Homework 2: Stochastic Gradient Descent

Contents

> **Warning**
>
> The submission of the homeworks has **NO** deadline. You can submit them whenever you want, on Virtuale. You are only required to upload it on Virtuale **BEFORE** your exam session, since the Homeworks will be a central part of the oral exam.
>
> You are asked to submit the homework as one of the two, following modalities:
>
> - A PDF (or Word) document, containing screenshoots of code snippets, screeshots of the results generated by your code, and a brief comment on the obtained results.
> - A Python Notebook (i.e. a `.ipynb` file), with cells containing the code required to solve the indicated exercises, alternated with a brief comment on the obtained results in the form of a markdown cell. We remark that the code **SHOULD NOT** be runned during the exam, but the student is asked to enter the exam with all the programs **already executed**, with the results clearly visible on the screen.
>
> Joining the oral exam with a non-executed code OR without a PDF file with the obtained results visible on that, will cause the student to be rejected.

Exercise 1: SGD vs GD on a Simple 1D Regression Problem

Consider the synthetic dataset

$$x^{(i)} = \frac{i}{N}, \qquad y^{(i)} = 2x^{(i)} + 1 + \varepsilon^{(i)}, \qquad \varepsilon^{(i)} \sim \mathcal{N}(0, 0.01),$$

with $N = 200$. We model the data with a linear function:

$$f_\Theta(x) = \Theta_0 + \Theta_1 x = \Theta^T \tilde{x},$$

if we define $\tilde{x} = [1, x]$ as we did during the lab session.

1. Implement the MSE loss:

$$\mathcal{L}(\Theta) = \frac{1}{N} \sum_{i=1}^{N} (f_\Theta(x^{(i)}) - y^{(i)})^2.$$

2. Implement **full GD** and **SGD** (mini-batch) using batch sizes:
   - $N_{\text{batch}} = 1$,
   - $N_{\text{batch}} = 10$,
   - $N_{\text{batch}} = 50$,
   - $N_{\text{batch}} = N$ (this recovers GD).
3. Plot for each method:
   - the loss curve (loss vs epoch),
   - the trajectory of parameters (Θ_0, Θ_1) in the 2D parameter space. This is similar to what you did in the previous homework: simply plot the value of $\Theta_0^{(k)}$ and $\Theta_1^{(k)}$ for all the ks in a 2-dimensional plot.
4. Discuss:
   - Why GD is smooth but slow for large N,
   - Why SGD is noisy but progresses faster,
   - How batch size affects the noise level and convergence stability.

Exercise 2: Variance of the Stochastic Gradient (1D Experiment)

Fix a parameter vector Θ, and repeatedly draw random mini-batches of the same size.

1. Choose batch sizes:

$$N_{\text{batch}} \in \{1, 5, 20, N\}.$$

2. At the **same** Θ, compute:

$$g_k = \nabla_\Theta \mathcal{L}(\Theta; \mathcal{M}_k)$$

   for 100 randomly sampled batches $\mathcal{M}_k$.

3. For each batch size, compute the empirical variance:

$$\text{Var}(g) = \frac{1}{100} \sum_{k=1}^{100} \|g_k - \bar{g}\|^2,$$

   where $\bar{g}$ is the average of the g_ks, defined as:

$$\bar{g} = \frac{1}{100} \sum_{k=1}^{100} g_k.$$

4. Plot the variance as a function of the batch size.
5. Comment:
   - Why the variance decreases with larger batches,

- Why SGD becomes more stable as N_{batch} increases,
- The trade-off between stability and computational cost.

Exercise 3: SGD in 2D

We now study SGD on the 2D non-convex function:

$$\mathcal{L}(\Theta_1, \Theta_2) = (\Theta_1^2 - 1)^2 + 10(\Theta_2 - \Theta_1^2)^2.$$

This function has:

- two valleys,
- multiple stationary points,
- strong curvature differences.

1. Treat $\Theta = (\Theta_1, \Theta_2)$ as a "parameter vector" updated by SGD:

   $$\Theta_{k+1} = \Theta_k - \eta\, g_k,$$

   where the ``gradient batch'' g_k is simulated by adding noise to the gradient:

   $$g_k = \nabla\mathcal{L}(\Theta_k) + \varepsilon_k, \qquad \varepsilon_k \sim \mathcal{N}(0, \sigma^2 I),$$

   where σ^2 is called **noise level** and represent the variance of the noise. Try different values of σ^2 to answer the following questions. *Note: σ^2 should always be lower than 1.*

2. Plot:

- level sets of $\mathcal{L}(\Theta_1, \Theta_2)$,
- trajectories of SGD for different noise levels and step sizes.

3. Discuss:
   - How noise helps escape shallow minima or bad regions,
   - How too much noise prevents convergence,

Exercise 4: An ML Project with SGD

For the final exercise, you will train a simple machine learning model using SGD on a real prediction task. To begin, download the Kaggle dataset: https://www.kaggle.com/datasets/mirichoi0218/insurance.

It contains approximately 1300 samples, and its associated task is to predict **individual medical insurance cost** based on numerical features:

- `age`
- `bmi`
- `children`

The task is:

$$\text{charges} \approx f_\Theta(\text{age}, \text{bmi}, \text{children}).$$

We use a simple linear model:

$$f_\Theta(x) = \Theta^T \tilde{x}.$$

1. Load & preprocess the dataset
   - Download `insurance.csv` from Kaggle.
   - Select numerical columns: `["age", "bmi", "children"]`.
   - Standardize each feature (mean 0, variance 1).
   - Standardize the target `"charges"`.
   - Add a bias column.
2. Consider the MSE loss:

$$\mathcal{L}(\Theta) = \frac{1}{N}\sum_{i=1}^{N}(\Theta^T\tilde{x}^{(i)} - \tilde{y}^{(i)})^2.$$

   Implement:

   - Full GD
   - SGD with batch sizes 1, 10, 50

   Use a fixed learning rate $\eta = 10^{-2}$ and a fixed number of epochs (equivalently, a fixed amount of maximum iterations for GD) of your choice.

3. Compare GD and SGD For each method:
   - Plot the loss vs epoch.
   - Plot the L2 norm of the **full gradient** $\|\nabla\mathcal{L}(\Theta_k)\|$ measured at the end of every epoch.
   - Report the final learned parameters.
4. Discuss:
   - Why GD gives a smooth curve and SGD oscillates.
   - Why larger batches reduce noise but cost more per iteration.

- Why all methods roughly converge to the same region.
- Why SGD is more suitable for large datasets, even when noisy.