Foundations of Deep Learning **Fall 2024**

Homework 2: Optimization, Loss Landscape and NTK

Lecturer: Aurelien Lucchi

The points of the best-two-out-of-three homeworks, including this one, will be contributed to the final score. The points of each problem in this exercise sheet are equally weighted. Period: 17 October 2024 18:00 - 21 November 2024 23:55 (Bern time).

Problem 1 (Stochastic gradient descent with momentum) (10 Points):

In the lecture you have already seen SGD, however in most machine learning applications, SGD is often used with *momentum*. In this problem wee will see multiple equivalent ways how the momentum method can be written. Consider the problem of minimizing a function $f : \mathbb{R}^d \to \mathbb{R}$, which can be written as

$$f(x) := \frac{1}{n}\sum_{i=1}^{n} f_i(x),$$

where each $f_i : \mathbb{R}^d \to \mathbb{R}$ is assumed to be convex and L_i-smooth. Furthermore, we require that the problem is well-posed, i.e. that $\arg\min f \neq \emptyset$ and that all f_i's are bounded from below. The momentum algorithm is then defined as follows:

Algorithm (Momentum). Let $\mathbf{x}^0 \in \mathbb{R}^d$ and $\mathbf{m}^{-1} = 0$, let $(\gamma_t)_{t\in\mathbb{N}} \subset]0, +\infty[$ be a sequence of step sizes, and let $(\beta_t)_{t\in\mathbb{N}} \subset [0, 1]$ be a sequence of momentum parameters. The **Momentum** algorithm defines a sequence $(\mathbf{x}^t)_{t\in N}$ satisfying for every $t \in N$

$$\mathbf{m}^t = \beta_t \mathbf{m}^{t-1} + \nabla f_{i_t}(\mathbf{x}^t), \tag{1}$$
$$\mathbf{x}^{t+1} = \mathbf{x}^t - \gamma_t \mathbf{m}^t \tag{2}$$

where $i_t \in \{1, \dots, n\}$ is sampled uniformly and i.i.d at each iteration.

We will now explore two other ways in which the momentum algorithm can be expressed.

a) The momentum method is often written in the *heavy ball* format, which is

$$\mathbf{x}^{t+1} = \mathbf{x}^t - \hat{\gamma}_t \nabla f_{i_t}(\mathbf{x}^t) + \hat{\beta}_t(\mathbf{x}^t - \mathbf{x}^{t-1}), \tag{3}$$

where $\hat{\beta}_t \in [0, 1]$ is another momentum parameter.
Show that (Momentum) and (3) are indeed equivalent, assuming $\gamma_{-1} = 1$ and $\mathbf{x}^{-1} = \mathbf{x}^0$. What is the relation between γ_t, β_t and $\hat{\gamma}_t, \hat{\beta}_t$?

b) Yet another equivalent algorithm is the *iterate-moving-average* (IMA) algorithm: start from $\mathbf{z}^{-1} = \mathbf{x}^0$ and iterate for $t \in \mathbb{N}$

$$\mathbf{z}^t = \mathbf{z}^{t-1} - \eta_t \nabla f_{i_t}(\mathbf{x}^t), \tag{4}$$
$$\mathbf{x}^{t+1} = \frac{\lambda_{t+1}}{\lambda_{t+1}+1}\mathbf{x}^t + \frac{1}{\lambda_{t+1}+1}\mathbf{z}^t. \tag{5}$$

Show that (Momentum) is equivalent to (IMA) in the sense that a sequence $(\mathbf{x}_t)_{t\in\mathbb{N}}$ generated by (Momentum) with parameters γ_t, β_t recovers (IMA), by choosing any parameters (η_t, λ_t) and a vector $\mathbf{z}^t$ satisfying

$$\beta_t\lambda_{t+1} = \frac{\gamma_{t-1}\lambda_t}{\gamma_t} - \beta_t, \quad \eta_t = (1 + \lambda_{t+1})\gamma_t, \quad \text{and} \quad \mathbf{z}^t = \mathbf{x}^{t+1} + \lambda_{t+1}(\mathbf{x}^{t+1} - \mathbf{x}^t).$$

Hint: Make use of the heavy ball formulation.

Problem 2 (Loss landscape in neural networks) (10 + 8 Points):

In this problem you will explore the implicit bias of the optimizer USAM towards flat minima (measured in terms of the trace of the Hessian) on the example of the *widening valley function*, defined as

$$L(\mathbf{u}, v) = \frac{1}{2}\|\mathbf{u}\|^2 v^2,$$

where $\|\cdot\|$ is the Euclidean norm, and $v \in \mathbb{R}, \mathbf{u} \in \mathbb{R}^d$.

a) Compute the gradient $\nabla L(\mathbf{u}, v)$, the Hessian matrix $\nabla^2 L(\mathbf{u}, v)$ and the Trace of the Hessian $\mathrm{Tr}(\nabla^2 L(\mathbf{u}, v))$.

b) Now consider Gradient Descent (GD), defined by the update rule

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \eta_k \nabla L(\mathbf{x}_k),$$

and the USAM optimizer, defined by the update rule

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \eta_k \nabla L(\mathbf{x}_k + \rho \nabla L(\mathbf{x}_k)),$$

where we concatenate the arguments into a new vector $\mathbf{x}_k = [\mathbf{u}_k, v_k] \in \mathbb{R}^{d+1}$, $\eta_k > 0$ is some learning rate schedule and $\rho > 0$ is another hyperparameter of USAM.
Write down the update rule for GD and USAM explicitly for the given loss function of the widening valley and simplify the update rule as much as possible.
Describe in your own words how the update rules of GD and USAM differ from each other.

c) **Bonus: Jupyter notebook** Now open the attached Jupyter notebook and do the following tasks:

(i) Implement functions to compute the loss, the gradient, the Hessian and the trace of the Hessian of the widening valley

(ii) Implement GD and USAM. A skeleton function is already provided.

(iii) Run GD and USAM for different step sizes η and values of ρ (for USAM) and 20 different random initializations $\mathbf{x}_0 = [\mathbf{u}_0, v_0] \sim \mathcal{N}(0, \mathbf{I}_{d+1})$. A skeleton function is already provided

(iv) Plot the trace of the Hessian and the final iterate on the loss surface. What conclusions do you draw from this?

Problem 3 (NTK) (10 Points):
Consider a two-layer neural network $f : \mathbb{R}^d \to \mathbb{R}$ with weights $\mathbf{v} \in \mathbb{R}^m, \mathbf{W} \in \mathbb{R}^{m\times d}$:

$$f(\mathbf{v}, \mathbf{W}; \mathbf{x}) = \frac{1}{\sqrt{m}} \sum_{r=1}^{m} v_r \sigma(\mathbf{w}_r^\top \mathbf{x}) = \frac{1}{\sqrt{m}} \mathbf{v}^\top \sigma(\mathbf{W}\mathbf{x})$$

where $\mathbf{x} \in \mathbb{R}^d$ is the input, $\mathbf{W} = (\mathbf{w}_r)_{r=1}^m \in \mathbb{R}^{m\times d}$ with $\mathbf{w}_r \in \mathbb{R}^d$ and $\mathbf{v} = (v_r)_{r=1}^m$ are the weights and σ is the activation function. We are given a training dataset $(\mathbf{x}_i)_{i=1}^N$ where each $\mathbf{x}_i \in \mathbb{R}^d$ is a feature vector such that $\|\mathbf{x}_i\| = 1$.

a) Compute the derivatives $\frac{\partial f(\mathbf{v},\mathbf{W};\mathbf{x})}{\partial v_r} \in \mathbb{R}$ and $\frac{\partial f(\mathbf{v},\mathbf{W};\mathbf{x})}{\partial \mathbf{w}_r} \in \mathbb{R}^d$, and then show that

$$\frac{\partial f(\mathbf{v}, \mathbf{W}; \mathbf{x})}{\partial \mathbf{v}} = \frac{1}{\sqrt{m}} \sigma(\mathbf{W}\mathbf{x});$$
$$\frac{\partial f(\mathbf{v}, \mathbf{W}; \mathbf{x})}{\partial \mathbf{W}} = \frac{1}{\sqrt{m}} (\sigma'(\mathbf{W}\mathbf{x}) \odot \mathbf{x})\mathbf{v}^\top,$$

where $\odot$ is the Hadamard product.

b) Recall the definition of the neural tangent kernel (NTK)

$$\begin{aligned}\mathbf{H}_{ij}(t) &= \left\langle \frac{\partial f(\mathbf{v}, \mathbf{W}; \mathbf{x}_i)}{\partial \mathbf{v}}, \frac{\partial f(\mathbf{v}, \mathbf{W}; \mathbf{x}_j)}{\partial \mathbf{v}} \right\rangle + \left\langle \frac{\partial f(\mathbf{v}, \mathbf{W}; \mathbf{x}_i)}{\partial \mathbf{W}}, \frac{\partial f(\mathbf{v}, \mathbf{W}; \mathbf{x}_j)}{\partial \mathbf{W}} \right\rangle_F \\ &= \left\langle \frac{\partial f(\mathbf{v}, \mathbf{W}; \mathbf{x}_i)}{\partial \mathbf{v}}, \frac{\partial f(\mathbf{v}, \mathbf{W}; \mathbf{x}_j)}{\partial \mathbf{v}} \right\rangle + \sum_{r=1}^{m} \left\langle \frac{\partial f(\mathbf{v}, \mathbf{W}; \mathbf{x}_i)}{\partial \mathbf{w}_r}, \frac{\partial f(\mathbf{v}, \mathbf{W}; \mathbf{x}_j)}{\partial \mathbf{w}_r} \right\rangle\end{aligned}$$

where $\langle\cdot\rangle$ and $\langle\cdot\rangle_F$ denotes the vector inner product and the Frobenius inner product respectively. As the network width $m \to \infty$, the matrix $\mathbf{H}(0)$ tends to its limit: $\mathbf{H}^\infty(0) = \lim_{m\to\infty} \mathbf{H}(0) = \mathbb{E}_{\mathbf{W},\mathbf{v}}[\mathbf{H}(0)]$ where the expected value is taken over the random initialization of $\mathbf{v}, \mathbf{W}$. Assume that the weights $\mathbf{v}, \mathbf{w}_r$ are initialized by i.i.d. standard Gaussian vectors from $\mathcal{N}(0, \mathbf{I})$, i.e. at time $t = 0$, each entry of $\mathbf{v}, \mathbf{W}$ is drawn independently from the univariate Gaussian distribution $\mathcal{N}(0, 1)$. Prove that the matrix $\mathbf{H}$ has the expression:

$$\mathbf{H}_{ij}^\infty(0) = \mathbb{E}_{\mathbf{w}\sim\mathcal{N}(0,\mathbf{I})} \left[\sigma(\mathbf{w}^\top \mathbf{x}_i)\sigma(\mathbf{w}^\top \mathbf{x}_j)\right] + \mathbf{x}_i^\top \mathbf{x}_j \mathbb{E}_{\mathbf{w}\sim\mathcal{N}(0,\mathbf{I})}[\sigma'(\mathbf{w}^\top \mathbf{x}_i)\sigma'(\mathbf{w}^\top \mathbf{x}_j)].$$

c) Now consider a family of activation functions $\sigma(z) = \sigma(z; n) = z^n \cdot \mathbf{1}_{z\geq 0}$. [1] It is known that

$$\mathbb{E}_{\mathbf{w}\sim\mathcal{N}(0,\mathbf{I})} \left[\sigma(\mathbf{w}^\top \mathbf{x}_i)\sigma(\mathbf{w}^\top \mathbf{x}_j)\right] = \frac{(-1)^n}{2\pi} (\sin\theta)^{2n+1} \left(\frac{1}{\sin\theta} \frac{\partial}{\partial\theta}\right)^n \left(\frac{\pi - \theta}{\sin\theta}\right) \tag{6}$$

where $\theta = \arccos \mathbf{x}_i^\top \mathbf{x}_j$.

i) For $n = 1$, using Eq. (6), show that

$$\mathbf{H}_{ij}^\infty(0) = \frac{1}{2\pi} \sin\theta + \frac{1}{\pi}(\pi - \theta)\cos\theta.$$

ii) For $n = 2$, using Eq. (6),[2] show that

$$\mathbf{H}_{ij}^\infty(0) = \frac{1}{2\pi} \left[5\sin\theta\cos\theta + (\pi - \theta)(1 + 4\cos^2\theta)\right].$$

[1] For $n = 0$, it is the step function; for $n = 1$, it is ReLU.

[2] Caution: $\left(\frac{1}{\sin\theta}\frac{\partial}{\partial\theta}\right)^2 = \left(\frac{1}{\sin\theta}\frac{\partial}{\partial\theta}\right) \cdot \left(\frac{1}{\sin\theta}\frac{\partial}{\partial\theta}\right)$, not $\left(\frac{1}{\sin^2\theta}\frac{\partial^2}{\partial\theta^2}\right)$!