CISC 371, Fall 2024
Practice for Quiz #3:
Artificial Neural Networks and Nonlinear Least Squares
Recommend Completion: 4PM on Monday October 28, 2024

Please read the details and instructions carefully before you begin to work on the two problems.

Statement of Academic Integrity

Learning Outcomes

On completion, a successful student will be able to:

- Specify a multilayer neural network as a problem in unconstrained optimization
- Compute the objective value and gradient terms of a neural network for a given input and for batches of inputs
- Assess neural network classification by comparison to data labels
- Specify a 3D Fermat-Weber problem using anchor points
- Solve a nonlinear least squares problem using `MATLAB` functions

Preparation

You should create a working directory, named as you prefer. Copy all of the provided code and data files into this directory. When you run the `p3q1_00000000` file, the code will create and use a `MATLAB` neural network, then fail because of student code needs to be modified. You should understand this code before attempting the questions. When you run the `p3q2_00000000` file, the code will execute and report incorrect values.

Question 1: Two-Layer Neural Network For Supervised Learning

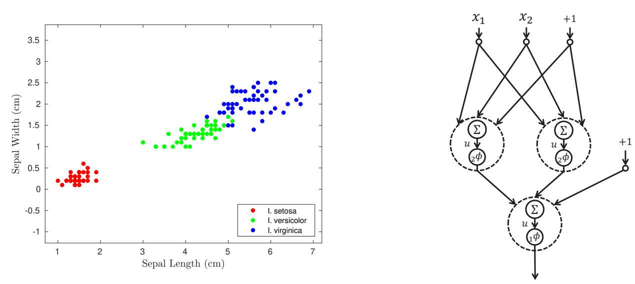
This question will use your code for steepest descent, with a fixed stepsize, to train a two-layer neural network for one of the earliest data sets that is still in use. It was described by Sir Ronald Fisher, who is a founder of both statistics and genetics. His data[1], gathered from the Gaspé Peninsula in Quebec, were measurements of 150 Iris flowers. We will use two of the measurements, which are the length and width of the sepals or flower coverings, so we have 150 observations of size 2. The additional attribute is the species: *I. setosa*, *I. virginica*, and/or *I. versicolor*. The code

[1] Fisher RA. "The use of multiple measurements in taxonomic problems", *Ann Eugen* 7(2):179-188; 1936

requires the use of a global variable, which is typical when using a `MATLAB` builtin function (such as `fminunc`).

A sample implementation of a single artificial neuron, or single cell, has been provided by the instructor in the `iristest.m` file. This will load Fisher's Iris data from a repository, select the species *I. versicolor* as the relevant feature, and use steepest descent to separate the data into the labeled sets. To perform the computations, you will need to copy and paste your own code for `steepfixed.m` into the starter code.

The goal for this question is to compute classifications for a multi-layer neural network that is trained on the provided data. The data and network are illustrated Figure 1.



(A) **(B)**

Figure 1: Data and a neural network for Fisher's Iris data. (A) Vectors in green represent data for *I. Versicolor*. (B) A network with two logistic neurons in the hidden layer and an output neuron.

The "starter" function for this question will first plot the data; next, it will use a `MATLAB` toolbox to perform the classifications. The provided code will then fail because the "hidden" layer does not compute the response correctly.

When you have correctly modified the provided code, the "starter" function should plot the classifications and these should be a better match to the initial labels of the data.

You should evaluate the accuracy of your results and the incorrect matches by producing and interpreting a confusion matrix.

Tasks

The tasks for this question are:

- Modify the code for function `annsteepfixed`, in the starter file `p3q1_00000000`, to compute steepest descent; this is straightforward duplication of your previous code and is indicated in one place
- Modify the code for function `anngradient` to compute the gradient of the network objective function; this requires understanding back-propagation and is indicated in two places
- Optionally, investigate convergence behavior by varying the learning rate and by plotting the weights of the neurons

Question 2: Nonlinear Least Squares For GPS Receivers

You must formulate this question as a nonlinear least squares (NLS) problem. We recommend – but do not require – that you use the `MATLAB` optimization function `lsqnonlin`.

In the course notes, a simplified form of the equations for locating a receiver of Global Positioning System (GPS) transmissions is presented. In this question, you will use data from six GPS satellites to find the location of the receiver.

The "starter" code brings the data into the current workspace as the variables `xgps` and `ygps`. The satellite locations are in a Cartesian coordinate frame that has Earth center at the origin, with distances measured in meters; the data are structured as a design matrix so each row is the transpose of a satellite's position vector. The $\vec{y}$ values are the pseudo-ranges, also measured in meters.

Your problem is to formulate an unweighted NLS to locate the receiver in Cartesian coordinates. You will need to create a function – preferably an anonymous function – that calculates the residual error that is to be minimized. You should test your function with at least two initial estimates of the receiver locations. It should not be surprising to discover that this function has multiple local minimizers and that the result depends on the initial estimate.

Hint to distinguish the solutions, for which you may use web-based calculators: where are you?

In terms of geodesy, which is the science of measuring Earth, your numerical answer will be in ECEF coordinates. Optionally, you can use `MATLAB` functions to find an ellipsoid that approximates the Earth with the World Geodetic System 1984 (WGS84) values, and convert from ECEF coordinates to latitude, longitude, and altitude above the nominal model ellipsoid. The altitude takes into account local terrain variations from the idealized mean ocean surface level in WGS84. This may help you in locating your results on a global map.

Tasks

The tasks for this question are:

- Modify the provided code to compute a vector-valued function that is of the form $\vec{g}(t)$; this can be anonymous function or code that is after the "starter" code, and is indicated in one place
- Solve the NLS problem by, for example, using the `MATLAB` function `lsqnonlin`; this is indicated in one place
- Use at least two initial estimates of the receiver location that produce distinct minimizers
- Interpret the results of these computations

Policies

- You must complete this question individually
- Although you are allowed to discuss the question with other students, you must write your own answers and `MATLAB` code
- The syllabus standards apply to this assignment