TRINITY COLLEGE DUBLIN

School of Computer Science and Statistics

Final Assignment 2024-25 CS7DS2 Optimisation for Machine Learning

- **You must do this assignment entirely yourself - you must not discuss or collaborate on the assignment with other students in any way, you must write answers in your own words and write code entirely yourself. If you use any online or other external content in your report you should take care to cite the source (that includes use of code assistants, chat GPT and the like). It is mandatory to complete the declaration that the work is entirely your own and you have not collaborated with anyone - the declaration form is available on Blackboard. All submissions will be checked for plagiarism.**
- **Reports must be typed and submitted as a separate pdf on Blackboard (not as part of a zip file).**
- **Include the source of code written for the assignment as an appendix in your submitted pdf report (the code itself as plain text, not a screenshot, so the plagiarism checker can run on it). Failure to do this will lead to a 50% reduction in your mark. Also include a separate zip file containing the executable code and any data files needed. Programs should be running code written in Python. Keep code brief and clean with meaningful variable names etc.**
- **Important: Your primary aim is to articulate that you understand what you're doing - not just running a program and quoting numbers it outputs. Generally most of the credit is given for the explanation/analysis as opposed to the code/numerical answer.**
- **Reports should typically be about 5 pages, with 10 pages the upper limit (excluding appendix with code).**

Assignment

1. You task is to investigate the use of Polyak's adaptive step size in mini-batch stochastic gradient descent (SGD).

   Suppose we have a model $y = F_\theta(x)$ that takes vector x as input and outputs value y. The model parameters are $\theta = (\theta_1, \cdots, \theta_n)$ and we have training data $\{(x^{(i)}, y^{(i)}), i = 1, \cdots, N\}$. To train the model the function f to be minimized has the form $f(\theta) = \sum_{i=1}^{N} \ell(x^{(i)}, y^{(i)}; \theta)$ where ℓ is a loss function, e.g. the square error $(y^{(i)} - F_\theta(x^{(i)}))^2$. The gradient is vector $\nabla f(\theta) = (\frac{\partial f}{\partial \theta_1}(\theta), \ldots, \frac{\partial f}{\partial \theta_n}(\theta))$ with $\frac{\partial f}{\partial \theta_j}(\theta) = \sum_{i=1}^{N} \frac{\partial \ell_\theta}{\partial \theta_j}(x^{(i)}, y^{(i)})$, $j = 1, \cdots, n$. Polyak's choice of step is $\alpha = \frac{f(\theta) - f^*}{\nabla f(\theta)^T \nabla f(\theta) + \epsilon}$, $\nabla f(\theta)^T \nabla f(\theta) = \sum_{j=1}^{n} \frac{\partial f}{\partial \theta_j}(\theta)^2$ and $f^* = \min_\theta f(\theta)$. In practice, we set $f^* = 0$.

   The paper `https://proceedings.mlr.press/v130/loizou21a/loizou21a.pdf` proposes extending Polyak's choice of step size to mini-batch SGD by letting $f_{\mathcal{N}}(\theta) = \sum_{i \in \mathcal{N}} \ell(x^{(i)}, y^{(i)}; \theta)$, where set $\mathcal{N}$ contains the indices of the training data points in the current mini-batch (so $\mathcal{N}$ changes at every SGD iteration), and choosing the step size:

   $$\alpha = \frac{f_{\mathcal{N}}(\theta) - f^*_{\mathcal{N}}}{\nabla f_{\mathcal{N}}(\theta)^T \nabla f_{\mathcal{N}}(\theta) + \epsilon}$$

   where $\nabla f_{\mathcal{N}}(\theta)^T \nabla f_{\mathcal{N}}(\theta) = \sum_{j=1}^{n} \frac{\partial f_{\mathcal{N}}}{\partial \theta_j}(\theta)^2$, $\frac{\partial f_{\mathcal{N}}}{\partial \theta_j}(\theta) = \sum_{i \in \mathcal{N}} \frac{\partial \ell_\theta}{\partial \theta_j}(x^{(i)}, y^{(i)})$ and $f^*_{\mathcal{N}} = \min_\theta f_{\mathcal{N}}(\theta)$. In practice, we set $f^*_{\mathcal{N}} = 0$.

(a) Using pytorch modify the mini-batch SGD implementation to use this variant of Polyak's step size. You should to do this yourself, do not use code from the internet or wherever. In your report give enough detail to allow a reader to understand how your code works (but try to keep it brief, and omit boring boiler-plate code). Design and implement tests to verify that your code works correctly, explain how you chose the tests and summarize the results. [10 marks]

(b) Generate some synthetic noisy training data for a linear regression model, similarly to what we did in the lectures. Use mini-batch SGD to train the model using (i) constant step size, (ii) Polyak's step size for a range of batch sizes and training data noise. Analyse the behaviour with constant step size vs Polyak's step size. For example, when close to the minimum the Polyak step size will probably increase (why?), if so then how does that interact with the SGD noise? How does the choice of mini-batch size affect the Polyak's step size? Choosing a good value of constant step size, how does the convergence rate compare with Polyak? Remember that since SGD involves randomness the results will be different each time it is run. [15 marks]

(c) Using the model and training data from the week 6 assignment compare SGD with constant step size vs Polyak's step size. Discuss. For example, do they both allow SGD to escape the local minimum? Do they both converge the same global minimum? Should the same mini-batch size be used for both, or different sizes, and why? [15 marks]

(d) Now use the transformer model from the ML module week 9 assignment. Train the model using SGD with constant step size, Polyak's step size and Adam and compare their performance. Remember that an important role of SGD noise is to help avoid overfitting, so be sure to evaluate that for constant step size, Polyak's step size and Adam. Do not just use the default values for the constant step size and Adam parameters but tune them so that these algorithms roughly work as well as they can [30 marks]

(e) Briefly investigate modifying Adam (or another method that uses momentum) to use the SGD Polyak step size. Don't spend too much time on this, but have a quick look using synthetic noisy training data for a linear regression model. [15 marks]

2. (a) Explain how line search can be used in gradient descent. Give one advantage and one disadvantage. Should line search be used with stochastic gradient descent? [5 marks]

(b) Describe how a change of variables can be used to enforce a constraint on the decision variables, illustrating with an example. [5 marks]

(c) Explain how a penalty can be added to the cost function so as to enforce a constraint on the decision variables. Illustrate with a brief example. [5 marks]