$ is the hypothesis for the next time step $t+h$. In this paper we give a unified framework for various update algorithms and study what happens as the step size h goes to zero, i.e. we move from discrete-time (DT) to continuous-time (CT). The case $h = 1$ is the main model of on-line learning for which relative (worst-case) loss bounds have been proven [Lit88, LW94, Vov90, KW97b].

Our starting point is the conventional gradient descent algorithm, whose CT version is given by:

$$\dot{\boldsymbol{\omega}}_t = -\eta \bigtriangledown_{\boldsymbol{\omega}} L_t(\boldsymbol{\omega}_t) \ , \ \text{where} \ \ \dot{\boldsymbol{\omega}} \stackrel{\text{def}}{=} \left(\frac{\partial \boldsymbol{\omega}_t[i]}{\partial t} \right) \ \text{and} \ \ \bigtriangledown_{\boldsymbol{\omega}} L_t(\boldsymbol{\omega}_t) \stackrel{\text{def}}{=} \left(\frac{\partial L_t(\boldsymbol{\omega})}{\partial \boldsymbol{\omega}[i]} \right) \Bigg|_{\boldsymbol{\omega} = \boldsymbol{\omega}_t} \ .$$

Here we use $\boldsymbol{x}[i]$ to denote the $i-th$ component of a vector $\boldsymbol{x}$, and $\eta > 0$ is a finite learning rate. Throughout the paper we assume the loss function has the property that the partial derivatives of the loss used in the updates are defined and finite.

Replacing the vector $\dot{\boldsymbol{\omega}}_t$ by its Euler discretization $\frac{\boldsymbol{\omega}_{t+1} - \boldsymbol{\omega}_t}{h}$ gives a DT version of gradient descent:

$$\boldsymbol{\omega}_{t+1} := \boldsymbol{\omega}_t - \eta \ h \ \bigtriangledown_{\boldsymbol{\omega}} L_t(\boldsymbol{\omega}_t) \ .$$

In this paper we study the relationship between the CT algorithms and their Euler-discretized DT versions.

Our first algorithm, which we call the *main update*, is the following generalization of gradient descent:

$$\dot{\boldsymbol{f}}\ (\boldsymbol{\omega}_t) = -\eta \bigtriangledown_{\boldsymbol{\omega}} L_t(\boldsymbol{\omega}_t) \ . \tag{1}$$

Here, $\boldsymbol{f} = (f_i)$ is a component-wise vector-map. In this paper, all f_i are assumed to be identical for the sake of simplicity. The function f is a continuous function defined on some open interval $\mathcal{I}$ of $\mathbf{R}$ and $f'(z) > 0$ and finite for all $z \in \mathcal{I}$. A typical example is $f(z) = \ln(z)$. We call f a *link function* because of its connection to the *canonical link function* for the exponential family of densities used in statistics [MN89, FT91]. (The statistical interpretation of (1) and the related discretized updates given below are the focus of another paper [KW97a].)

Note that if f is increasing on its domain, then so is its inverse f^{-1}. Thus if f is a link function, then f^{-1} is so as well. For the vector of n original parameters $\boldsymbol{\omega}$ we introduce a vector of n *alternate* parameters $\boldsymbol{\theta}$ as follows:

$$\boldsymbol{\theta} = \boldsymbol{f}(\boldsymbol{\omega}) \ \text{and} \ \boldsymbol{\omega} = \boldsymbol{f^{-1}}(\boldsymbol{\theta}) \ .$$

The differential equations (1) of our main update are now

$$\dot{\boldsymbol{\theta}}_t = -\eta \bigtriangledown_{\boldsymbol{\omega}} L_t(\boldsymbol{\omega}_t) \quad \text{or} \quad \frac{d\boldsymbol{\theta}_t[i]}{dt} = -\eta \frac{\partial L_t(\boldsymbol{\omega}_t)}{\partial \boldsymbol{\omega}_t[i]}, \ \text{for all} \ i \ .$$

Table 1: Several discrete-time gradient-based algorithms: `EGU`—Unnormalized Exponentiated Gradient [KW97b], `EG`—Exponentiated Gradient [KW97b], and `BEG`—Bounded Exponentiated Gradient [Byl97]. Here $\nabla_{t,i}$ is short hand for $\frac{\partial L_t(\boldsymbol{\omega}_t)}{\partial \boldsymbol{\omega}_t[i]}$

link $\boldsymbol{\theta} = \boldsymbol{f}(\boldsymbol{\omega})$	$\boldsymbol{\theta} = \ln(\boldsymbol{\omega}), \boldsymbol{\omega}[i] \in (0, \infty)$	$\boldsymbol{\theta} = \ln(\boldsymbol{\omega}), \boldsymbol{\omega}[i] \geq 0, \sum_i \boldsymbol{\omega}[i] = 1$	$\boldsymbol{\theta} = \ln \frac{\boldsymbol{\omega}}{1-\boldsymbol{\omega}}, \boldsymbol{\omega}[i] \in (0, 1)$
$\Delta_f(\boldsymbol{\omega}^*, o)$	$\sum_i (\boldsymbol{\omega}^*[i] \ln \frac{\boldsymbol{\omega}^*[i]}{\boldsymbol{\omega}[i]} - \boldsymbol{\omega}^*[i] + \boldsymbol{\omega}[i])$	$\sum_i (\boldsymbol{\omega}^*[i] \ln \frac{\boldsymbol{\omega}^*[i]}{\boldsymbol{\omega}[i]})$	$\sum_i [\boldsymbol{\omega}^*[i] \ln \frac{\boldsymbol{\omega}^*[i]}{\boldsymbol{\omega}[i]} + (1 - \boldsymbol{\omega}^*[i]) \ln \frac{1-\boldsymbol{\omega}^*[i]}{1-\boldsymbol{\omega}[i]}]$
main update	EGU $\boldsymbol{\omega}_{t+h}[i] = \boldsymbol{\omega}_t[i] e^{-\eta h \nabla_{t,i}}$	EG $\boldsymbol{\omega}_{t+h}[i] = \frac{\boldsymbol{\omega}_t[i] e^{-\eta h \nabla_{t,i}}}{\sum_j \boldsymbol{\omega}_t[j] e^{-\eta h \nabla_{t,j}}}$	BEG $\boldsymbol{\omega}_{t+h}[i] = \frac{\boldsymbol{\omega}_t[i] e^{-\eta h \nabla_{t,i}}}{1 - \boldsymbol{\omega}_t[i](1 - e^{-\eta h \nabla_{t,i}})}$
dual update	dual EGU $\boldsymbol{\omega}_{t+h}[i] =$ $\boldsymbol{\omega}_t[i]\,(1 - \eta h \nabla_{t,i})$	dual EG $\boldsymbol{\omega}_{t+h}[i] =$ $\boldsymbol{\omega}_t[i] \frac{1 - \eta h \nabla_{t,i}}{1 - \eta h \sum_j \boldsymbol{\omega}_t[j] \nabla_{t,j}}$	dual BEG $\boldsymbol{\omega}_{t+h}[i] =$ $\boldsymbol{\omega}_t[i]\,(1 - \eta h (1 - \boldsymbol{\omega}_t[i]) \nabla_{t,i})$

Multiplying the last equalities by $\frac{d\boldsymbol{\omega}_t[i]}{d\boldsymbol{\theta}_t[i]} > 0$ yields a *dual* update of (1) that is equivalent to (1) in the CT case:

$$\dot{\boldsymbol{\omega}}_t = -\eta \, \nabla_{\boldsymbol{\theta}} \, L_t(\boldsymbol{\omega}_t) \quad \text{or} \quad \dot{\boldsymbol{f}^{-1}}(\boldsymbol{\theta}_t) = -\eta \, \nabla_{\boldsymbol{\theta}} \, L_t(\boldsymbol{f}^{-1}(\boldsymbol{\theta}_t)). \tag{2}$$

Here $\nabla_{\boldsymbol{\theta}} L_t(\boldsymbol{\omega}_t)$ is short hand for $\nabla_{\boldsymbol{\theta}} L\left(\boldsymbol{f}^{-1}(\boldsymbol{\theta})\right)$. Even though in CT the main update and its dual are equivalent, this is not always so for the DT versions. Euler discretization of the main update (1) gives

$$\boldsymbol{\theta}_{t+h} := \boldsymbol{\theta}_t - \eta \, h \;\; \nabla_{\boldsymbol{\omega}} \, L_t(\boldsymbol{\omega}_t) \quad \text{or} \quad \boldsymbol{\omega}_{t+h} := \boldsymbol{f}^{-1}\left(\boldsymbol{f}(\boldsymbol{\omega}_t) - \eta \, h \;\; \nabla_{\boldsymbol{\omega}} \, L_t(\boldsymbol{\omega}_t)\right) \;\; . \tag{3}$$

Note that in the above update the $\boldsymbol{\theta}$ parameters are updated additively based on the gradient with respect to the alternate parameters. The Euler-discretization of the dual update (2) gives

$$\boldsymbol{\omega}_{t+h} := \boldsymbol{\omega}_t - \eta \, h \;\; \nabla_{\boldsymbol{\theta}} \, L_t(\boldsymbol{\omega}_t) \quad \text{or} \quad \boldsymbol{\theta}_{t+h} := \boldsymbol{f}(\boldsymbol{f}^{-1}(\boldsymbol{\theta}_t) - \eta \, h \;\; \nabla_{\boldsymbol{\theta}} \, L_t(\boldsymbol{\omega}_t)) \;\; . \tag{4}$$

For example if f is the identity function then both the main update and its dual collapse to the conventional gradient descent update. However if $f(x) = \ln(x)$ then the discretized version (3) of the main update with $h = 1$ gives the Unnormalized Exponentiated Gradient Update (EGU) of [KW97b] (See the Table 1 for more examples).

In the next section we discuss the purpose and desired properties of link functions. The key tool for analyzing an update is a distance function associated with an update [Lit89, KW97b]. Here we use a general form that is based on an arbitrary link function. This form was introduced in [JK97] and was inspired by the definition of matching loss function given in [HKW95, AHW95, JK97].

$$\Delta_{\boldsymbol{f}}(\boldsymbol{\omega}^*, \boldsymbol{\omega}) = \sum_i \int_{\boldsymbol{\omega}[i]}^{\boldsymbol{\omega}^*[i]} (f_i(r) - f_i(\boldsymbol{\omega}[i])) dr \tag{5}$$

This distance function is usually asymmetric. In [KW97b] discretized updates are derived from the distance functions. In Section 3 we extend this method to derive the continuous-time updates also from the distance

functions. The general form of the link function based main update (3) was developed independently by [GLS97] for the case of classification with a linear threshold function. They used an alternate form of the above distance function as their starting point. Statistical interpretations of these distance functions and their relation to Information Geometry [Ama85] will be discussed in [KW97a].

We now generalize the methodology of worst-case relative loss bounds to our discrete-time and continuous-time cases. Imagine that a DT algorithm only sees the examples at the times $0, h, 2h, \ldots, T$. The algorithm incurs an instantaneous loss $L_t(\boldsymbol{\omega}_t) = L(\boldsymbol{\omega}_t, (\boldsymbol{x}_t, y_t))$ at each time t and a cumulative loss $\sum_{t=0,h,\ldots,}^{T} hL_t(\boldsymbol{\omega}_t)$ over the entire duration. Since the instantaneous loss can change arbitrarily, it is not fruitful to bound the cumulative loss. So we aim for the weaker goal of bounding the cumulative loss of the algorithm relative to the cumulative loss $\sum_{t=0,h,\ldots,}^{T} hL_t(\boldsymbol{\omega}^*)$ of the best off-line parameter vector $\boldsymbol{\omega}^*$.

A CT algorithm now sees a stream of examples in continuous-time $t \in [0, T]$. The cumulative loss of the CT algorithm is now defined as the integral $\int_{t=0}^{T} L_t(\boldsymbol{\omega}_t)dt$ over the time duration. The cumulative loss of the best off-line parameter vector is defined similarly as the integral $\int_{t=0}^{T} L_t(\boldsymbol{\omega}^*)dt$. Note that the cumulative loss of a DT algorithm is defined as the integral approximation of the associated CT one.

The CT bounds we obtain in this paper will have the form

$$\int_{t=0}^{T} L_t(\boldsymbol{\omega}_t)dt \leq \int_{t=0}^{T} L_t(\boldsymbol{\omega}^*)dt + \frac{\Delta_{\boldsymbol{f}}(\boldsymbol{\omega}^*, \boldsymbol{\omega}_0)}{\eta} \quad .$$

Note that such a bound improves as η is increased. To obtain such bounds we will need only mild assumptions on the stream of examples $(\boldsymbol{x}_t, y_t)_{t\in[0,T]}$ (see Corollary 2 in Section 4).

We shall also obtain similar bounds in the DT cases, with sums replacing integrals. In these bounds, an additional term will appear on the right hand side. (We can show that as h goes to zero, this term will also go to zero.) This term will also be seen to grow proportional to the learning rate η. Optimizing this DT bound (for any fixed h) will hence involve a careful tuning of η. Finally, this term is also responsible for the appearance of a pair of dual norms in the relative loss bounds for the DT linear and logistic regression [KW97b] for the GD and EG algorithms. The pair of dual norms characterizes the radically different behavior of the two algorithms that shows up experimentally [KW97b]. One of the norms measures the instances $\boldsymbol{x}_t$ and the corresponding dual norm measures the off-line parameter vector $\boldsymbol{\omega}^*$. So far this has only been done for two pairs of dual norms [KW97b, HKW95], one pair characterizing the GD algorithm and another one for the EG algorithm. However for the case of linear threshold functions the updates for more general pairs of dual norms have been developed and analyzed [GLS97]. This paper helps to explain why the dual norms disappear in the CT case, as well as gives a general form for a relative loss bound that holds both in the CT and DT cases.

2 Properties of link functions and distance functions

One of the purposes of the link function is to enforce interval constraints on the components of $\boldsymbol{\omega}$. If the components are to lie in (a, b) then we choose a link function with (a, b) as its domain and use the

discretized version (3) of our main update. For example if the interval is $(0, \infty)$, then $f(z) = \ln(z)$ is a suitable choice, giving the EGU algorithm. Similarly, if the interval is $(0, 1)$, then the inverse of the sigmoid function $f(z) = \ln \frac{z}{1-z}$ is a suitable choice, giving the BEG algorithm [Byl97]. Note that if we don't have any information about $\nabla_{\boldsymbol{\omega}} L(\boldsymbol{\omega}_t)$ and the learning rate η, then $\boldsymbol{f}(\boldsymbol{\omega}_t) - \eta\ h\ \nabla_{\boldsymbol{\omega}} L(\boldsymbol{\omega}_t)$ might not lie in the range of $\boldsymbol{f}$. Thus the discretized update (3) might not always be defined. A good property of the link functions f for our main update is that the range of f is $\mathbf{R}$. Note that $ln(z)$ and $\ln \frac{z}{1-z}$ have that property. Recall that the dual update (4) for the link function $\ln(z)$ is not always defined when η times the gradient is large.

There is a folklore method used in the neural network community that is different from the above in a subtle way. Assume we want to keep the components of $\boldsymbol{\omega}$ within a certain interval (a, b). Then as above one finds a function f with domain (a, b) and range $\mathbf{R}$ and does gradient descent with respect to the new unconstrained parameters $\boldsymbol{\theta} = \boldsymbol{f}(\boldsymbol{\omega})$. We call this the *reparameterized GD update*.

$$\dot{\boldsymbol{\theta}} = -\eta\ \nabla_{\boldsymbol{\theta}}\ L(\boldsymbol{\omega}_t)\ \ , \quad \boldsymbol{\theta}_{t+h} := \boldsymbol{\theta}_t - \eta\ h\ \ \nabla_{\boldsymbol{\theta}}\ L_t(\boldsymbol{\omega}_t)\ \ . \tag{6}$$

For example if $f(z) = ln(z)$ then the components of $\boldsymbol{\omega}$ are kept nonnegative. In a simple mixture estimation problem it was experimentally found that this reparameterization method did not work as well as the main update [HSSW97]. Another indication is that we are not aware of any relative loss bounds proven for the reparameterized GD update (except when f is the identity). However there is a large body of relative loss bounds for the main update with various choices for the link function.

The four CT updates are summarized below (recall that $\boldsymbol{\theta} = \boldsymbol{f}(\boldsymbol{\omega})$):

Name	Continuous-time version
GD	$\dot{\boldsymbol{\omega}}_t = -\eta\ \nabla_{\boldsymbol{\omega}}\ L_t(\boldsymbol{\omega}_t)$
Main	$\dot{\boldsymbol{\theta}}_t = -\eta\ \nabla_{\boldsymbol{\omega}}\ L_t(\boldsymbol{\omega}_t)$
Dual	$\dot{\boldsymbol{\omega}}_t = -\eta\ \nabla_{\boldsymbol{\theta}}\ L_t(\boldsymbol{\omega}_t)$
Reparametrized GD	$\dot{\boldsymbol{\theta}}_t = -\eta\ \nabla_{\boldsymbol{\theta}}\ L(\boldsymbol{\omega}_t)$

The DT versions arise from replacing $\dot{\boldsymbol{\omega}}_t$ and $\dot{\boldsymbol{\theta}}_t$ by $(\boldsymbol{\omega}_{t+h} - \boldsymbol{\omega}_t)/h$ and $(\boldsymbol{\theta}_{t+h} - \boldsymbol{\theta}_t)/h$, respectively.

As mentioned in the introduction we will analyze the updates derived from the link functions f using the associated distance functions $\Delta_{\boldsymbol{f}}$ given in (5). The following properties of a distance function follow immediately from the definitions:

(i) $\Delta_{\boldsymbol{f}}(\boldsymbol{\omega}^*, \boldsymbol{\omega}) \geq 0$ and equality implies $\boldsymbol{\omega}^* = \boldsymbol{\omega}$, (ii) $\Delta_{\boldsymbol{f}}(\boldsymbol{\omega}^*, \boldsymbol{\omega}) = \Delta_{\boldsymbol{f^{-1}}}(\boldsymbol{\theta}, \boldsymbol{\theta}^*)$, and (iii) $\Delta_{\boldsymbol{f}}(\boldsymbol{\omega}^*, \boldsymbol{\omega})$ is convex in $\boldsymbol{\omega}^*$ for every $\boldsymbol{\omega}$.

3 Derivation of updates from the distance function

Next we generalize a principled method that Kivinen and Warmuth [KW97b] used for deriving their algorithms. The aim is to minimize the cumulative loss $\sum_t h\ L_t(\boldsymbol{\omega}_t)$. The parameter vector $\boldsymbol{\omega}_t$ represents all the prior learning experience. To determine the updated parameter vector $\boldsymbol{\omega}_{t+h}$ one wants to minimize the

current loss $h\ L_t(\boldsymbol{\omega}_t)$ but also stay close to the previous parameter vector $\boldsymbol{\omega}_t$. Kivinen and Warmuth express this in terms of minimizing the function $U_t^h(\boldsymbol{\omega})$, where previously the step size h was always one:

$$\Delta_{\boldsymbol{f}}(\boldsymbol{\omega}, \boldsymbol{\omega}_t) + \eta\ h\ L_t(\boldsymbol{\omega}) \quad \approx \quad U_t^h(\boldsymbol{\omega}) \stackrel{\text{def}}{=} \Delta_{\boldsymbol{f}}(\boldsymbol{\omega}, \boldsymbol{\omega}_t) + \eta\ h\ (L_t(\boldsymbol{\omega}_t) + (\boldsymbol{\omega} - \boldsymbol{\omega}_t) \bigtriangledown_{\boldsymbol{\omega}} L_t(\boldsymbol{\omega}_t)) \tag{7}$$

Note that, since $\Delta_{\boldsymbol{f}}(\boldsymbol{\omega}, \boldsymbol{\omega}_t)$ is convex in $\boldsymbol{\omega}$, the Taylor approximation of $L_t(\boldsymbol{\omega})$ assures that $U_t^h(\boldsymbol{\omega})$ is convex in $\boldsymbol{\omega}$. (See [HSSW97] for further motivations of the approximation step.) We may then minimize $U_t(\boldsymbol{\omega})$ by setting all its partial derivatives with respect to $\boldsymbol{\omega}[i]$ to zero, giving the main update:

$$\boldsymbol{\omega}_{t+1} := \boldsymbol{f^{-1}}(\boldsymbol{f}(\boldsymbol{\omega}_t) - \eta \bigtriangledown_{\boldsymbol{\omega}} L_t(\boldsymbol{\omega}_{t+1})) \quad \text{or} \quad \frac{\boldsymbol{f}(\boldsymbol{\omega}_{t+1}) - \boldsymbol{f}(\boldsymbol{\omega}_t)}{h} = -\eta \bigtriangledown_{\boldsymbol{\omega}} L_t(\boldsymbol{\omega}_t) \tag{8}$$

This clearly becomes the continuous-time update (1) as $h \to 0$.

The dual updates (2,4) and the reparameterized gradient descent updates (6) can be derived in a similar fashion by minimizing

$$U_t^h(\boldsymbol{\theta}) = \Delta_{\boldsymbol{f}}(\boldsymbol{\theta}, \boldsymbol{\theta}_t) + \eta\ h\ (L_t(\boldsymbol{\omega}_t) + (\boldsymbol{\theta} - \boldsymbol{\theta}_t) \bigtriangledown_{\boldsymbol{\theta}} L_t(\boldsymbol{\omega}_t)) \quad \text{and} \quad U_t^h(\boldsymbol{\theta}) = ||\boldsymbol{\theta} - \boldsymbol{\theta}_t||_2^2/2 + \eta\ h\ (L_t(\boldsymbol{\omega}_t) + (\boldsymbol{\theta} - \boldsymbol{\theta}_t) \bigtriangledown_{\boldsymbol{\theta}} L_t(\boldsymbol{\omega}_t)) \ , \tag{9}$$

respectively. Note that for gradient descent half of the squared Euclidean distance is the distance function.

4 Relative Loss Bounds for On-line Learning

Here we bound the cumulative loss L of the on-line learning algorithm relative to the cumulative loss $L_{\boldsymbol{\omega}^*}$ of any fixed parameter vector $\boldsymbol{\omega}^*$ (i.e., $L_{\boldsymbol{\omega}^*}$ is obtained by using $\boldsymbol{\omega}^*$ instead of $\boldsymbol{\omega}_t$ as the parameter vector in each trial t, and accumulating the losses). The bound will hold for any parameter vector $\boldsymbol{\omega}^*$, in particular for the "best" fixed off-line parameter vector.

The relative loss bounds are obtained in the discretized setting with fixed step size $h \in (0, 1]$. The example sequence as well as the instantaneous loss are defined at times $t = 0, h, 2h, \ldots,$. Recall that we assumed that the partial derivatives of $L_t(\boldsymbol{\omega}_t)$ w.r.t. the components of $\boldsymbol{\omega}_t$ are defined and finite.

Theorem 1 *Let $\boldsymbol{f}$ be any link map and $(\boldsymbol{x}_t, y_t)$ be a stream of examples.*

- *For the main discrete update (8), if $L_t(\boldsymbol{\omega})$ is convex in $\boldsymbol{\omega}$ then*

$$\sum_{t=0,h,\ldots}^{T} h L_t(\boldsymbol{\omega}_t) \leq \sum_{t=0,h,\ldots}^{T} h L_t(\boldsymbol{\omega}^*) + \frac{1}{\eta}\left(\Delta_{\boldsymbol{f}}(\boldsymbol{\omega}^*, \boldsymbol{\omega}_0) - \Delta_{\boldsymbol{f}}(\boldsymbol{\omega}^*, \boldsymbol{\omega}_T)\right) + \frac{1}{\eta} \sum_{t=0,h,\ldots}^{T} \Delta_{\boldsymbol{f}}(\boldsymbol{\omega}_t, \boldsymbol{\omega}_{t+h}) \ ,$$

  for any fixed vectors $\boldsymbol{\omega}^*, \boldsymbol{\omega}_0 \in dom_f$.

- *For the dual discrete update (4), if $L_t(\boldsymbol{\omega}) = L_t(\boldsymbol{f^{-1}}(\boldsymbol{\theta}))$ is convex in $\boldsymbol{\theta}$ then*

$$\sum_{t=0,h,\ldots}^{T} h L_t(\boldsymbol{\omega}_t) \leq \sum_{t=0,h,\ldots}^{T} h L_t(\boldsymbol{\omega}^*) + \frac{1}{\eta}\left(\Delta_{\boldsymbol{f}}(\boldsymbol{\omega}_0, \boldsymbol{\omega}^*) - \Delta_{\boldsymbol{f}}(\boldsymbol{\omega}_T, \boldsymbol{\omega}^*)\right) + \frac{1}{\eta} \sum_{t=0,h,\ldots}^{T} \Delta_{\boldsymbol{f}}(\boldsymbol{\omega}_{t+h}, \boldsymbol{\omega}_t) \ ,$$

for any fixed vectors $\boldsymbol{\omega}^*, \boldsymbol{\omega}_0 \in range_{\boldsymbol{f^{-1}}}$.

- *For the reparametrized discrete update (6), if* $L_t(\boldsymbol{\omega}) = L_t(\boldsymbol{f^{-1}}(\boldsymbol{\theta}))$ *is convex in* $\boldsymbol{\theta}$ *then*

$$\sum_{t=0,h,\ldots}^{T} hL_t(\boldsymbol{\omega}_t) \leq \sum_{t=0,h,\ldots}^{T} hL_t(\boldsymbol{\omega}^*) + \frac{1}{2\eta}\left(||\boldsymbol{\theta}_0 - \boldsymbol{\theta}^*||_2^2 - ||\boldsymbol{\theta}_T - \boldsymbol{\theta}^*||_2^2\right) + \frac{1}{2\eta}\sum_{t=0,h,\ldots}^{T} ||\boldsymbol{\theta}_{t+h} - \boldsymbol{\theta}_t||_2^2 \ ,$$

for any fixed vectors $\boldsymbol{\omega}^*, \boldsymbol{\omega}_0 \in range_{\boldsymbol{f^{-1}}}$.

Proof The second bound follows from the proof of the first one by the properties of distance functions. The third bound follows from a small modification of the proof of the first one.

For the first bound we Taylor-expand $L_t(\boldsymbol{\omega}^*)$ around $\boldsymbol{\omega}_t$ and apply the main discrete update (8):

$$L_t(\boldsymbol{\omega}^*) \geq L_t(\boldsymbol{\omega}_t) + (\boldsymbol{\omega}^* - \boldsymbol{\omega}_t)\cdot \bigtriangledown_{\boldsymbol{\omega}} L_t(\boldsymbol{\omega}_t) = L_t(\boldsymbol{\omega}_t) + \frac{1}{\eta}(\boldsymbol{\omega}^* - \boldsymbol{\omega}_t)\cdot\left(\frac{\boldsymbol{f}(\boldsymbol{\omega}_t) - \boldsymbol{f}(\boldsymbol{\omega}_{t+h})}{h}\right) \ . \tag{10}$$

The first bound of the theorem now follows by summing over $t = 0, h, \ldots, T$ and using the following property of the distance function that is easy to check (see full paper):

$$(\boldsymbol{\omega}^* - \boldsymbol{\omega}_t)\cdot(\boldsymbol{f}(\boldsymbol{\omega}_t) - \boldsymbol{f}(\boldsymbol{\omega}_{t+h})) = \Delta_{\boldsymbol{f}}(\boldsymbol{\omega}^*, \boldsymbol{\omega}_{t+h}) - \Delta_{\boldsymbol{f}}(\boldsymbol{\omega}^*, \boldsymbol{\omega}_t) - \Delta_{\boldsymbol{f}}(\boldsymbol{\omega}_t, \boldsymbol{\omega}_{t+h}) \ . \tag{11}$$

■

We will show now that in the CT case the upper bounds simplify in that the last sum in each of the bounds becomes 0 as $h \to 0$. We will need to assume that, for the CT systems to have solutions, $\sum \frac{1}{f'(\boldsymbol{\omega}_t[i])}\frac{\partial L_t(\boldsymbol{\omega}_t)}{\partial \boldsymbol{\omega}_t[i]}$ must be integrable w.r.t. time.

Corollary 2 *For the continuous-time versions of the three updates in Theorem 1, the corresponding three bounds are*

$$\begin{aligned}
\int_{t=0}^{T} L_t(\boldsymbol{\omega}_t)dt &\leq \int_{t=0}^{T} L_t(\boldsymbol{\omega}^*)dt + \frac{1}{\eta}\left(\Delta_{\boldsymbol{f}}(\boldsymbol{\omega}^*, \boldsymbol{\omega}_0) - \Delta_{\boldsymbol{f}}(\boldsymbol{\omega}^*, \boldsymbol{\omega}_T)\right) \ , \\
\int_{t=0}^{T} L_t(\boldsymbol{\omega}_t)dt &\leq \int_{t=0}^{T} L_t(\boldsymbol{\omega}^*)dt + \frac{1}{\eta}\left(\Delta_{\boldsymbol{f}}(\boldsymbol{\omega}_0, \boldsymbol{\omega}^*) - \Delta_{\boldsymbol{f}}(\boldsymbol{\omega}_T, \boldsymbol{\omega}^*)\right) \ , \\
\int_{t=0}^{T} L_t(\boldsymbol{\omega}_t)dt &\leq \int_{t=0}^{T} L_t(\boldsymbol{\omega}^*)dt + \frac{1}{\eta}\left(||\boldsymbol{\theta}_0 - \boldsymbol{\theta}^*||_2^2 - ||\boldsymbol{\theta}_T - \boldsymbol{\theta}^*||_2^2\right) \ .
\end{aligned}$$

Proof We only prove the first part. For the CT main update (1), Inequality (10) becomes

$$L_t(\boldsymbol{\omega}^*) \geq L_t(\boldsymbol{\omega}_t) + (\boldsymbol{\omega}^* - \boldsymbol{\omega}_t)\cdot \bigtriangledown L_t(\boldsymbol{\omega}_t) = L_t(\boldsymbol{\omega}_t) - \frac{1}{\eta}(\boldsymbol{\omega}^* - \boldsymbol{\omega}_t)\cdot \dot{\boldsymbol{f}}\ (\boldsymbol{\omega}_t) = L_t(\boldsymbol{\omega}_t) + \frac{1}{\eta}\dot{\Delta}_{\boldsymbol{f}}\ (\boldsymbol{\omega}^*, \boldsymbol{\omega}_t) \ .$$

The last equality can be checked by applying the chain rule $\dot{\Delta}_{\boldsymbol{f}}\ (\boldsymbol{\omega}^*, \boldsymbol{\omega}_t) = \sum_{i=1}^{n} \frac{\partial \Delta_{\boldsymbol{f}}(\boldsymbol{\omega}^*, \boldsymbol{\omega}_t)}{\partial \boldsymbol{\omega}_t[i]}\ \dot{\boldsymbol{\omega}}_t\ [i]$. The first part follows now by integration.

■

The above proof does not show however that the additional terms (the last summations in each of the three bounds of Theorem 1) become zero as h goes to 0, i.e.,

$$\lim_{h \to 0} \sum_{t=0,h,\ldots}^{T} \Delta_{\boldsymbol{f}}(\boldsymbol{\omega}_t, \boldsymbol{\omega}_{t+h}) = 0. \tag{12}$$

This is shown in the full paper.

The methods in [KW97b, HKW95, JK97, Byl97] for linear and logistic regression can be interpreted as upper bounding this additional sum by $\eta h c_f X^2 \sum_{t=0,h,\ldots}^{T} h L_t(\boldsymbol{\omega}_t)$, where X is an upper bound on the norm of the instances and c_f is a constant. The distance function introduces a dual norm of the off-line parameter vector $\boldsymbol{\omega}^*$. Tuning of η then gives a relative loss bound of the usual form that grows proportional with the square of a pair of dual norms.

5 The Batch Setting, Monotone Convergence, and Hopfield Nets

We now consider the situation where the function $L(\boldsymbol{\omega})$ is fixed, i.e., invariant of time t. This is the case in the batch version of the learning problem where the example set $\{\boldsymbol{x_t}, y_t\}$ is fixed in advance. The loss function $L(\boldsymbol{\omega})$ is then simply the total loss on all examples as a function of the parameter vector $\boldsymbol{\omega}$ (e.g. weight-vector of a neural network).

The relative loss bounds in the on-line learning case can be used to show bounds on the speed of convergence on the training data. More interestingly, using a simple averaging argument these bounds can also be converted into bounds on the generalization error [KW97b].

Whether or not relative loss bounds are obtainable the above way, it is easy to establish in the CT case monotone convergence of the loss. We only need the assumption that the time-invariant L function is bounded from below. By monotone convergence we mean that as $t \to \infty$, $L(\boldsymbol{\omega}_t)$ approaches some fixed value L^* in a monotone way.

Recall that L is bounded from below. Thus to establish monotone convergence for the main CT update (1), it suffices to show that $\dot{L}\ (\boldsymbol{\omega}) < 0$ if at least one component of $\dot{\boldsymbol{\omega}}\ [i]$ is non-zero. (See [HKP91, p. 56] for a similar derivation in the context of Hopfield nets.) If at least one component of $\dot{\boldsymbol{\omega}}\ [i]$ is non-zero then

$$\dot{L}\ (\boldsymbol{\omega}) \ = \ \sum_{i=1}^{n} \frac{\partial L(\boldsymbol{\omega})}{\partial \boldsymbol{\omega}[i]}\ \dot{\boldsymbol{\omega}}_i \ \ = \ \ -\,(1/\eta) \sum_{i=1}^{n} \dot{\boldsymbol{f}}_i\ (\boldsymbol{\omega}[i])\ \dot{\boldsymbol{\omega}}\ [i] \ = \ \ -\,(1/\eta) \sum_{i=1}^{n} f_i'(\boldsymbol{\omega}[i])\ \dot{\boldsymbol{\omega}}\ [i]^2 \ < \ 0$$

The function L is also time-invariant in fixed-weights recurrent networks such as the Hopfield network [HKP91]. In this case the loss function $L_{\boldsymbol{W}}(\boldsymbol{x})$ is usually a scalar Liapunov function and $\boldsymbol{x}$ is the activity-vector of the n neuron states. Note that here we use the more customary parameter vector $\boldsymbol{x}$ in place of $\boldsymbol{\omega}$ used in the rest of the paper. The subscript $\boldsymbol{W}$ is the fixed interconnection weight matrix between the pairs of neurons. Thus the function $L_{\boldsymbol{W}}$ here tracks the evolution of the neuronal states and not the weights. The above argument yields convergence of continuous-time Hopfield networks [HKP91] as a special case.

For example a simple Liapunov function for Hopfield nets is $L_{\boldsymbol{W}}(\boldsymbol{x}) = -\frac{1}{2}\boldsymbol{x}^T\boldsymbol{W}\boldsymbol{x}$. Since the activity of each state must lie in $(0,1)$ we would naturally choose a link function f with domain $(0,1)$. So we choose $\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x})$, where $f(z) = \ln\frac{z}{1-z}$ is the inverse of the logistic function. Now our main update (3) for CT becomes $\dot{\boldsymbol{y}} = \frac{1}{2}(\boldsymbol{W} + \boldsymbol{W^T})\boldsymbol{x}$, which corresponds to the BEG algorithm of Tom Bylander [Byl97], and is also a convergent update for Hopfield nets with arbitrary $\boldsymbol{W}$. When $\boldsymbol{W}$ is symmetric, we have $\dot{\boldsymbol{y}} = \boldsymbol{W}\boldsymbol{x}$, a known update for the Hopfield net. The dual of this update is $\dot{\boldsymbol{x}} = \boldsymbol{x} \times (\boldsymbol{1} - \boldsymbol{x}) \times \boldsymbol{W}\boldsymbol{x}$, where $\times$ denotes component-wise multiplication. The discrete-time version of this dual update avoids the use of the logarithm and exponentiation that is needed in the discrete-time version of the main update.

The standard CT Hopfield update is also derivable in our framework via the Hopfield Liapunov function

$$L_{\boldsymbol{W}}(\boldsymbol{x}) = -\frac{1}{2}\boldsymbol{x}^T\boldsymbol{W}\boldsymbol{x} + \gamma\sum_i \int_0^{\boldsymbol{x}[i]} f(z)dz = -\frac{1}{2}\boldsymbol{x}^T\boldsymbol{W}\boldsymbol{x} + \gamma\sum_i \left(\boldsymbol{x}[i]\ln\boldsymbol{x}[i] + (1-\boldsymbol{x}[i])\ln(1-\boldsymbol{x}[i])\right) \quad , \quad (13)$$

where f is the inverse of the logistic function. This choice of $L_{\boldsymbol{W}}$ yields the CT update $\dot{\boldsymbol{y}} = -\boldsymbol{y} + \frac{1}{2}(\boldsymbol{W} + \boldsymbol{W}^T)\boldsymbol{x}$ for the arbitrary $\boldsymbol{W}$ case and $\dot{\boldsymbol{y}} = -\boldsymbol{y} + \boldsymbol{W}\boldsymbol{x}$ for the symmetric $\boldsymbol{W}$ case, which is the familiar Hopfield update. The following Liapunov function differs from the standard one by an additive constant:

$$L_{\boldsymbol{W}}(\boldsymbol{x}) = -\frac{1}{2}\boldsymbol{x}^T\boldsymbol{W}\boldsymbol{x} + \gamma\sum_i \left(\boldsymbol{x}[i]\ln\frac{\boldsymbol{x}[i]}{0.5} + (1-\boldsymbol{x}[i])\ln\frac{(1-\boldsymbol{x}[i])}{0.5}\right) = -\frac{1}{2}\boldsymbol{x}^T\boldsymbol{W}\boldsymbol{x} + \gamma\Delta_{\boldsymbol{f}}(\boldsymbol{x}, (0.5,\ldots,0.5)).$$

The distance to the uniform start vector $(0.5,\ldots,0.5)$ may be seen as a regularization term. With this Liapunov function the standard Hopfield update is again the BEG update.

6 Conclusion

This paper has presented a general update algorithm for continuous- and discrete-time nonlinear gradient-descent, yielding a large number of different particular updates via the choice of a link function. Relative loss bounds have been obtained for the general update in an on-line setting for both the continuous- and discrete-time cases. These bounds have helped explain why the dual norms disappear in the CT case.

In the batch setting, the continuous-time general algorithm was shown to have a simple proof of convergence. Convergence of Hopfield recurrent neural networks was shown as a special case.

We introduced a large number of different updates in this paper. However, the relative loss bounds also provide a yard stick for discerning between the updates [KW97b, HKW95]. The larger research goal is a systematic study of learning algorithms in simple settings using the link function as way of characterizing the update. The CT case is a simplified extreme case that can aide the understanding of the DT case.

Acknowledgments

The authors thank Mark Herbster, Tommi Jaakkola, Jyrki Kivinen, Nick Littlestone, Eduardo Sontag, and Joel Yellin for interesting discussions.

References

[AHW95] P. Auer, M. Herbster, and M. K. Warmuth. Exponentially many local minima for single neurons. In *Proc. 1995 Neural Information Processing Conference*, pages 316–317. MIT Press, Cambridge, MA, November 1995.

[Ama85] S. Amari. *Differential Geometrical Methods in Statistics*. Springer Verlag, Berlin, 1985.

[Byl97] T. Bylander. The binary exponentiated gradient algorithm for learning linear functions. In *Proc. 8th Annu. Conf. on Comput. Learning Theory*, pages 184–192. ACM, 1997. To appear.

[FT91] L. Fahrmeir and G. Tutz. *Multivariate Statistical Modelling Based on Generalized linear Models*. Springer-Verlag, New-York, 1991.

[GLS97] A. J. Grove, N. Littlestone, and D. Schuurmans. General convergence results for linear discriminant updates. In *Proc. 8th Annu. Conf. on Comput. Learning Theory*, pages 171–183. ACM, 1997. To appear.

[HKP91] J. Hertz, A. Krogh, and R.G. Palmer. *Introduction to the Theory of Neural Computation*. Addison-Wesley, New York, 1991.

[HKW95] D. P. Helmbold, J. Kivinen, and M. K. Warmuth. Worst-case loss bounds for sigmoided linear neurons. In *Proc. 1995 Neural Information Processing Conference*, pages 309–315. MIT Press, Cambridge, MA, November 1995.

[HSSW97] D. Helmbold, R. E. Schapire, Y. Singer, and M. K. Warmuth. A comparison of new and old algorithms for a mixture estimation problem. *Machine Learning*, 1997. To appear.

[JK97] M. K. Warmuth J. Kivinen. Relative loss bounds for multidimensional regression problems. In *Advances in Neural Information Processing Systems, 11*, Cambridge, MA, 1997. MIT Press. To appear.

[KW97a] K. Kazoury and M. K. Warmuth. Relative loss bounds and the exponential family of distributions. Unpublished manuscript., 1997.

[KW97b] Jyrki Kivinen and Manfred K. Warmuth. Additive versus exponentiated gradient updates for linear prediction. *Information and Computation*, 132(1):1–64, January 1997.

[Lit88] N. Littlestone. Learning when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2:285–318, 1988.

[Lit89] N. Littlestone. *Mistake Bounds and Logarithmic Linear-threshold Learning Algorithms*. PhD thesis, Technical Report UCSC-CRL-89-11, University of California Santa Cruz, 1989.

[LW94] N. Littlestone and M. K. Warmuth. The weighted majority algorithm. *Information and Computation*, 108(2):212–261, 1994.

[MN89] P. McCullagh and J. A. Nelder. *Generalized Linear Models*. Chapman & Hall, 1989.

[Vov90] V. Vovk. Aggregating strategies. In *Proc. 3rd Annu. Workshop on Comput. Learning Theory*, pages 371–383. Morgan Kaufmann, 1990.