TRINITY COLLEGE DUBLIN

School of Computer Science and Statistics

**Week 6 Assignment** CS7CS2 Optimisation for Machine Learning

**Rules of the game:**

- **Its ok to discuss with others, but do not show any code you write to others. You must write answers in your own words and write code entirely yourself. All submissions will be checked for plagiarism.**
- **Reports must be typed (no handwritten answers please) and submitted as a separate pdf on Blackboard (not as part of a zip file please).**
- **Important: For each problem, your primary aim is to articulate that you understand what you're doing - not just running a program and quoting numbers it outputs. Long rambling answers and "brain dumps" are not the way to achieve this. If you write code to carry out a calculation you need to discuss/explain what that code does, and if you present numerical results you need to discuss their interpretation. Generally most of the credit is given for the explanation/analysis as opposed to the code/numerical answer. Saying "see code" is not good enough, even if code contains comments. Similarly, standalone numbers or plots without further comment is not good enough.**
- **When your answer includes a plot be sure to (i) label the axes, (ii) make sure all the text (including axes labels/ticks) is large enough to be clearly legible and (iii) explain in text what the plot shows.**
- **Include the source of code written for the assignment as an appendix in your submitted pdf report. Also include a separate zip file containing the executable code and any data files needed. Programs should be running code written in Python i.e. so that we can unzip your submission and just directly run it to check that it works. Keep code brief and clean with meaningful variable names etc.**
- **Reports should typically be not more than aboput 5 pages, with 10 pages the absolute upper limit (excluding appendix with code). If you go over 10 pages then the extra pages will not be marked.**

## Obtaining Functions

- Download the loss function to use in the assignment from `https://www.scss.tcd.ie/Doug.Leith/CS7DS2/week6.php`. Important: You must fetch your own copy of the functions, do not use the dataset downloaded by someone else.
- Please cut and paste the functions and include in your submission.

.

## Assignment

(a) (i) Write python code that implements mini-batch Stochastic Gradient Descent(SGD). Explain how the code implements the SGD algorithm. The code should allow use of a constant step size, Polyak, RMSProp, Heavy Ball and Adam steps (you already wrote code for all of these in the last assignment, so you can reuse that here).

(ii) You'll be looking at minimising the function that you downloaded. First generate the training data points $T$ by calling function generate_trainingdata(). The loss function $f(x, N)$ has the form:

$$f(x, N) = \sum_{w \in N} loss(x, w)$$

where (i) $x$ is a vector with two elements that is to be chosen to as to minimise function f() and (ii) $N$ is a subset/mini-batch of the training data over which

to calculate the loss. Plot a wireframe and a contour plot of $f$ for $N = T$ (i.e. using the full training data). You'll need to select a suitable range of $x$ values over with to plot, explain your choice.

(iii) Use either sympy or finite differences to calculate the derivative of $f$. Give the code you use for this, and explain its operation.

(b) (i) Use gradient descent with a constant step-size to minimise the loss function starting from initial $x = [3, 3]$. You'll need to select an appropriate step size, and of course to explain your choice. Show on a contour plot how $f$ and $x$ change over time.

(ii) Now use mini-batch SGD with a mini-batch size of 5 and the same step size as in (i) to minimise the loss function starting from initial $x = [3, 3]$. Run the SGD several times and plot how $f$ and $x$ change over time. Comment on (i) how the behaviour varies from run to run and (ii) compares to that of the gradient descent you used in (i).

(iii) Keeping the step size fixed, vary the mini-batch size (you'll need to select a suitable range of values). What is the effect on how $f$ and $x$ change over time? Does the point to which $x$ converges change? If so, why? What does this suggest regarding the regularising effect, if any, of the SGD gradient "noise"?

(iv) Keeping the mini-batch size fixed at 5, vary the step size. What is the effect on how $f$ and $x$ change over time? Compare with what you saw in (iii).

(c) Now use mini-batch SGD to minimise the loss function with the following choices of step:

(i) Polyak step size

(ii) RMSProp

(iii) Heavy Ball

(iv) Adam

You'll need to select appropriate parameters for (ii)-(iv), and to explain your choices. Discuss how these different algorithms affect how $f$ and $x$ change over time. How is their behaviour affected by the choice of mini-batch size? You can use the constant step size results from (b) as a baseline for comparison.