Optimization Algorithms
Coding Assignment 4

Sayantan Auddy & Marc Toussaint
Learning & Intelligent Systems Lab, TU Berlin
Marchstr. 23, 10587 Berlin, Germany

Winter 2024/25

The fourth coding assignment includes a single task:

a) Evaluate the impact of the initial step size and decay rate in stochastic gradient descent

Deadline: Wednesday 05.02.2025 at 11.55 pm.

To work on the assignment, first check that the remote `upstream` is pointing to our repository.

```
git remote -v
```

You should see two remotes, similar to:

```
origin git@git.tu-berlin.de:auddy/optalg-w24-public.git (fetch)
origin git@git.tu-berlin.de:auddy/optalg-w24-public.git (push)
upstream https://git.tu-berlin.de/lis-public/optalg-w24-public (fetch)
upstream https://git.tu-berlin.de/lis-public/optalg-w24-public (push)
```

Otherwise, follow the instructions in `coding-readme.pdf`.

Now, you can merge our repository with your fork. If you have only modified the files in `assignments/a1_###`, `assignments/a2_###` and `assignments/a3_###`, the merge should be automatic (git will use the recursive strategy and it will ask for a commit message). Otherwise, you have to make sure that, after the merge, all files outside `assignments/a1_###`, `assignments/a2_###` and `assignments/a3_###`, are exactly the same as in our repository.

```
git fetch upstream
git merge upstream/main
```

1 Solver: Stochastic Gradient Descent

Implement stochastic gradient descent $x' = x_k - \alpha_k \nabla f_i(x_k)$ using a learning rate with decay,

$$\alpha_k = \frac{\alpha_0}{(1 + \alpha_0 \lambda k)}, \tag{1}$$

where α_0, and λ are user-defined constants.

The input of the `solve` function is an object of a new class called `NLP_stochastic`, that represents a stochastic optimization problem of the form,

$$\min_x f(x) = \frac{1}{N} \sum_{i=1}^{N} f_i(x) \tag{2}$$

Importantly, you can only query the function value and gradient of one index at a time, i.e. $f_i(x), \nabla f_i(x)$ for some chosen i. `NLP_stochastic.evaluate_i` has two inputs: `x` (the variable) and `i` (the index of the cost function you want to query), and returns the value and gradient of $f_i(x)$. `NLP_stochastic.getNumSamples()` returns the value of N.

In your implementation, you need to choose an appropriate α_0 and λ, and a strategy/ordering to choose which index to query next. The solver should fulfill the following requirements:

- The number of calls to `NLP_stochastic.evaluate_i(x,i)` is limited to 10.000. The solver should return a solution before reaching the maximum number of queries. Using the number of queries as termination criteria is a valid solution.
- The returned solution x^* is close to the real optimum, with $||x^* - x_{\text{opt}}|| \leq 0.05$.

```
cd assignments/a4_sgd
python3 test.py
python3 play.py
```

Once you are done, stage, commit and push `assignments/a4_sgd/solution.py`.