Perceptron Training Algorithms designed using Discrete-Time Control Liapunov Functions

Oumar Diene and Amit Bhaya

Abstract—Perceptrons, proposed in the seminal paper McCulloch–Pitts of 1943, have remained of interest to neural network community because of their simplicity and usefulness in classifying linearly separable data. Gradient descent and conjugate gradient are two widely used techniques for solving a set of linear inequalities. In finite precision implementation, the numerical errors could cause a loss of the residue orthogonality, which, in turn, results in loss of convergence. This paper takes a recently proposed control-inspired approach, to the design of iterative perceptron training algorithms, by regarding certain training/algorithm parameters as controls and then using a control Liapunov technique to choose appropriate values of these parameters.

I. INTRODUCTION

PERCEPTRONS, proposed in the seminal paper McCulloch–Pitts of 1943, have remained of interest to neural network community because of their simplicity and usefulness in classifying linearly separable data. The recent focus on support vector machines, which can be viewed as an outgrowth of perceptrons, has also rekindled interest in the basic properties of perceptrons and their so-called training algorithms. Training a perceptron involves (recursively) solving a set of linear inequalities and several algorithms have been proposed in the literature.

The perceptron training rule is an error-correcting rule used to generate the weights and threshold for a linear threshold gate (perceptron) such that a linear decision surface is synthesized that discriminates between two linearly separable class samples clustered in feature space. The main advantage of the perceptron training rule is its low computational requirements and its ability to guarantee convergence to a solution for linearly separable problems [1]. The main drawback of this rule is the fact that it does not converge to useful approximate solutions for linearly nonseparable problems [2].

The Ho-Kashyap algorithm determines a mean squared error by choice of the control law

\[ \lim_{k \to \infty} \text{residue} = 0 \]

I.e., designing control signal \( u_k \) in order to zero the steady state error by choice of the control law \( f(\cdot) \) i.e., \( \lim_{k \to \infty} r_k = 0 \), or, equivalently, \( \lim_{k \to \infty} Ax_k = b \).

System (3), represented in block diagram form in figure 1, models an iterative method as a feedback control system (plant \{I, I, A, 0\} with controller in unitary feedback configuration). In control theory, both classical and modern, there are various techniques of solving the signal regulation problem:
orthogonality (see, e.g., [6]). This indicates that, in situ-
ations (initialization of conjugate directions), used to obtain re-
side convergence in the batch mode, but without the use of any
parameters by CLF still ensures the decrease of some norm
where the assumption does not hold, the determination of CG
method, since it does not use the assumption
The CLF method
show that, by using appropriate quadratic CLFs, choosing
allocation, state feedback) result in controllers that explicitly
result in controllers that explicitly
the CG method, shown in algorithm 1. The CLF method
Let x represent the n-dimensional weight vector. Then, the
training of a single perceptron can be formulated as: find an
n-vector x such that
\[ Ax > 0. \]
This can be formulated [4] as the problem of finding an
element \((x, b)\) belonging to the set
\[ S^+ := \{ x \in \mathbb{R}^n, b \in \mathbb{R}^m : Ax = b > 0 \}. \]
The positive vector \(b \in \mathbb{R}^m\) is henceforth referred to as the
margin vector. Ho and Kashyap [3] have proposed an
iterative algorithm for finding an element of \(S^+\). In this case,
the components of the margin vector \(b\) are first initialized to
small positive values, and the Moore-Penrose inverse is used
to generate an MSE minimum norm solution for \(x_k\) which
minimizes the objective function
\[ J(x_k, b_k) = \|Ax_k - b_k\|^2; \]
where \((\cdot)^\dagger\) denotes the Moore-Penrose inverse. Next, a new
estimate for the margin vector is computed by performing the
constrained descent
\[ b_{k+1} = b_k + \frac{1}{2} (|r_k| - r_k), \]
where \(r_k = b_k - Ax_k\). A new estimate of \(x\), \(x_{k+1}\) can now be computed using (7) and employing the updated
margin vector from (8), this process is iterated until all the
components of \(r_k\) are zero, which is an indication of linear
separability of the training set, or until \(r_k > 0\), which in
this case is an indication of nonlinear separability of the
training set [4]. A new algorithm can be obtained by using
conjugate gradient techniques to obtain the solution \(x_k\) which
minimizes the objective function \(J(x_k, b_k)\) and compute a
new estimate for the margin vector. A recently proposed
control Lyapunov function (CLF) analysis of the CG method
viewed as a dynamic system in the standard feedback con-
figuration [12] is applied to the case of perceptron training.
The main technique consists of an appropriate choice of a
control Lyapunov function (CLF), and has been used in [12]
to analyze and design Krylov methods and in [7] to design
new adaptive filtering algorithms.
Combining the conjugate gradient technique with the Ho-
Kashyap technique [4], system (3) can be rewritten as:
\[
\begin{align*}
  x_{k+1} & = x_k + \alpha_k p_{x_k} \\
  b_{k+1} & = b_k + \alpha_k p_{b_k} \\
  r_k & = b_k - Ax_k \\
  p_{x_k} & = \beta_k p_{x_{k-1}} + A^T r_k \\
  p_{b_k} & = \beta_k p_{b_{k-1}} + |r_k| - r_k
\end{align*}
\]
Figure 2 shows the block diagram representing system (9),
and the main result of this paper is stated as follows:
Then,\[
\Delta V_r = V_r(k+1) - V_r(k) = \langle r_{k+1}, r_{k+1} \rangle - \langle r_k, r_k \rangle
\]
\[
= \langle r_k, r_k \rangle - \langle \alpha_k \beta_k p_k, r_k \rangle - \alpha_k \beta_k p_k - \langle r_k, r_k \rangle
\]
\[
= \langle \alpha_k \beta_k (r_k, p_k) + \alpha_k^2 (p_k, p_k) - (r_k, r_k) \rangle
\]
\[
= -2\alpha_k \langle r_k, p_k \rangle + \alpha_k^2 (p_k, p_k).
\] (18)

The value of \( \alpha_k \), given by (13), is determined calculating \( \frac{\partial \Delta V_r}{\partial V_r} \) and setting it to zero. Since \( p_k^T r_k \neq 0 \), substituting (13) into (18) yields
\[
\Delta V_r = \frac{\langle r_k, p_k \rangle^2}{\langle p_k, p_k \rangle} < 0.
\] (19)

From (16), (13), (11), and (10), that
\[
\langle r_k, p_k \rangle = \frac{\beta_k}{\beta_k} \langle r_{k-1}, p_{k-1} \rangle + \langle r_k, r_k \rangle = r_{k-1}^T A A^T r_k + r_{k-1}^T (r_k - |r_k|) > 0.
\] (20)

From (19), it can be concluded that the 2-norm of the residue \( r_k \) decreases monotonically.

The next step is to determine the parameter \( \beta_k \), and for this purpose define
\[
\Rightarrow \beta_{k+1} = -\langle r_{k+1}^T, p_k \rangle
\] (22)

Substituting (22) in (21) leads
\[
V_{p_{k+1}} = \langle r_{k+1}^T, p_{k+1} \rangle - \frac{\langle r_{k+1}^T, p_{k+1} \rangle^2}{\langle p_k, p_k \rangle} \leq \langle r_{k+1}^T, r_{k+1} \rangle
\] (23)

From (19) and (11) it can be concluded that \( r_{k+1} \) decreases in 2-norm, but not necessarily monotonically. Thus (23) implies that \( p_{k+1} \) decreases in 2-norm, although not necessarily monotonically, concluding the proof.

The above derivations can be summarized in the form of the following algorithm

**Algorithm 2:** Batch mode CG-CLF for Perceptron Training

Choose \( x_0, b_0 > 0 \)

Calculate \( r_0 = b_0 - A x_0, p_{x0} = A^T r_0 \)

Calculate \( p_{b0} = |r_0| - r_0, p_{b0} = A p_{x0} - p_{b0} \)

For \( k = 0, 1, \ldots \), until convergence with \( b_{\text{final}} > 0 \)

\[
\alpha_k = \frac{\langle r_k, p_k \rangle}{\langle p_k, p_k \rangle}
\]

\[
\begin{align*}
x_{k+1} &= x_k + \alpha_k p_{x_k} \\
b_{k+1} &= b_k + \alpha_k p_{b_k} \\
r_{k+1} &= r_k - \alpha_k p_k \\
r'_{k+1} &= (A A^T + I) r_{k+1} - \alpha_k p_k \\
\beta_{k+1} &= -\frac{\langle r'_{k+1}, p_k \rangle}{\langle p_k, p_k \rangle}
\end{align*}
\]

\[
\begin{align*}
p_{x_{k+1}} &= \beta_{k+1} p_{x_k} + A^T r_{k+1} \\
p_{b_{k+1}} &= \beta_{k+1} p_{b_k} + |r_{k+1}| - r_{k+1} \\
p_{k+1} &= \beta_{k+1} p_{k} + r'_{k+1}
\end{align*}
\]

End

**Observation 1:** Notice that theorem 1 guarantees that algorithm 2 will converge to a solution of \( A x_k = b_k \), but
does not ensure that $b_k > 0$. However, the experimental results below show that algorithm 2 converges to a solution $x_k$ such that $Ax_k = b_k > 0$, and a rigorous proof that convergence to $S^*$, instead of just to $S$, occurs is being sought.

In [7], a steepest descent (SD) method for adaptive filters was proposed. The results in [7] show that it is possible for the SD method to design, by CLFs, a SD method in order to design, by CLFs, a SD method for batch mode perceptron training (SD-CLF). The system of equations representing the regulation problem, equivalent to the batch mode perceptron training problem by the SD-CLF method, is given by (24) below

$$\begin{align*}
x_{k+1} &= x_k + \alpha_k A^T r_k \\
\lambda_{k+1} &= b_k + \alpha_k (r_k | - r_k) \\
r_k &= b_k - Ax_k
\end{align*}$$

A convergence result can now be stated as follows:

**Theorem 2:** With the assumption (10), the choice

$$\alpha_k = \frac{(r_k, r'_k)}{(r'_k, r'_k)}$$

is optimal and ensures that the equilibrium of the system (24) is asymptotically stable (i.e. $r_k \to 0$), where $r'_k$ is given by (11).

**Proof:** From (24), one can write

$$\begin{align*}
r_{k+1} &= b_{k+1} - Ax_{k+1} \\
&= b_k + \alpha_k (r_k | - r_k) - Ax_k - \alpha_k A^T r_k \\
&= r_k - \alpha_k (A A^T + I) r_k - |r_k| \\
&= r_k - \alpha_k r'_k,
\end{align*}$$

where $r'_k$ is given by (11). Once again choose the control Liapunov function $V_r(k)$ given by (17), then

$$\Delta V_r = V_r(k+1) - V_r(k) = \langle r_{k+1}, r_{k+1} \rangle - \langle r_k, r_k \rangle = \langle r_k - \alpha_k r'_k, r_k - \alpha_k r'_k \rangle - \langle r_k, r_k \rangle = \langle r_k, r_k \rangle - 2\alpha_k \langle r_k, r'_k \rangle + \alpha_k^2 \langle r'_k, r'_k \rangle - \langle r_k, r_k \rangle = -2\alpha_k \langle r_k, r'_k \rangle + \alpha_k^2 \langle r'_k, r'_k \rangle.$$  

(27)

The optimal value of $\alpha_k$ (which makes $\Delta V_r$ as negative as possible), given by (25), is determined calculating $\frac{\partial \Delta V_r}{\partial \alpha_k}$ and setting it to zero. Substituting (25) into (27) yields

$$\Delta V_r = \frac{\langle r_k, r'_k \rangle^2}{\langle r'_k, r'_k \rangle} < 0,$$

(28)

which implies that $||r_k||$ is a decreasing sequence.

The pseudocode of the extension of the SD method for batch mode perceptron training, based on theorem 2 and referenced in this paper as SD-CLF, is presented in algorithm 3.

**Algorithm 3:** Batch mode SD-CLF for Perceptron Training

Choose $x_0, b_0 > 0$

Calculate $r_0 = b_0 - Ax_0$

For $k = 0, 1, \ldots$, until convergence

$$\begin{align*}
r'_k &= (A A^T + I) r_k - |r_k| \\
\alpha_k &= \frac{(r_k, r'_k)}{(r'_k, r'_k)} \\
x_{k+1} &= x_k + \alpha_k A^T r_k \\
\lambda_{k+1} &= b_k + \alpha_k (r_k | - r_k) \\
r_{k+1} &= r_k - \alpha_k r'_k
\end{align*}$$

End

Notice that algorithm 3 has 4 steps less than algorithm 2 and correspondingly a lower computational cost, but slower convergence in general. Observation 1 does not apply, since $b_0 > 0$ and $\alpha_k > 0$.

**IV. Numerical Simulations**

In order to verify the convergence properties of the proposed batch mode algorithms, several simulations are performed. The performance of the proposed algorithms are compared with the Ho-Kashyap (HK) and its batch mode adaptive version (AHK) [4] and the conjugate gradient algorithm of [9] (called CGA). The experiment was performed for various linearly and non-linearly separable sets. The set chosen is the iris plants database from the UCI ML repository [14]. This is a frequently used benchmark dataset in the pattern recognition literature and contains three classes of 50 instances each, where each class refers to a type of iris plant. Each data point consists of four numeric, predictive attributes and the class. The attributes are sepal length, sepal width, petal length and petal width, all measured in cm. The three class labels, *iris setosa*, *iris versicolor* and *iris virginica* correspond to the three types of iris plant. It is known that *iris setosa* is linearly separable from the other two classes, but the latter are not linearly separable from each other even with respect to all four attributes. Though this a three-class problem, it can be converted to a two-class problem thus allowing the use of perceptron learning [9]. One way to do this is to convert it into 3 two class problems by making samples belonging to class $w_i$ ($1 \leq w_i \leq 3$) positive and those not belonging to it, negative. In the first case, samples belonging to *iris setosa* are marked as class +1 and those belonging to *iris versicolor* are assigned a class label –1. In the second case, samples belonging to *iris setosa* are again taken as belonging to the class +1 and those belonging to *iris virginica* are assigned a class label –1. In the third and last case, the *iris versicolor* samples are labeled as class +1 while the *iris virginica* samples are class –1.

For computational purposes (but not for the graphs in the figures below), the input vectors are augmented by adding an additional input of +1 for the bias. The negative class patterns are again marked as class +1 and those belonging to *iris virginica* are assigned a class label –1. In the third and last case, the *iris versicolor* samples are labeled as class +1 while the *iris virginica* samples are class –1.

Figures 3, 4 and 5 plot the separating hyperplanes obtained by using the algorithms 2, 3, HK, AHK and CGA. As the perceptron training problem can be solved by support vector machine (SVM), the obtained results are compared with the results of the least-square SVM (LS-SVM) of [15]. It can be noticed that, for the two linearly separable cases, all the considered algorithms classify correctly all the samples, but
only the algorithms 2 and HK reach the stopping criterion set as $\|r_k\| \leq 0.001$. The HK and AHK algorithms are very sensitive to the choice of the parameters $\rho_1$ and $\rho_2$, i.e., improper choice of the parameter values $\rho_1$ and $\rho_2$ would lead to poor performance. Then the choices $\rho_1 = 1.5$ and $\rho_2 = 1/\lambda_{\text{max}}$ ($\lambda_{\text{max}}$ is the highest eigenvalue of $A^T A$), as in [9], result in a residual vector with relatively high norm, as shown in table I where

$$\%CC = \frac{\text{Number of samples correctly classified}}{\text{Number of samples}} \times 100,$$

and $M$ is the margin distance defined as the minimum of the distances from the correctly classified samples to the separating hyperplane, the stopping criterion is $\|r_k\| \leq 0.001$ and the maximum number of iterations is equal to 1000.

The results in table I show that algorithm 2 returns a residual vector with the lowest norm, in linearly separable cases. The proposed algorithms present basically the same performance in the linearly non-separable case, except the AHK algorithm which fails, classifying all samples as belonging to the same class.

Notice that the computational cost of one iteration of algorithm 3 is lower than one of algorithm 2, but if the number of iterations of algorithm 3 is too high, its total computational cost and time can be higher than the total computational cost of algorithm 2. To illustrate this, we run the same algorithm 100 times with the same initial conditions and determine the mean time $t$ (in seconds) and the number of iterations. The stopping criterion is again set as $\|r_k\| \leq 0.001$. The results of this experiment are shown in table II. As expected, although algorithm 3 is simpler than algorithm 2, in some cases algorithm 2 converges more rapidly and therefore spends less mean time than the
algorithms 3, AHK and CGA.

Other numerical tests were performed using the sets “Iono-
sphere”, “BUPA Liver Disorders” and “Pima Indian Dia-
betes”, confirming Observation 1. The classification results
are shown in table III. More details on these three sets can
be found in [14].

The numerical results obtained show that the proposed
batch mode conjugate gradient algorithm presents equivalent
or better performance than the algorithms HK, AHK and 
CGN without using any empirical learning parameter and
any heuristic reinitialization of the direction vector.

V. CONCLUSIONS

We have presented conjugate gradient and steepest descent
algorithms for perceptron training in batch mode and proved
the asymptotic stability of the algorithms in theorems 1 and
2. Their performance have been compared with the
algorithms CGA [9], HK, AHK [4] and LS-SVM [15].

The HK algorithm uses the pseudo-inverse of the pattern
matrix in order to determine the solution vector. However the
determination of the pseudo-inverse become unstable as the
matrix \( A^T A \) becomes close to singular, on the other hand
the computational cost of this process increases highly for
large scale problems, i.e., separation of classes with large
number of attributes. The AHK algorithm does not calculate
the pseudo-inverse of the pattern matrix, but the empirical
choice of the learning parameters \( p_1 \) and \( p_2 \) is critical for
the convergence of the algorithm. Numerical results in [9]
and this paper show that improper choice of the learning
parameters values \( p_1 \) and \( p_2 \) lead to poor performance of
the AHK algorithm. Nagaraja and Bose [9] have presented
a conjugate gradient algorithm for the perceptron training,
however they use an heuristic technique based on conju-
gate gradient reinitialization [10] in order to ensure the
convergence of the algorithm. This is due to the fact that
the standard conjugate algorithm uses orthogonality of the
residues to simplify the formulas of the parameters necessary
for convergence [6]. In an finite precision implementation
the numerical errors could cause a loss of the orthogonality
of the residues, leading to a non-convergence of the method.
To solve this problem, this paper does not use any heuristic
technique, in contrast to [9], and proposes an analysis using
the linear structure of system (3) and the bilinear structure
of the dynamical system solvers (9) and (24) considered,
and the asymptotic stability of the algorithms is proved in
theorems 1 and 2. The proof of the positivity of the margin
vector \( b_0 \) found by algorithm 2 is a topic under investigation.

REFERENCES


training,” IEEE Trans. on Neural Networks, vol. 3, no. 1, pp. 51–61,


solving linear systems,” Journal of Research of the National Bureau

control liapunov functions,” IEEE Signal Processing Letters, vol. 13,

conjugate gradient algorithm for linear constraints,” in Proceeding
of Joint Automatic Control Conference, vol. 19-21, June 1974,
Constraints.pdf

algorithm for perceptron training,” Neurocomputing, vol. 69, pp. 368–
386, 2006.


systems with feedback control,” in Proc. 42nd IEEE Conference on
Decision and Control, Maui, Hawaii, USA, December 2003, pp. 2374–
2380.

[12] ——, Control Perspectives on Numerical Algorithms and Matrix Pro-


http://www.ics.uci.edu/~mlearen/MLRepository.html

classifiers via gradient systems with discontinuous righthand sides,”