Exam: Optimization for Machine Learning

M2 IASD/MASH – Thursday, December 15, 2022

This is an open book exam, meaning you can consult any written or printed material. Electronic devices are prohibited. You must justify all of your answers. If you cannot solve a question, do not hesitate to admit its result in order to solve the next ones.

Ex. 1 — ***(gradient descent)***

1. We consider $B \in \mathbb{R}^{n\times d}$ and $L : \mathbb{R}^n \to \mathbb{R}$ a differentiable function. We denote $f(x) := L(Bx)$. Write and prove a formula for $\nabla f(x)$ as a function of $\nabla L(Bx)$.
2. We consider the logistic classification loss $L(z) = \sum_{k=1}^n \ell(z_k)$ where $\ell(s) := \log(1+e^s)$. Compute $\ell'(s)$ and then give the expression for $\nabla L(z)$.
3. For data $(a_i, y_i) \in \mathbb{R}^d \times \{-1, 1\}$, where $i = 1, \ldots, n$, the logistic classification problem minimizes
$$f(x) := \sum_{i=1}^n \ell(-y_i \langle a_i,\, x\rangle).$$
Define a matrix B so that f has the same form as in question 1. Using this matrix, how would you compute ∇f ?
4. Given a differentiable function f, we consider for $t \in \mathbb{R}$, $g(t) := f(x - t\nabla f(x))$. Compute $g'(t)$ (you can for instance do a Taylor expansion of $g(t+\epsilon)$ for small ϵ). Show that if x is not a stationary point of f (i.e. $\nabla f(x) \neq 0$), then $g'(0) < 0$. What does this property tell us about gradient descent ?

Ex. 2 — ***(automatic differentiation)***

1. Consider a function $f\colon \mathbb{R}^2 \to \mathbb{R}^5$. Is it better to use forward-mode or reverse-mode autodiff? Explain why.
2. Consider the function $g\colon \mathbb{R} \to \mathbb{R}$ defined as $y = g(x) := \sqrt{x^2+1} \times \log(x^2+1)$. Decompose g as a sequence of elementary steps (forward pass).
3. Compute $g'(x) = \partial g(x)/\partial x$ using reverse-mode differentiation (backward pass).
4. Consider the function $h(a) = \min_{x\in\mathbb{R}} \frac{a}{2}x^2 + bx + c$, where $a > 0$. Denote by $x^\star(a)$ the solution of the minimum. 1) Using Danskin's theorem, give the expression of $h'(a)$ in terms of $x^\star(a)$. 2) Derive $x^\star(a)$ and deduce from it a closed-form for $h(a)$. 3) Use it to derive $h'(a)$ and check that your answer matches what you obtained in 1).

Ex. 3 — ***(stochastic gradient descent)***

In this exercise, we consider a stochastic optimization problem of the form

$$\text{minimize}_{x\in\mathbb{R}^d} f(x) = \frac{1}{n}\sum_{i=1}^n f_i(x), \tag{1}$$

where every f_i is $\mathcal{C}^1$ depends solely on the ith data point in a dataset of size n. We also assume that the objective function f is strongly convex, and we let f^* denote its minimal value.

We assume that n is so large that using the entire dataset at once is not possible in practice. The goal of the exercise is to investigate the use of *dynamic batch stochastic gradient* techniques.

1. Write down an iteration of a batch stochastic gradient method applied to problem (1). How does the batch size need to be chosen for the method to correspond to gradient descent?
2. In the rest of the exercise, we let n_k denote the batch size at iteration k, and we suppose that $n_k = \lceil \tau^k \rceil$ for some $\tau > 1$. Such an approach is called *dynamic batch stochastic gradient.* Recall the notion of epoch, and use that notion to measure the cost of the kth iteration of such a method.
3. Explain how using an increasing batch size leads to stochastic gradient estimates that are less noisy as the algorithm progresses.
4. Name another technique that can be used to reduce the variance of stochastic gradient estimates.

5. Under appropriate assumptions on the stepsize, it can be shown that there exists a quantity $\tau \in (0, 1)$ such that the iterates computed by the dynamic batch stochastic gradient method

$$\mathbb{E}\left[f(x_K) - f^*\right] \leq c\tau^{-K} \tag{2}$$

for some constant $c > 0$.
   a) Using (2), bound the worst-case number of iterations needed to satisfy $\mathbb{E}\left[f(x_K) - f^*\right] \leq \epsilon$, where $\epsilon = c\tau^{-\ell}$ for some $\ell \in \mathbb{N}$.
   b) By expressing the result from the previous question in terms of epochs, show that the algorithm satisfies $\mathbb{E}\left[f(x_K) - f^*\right] \leq \epsilon$ in at most $\mathcal{O}\left(\frac{1}{n\epsilon}\right)$ epochs.
   You may use that $\lceil t \rceil \leq 2t$ *and* $\sum_{k=0}^{m} t^k = \frac{t^{m+1}-1}{t-1}$ *for any* $t > 1$.
6. Gradient descent applied to the problem (1) satisfies $f(x_K) - f^* \leq \epsilon$ after at most $\mathcal{O}\left(\log(1/\epsilon)\right)$ iterations.
   a) Given that n is very large by assumption, explain how that result can be considered worse that the one obtained in question 5.
   b) Find a setting for the dynamic batch method for which the results will actually be comparable to, or even in favor of, gradient descent.
7. Consider an instance of problem (1) on which we apply the batch stochastic gradient technique described at the beginning of the exercise. Suppose that we notice that the components of the stochastic gradient estimates differ by orders of magnitude. What advanced stochastic gradient method would you try in that setting, and why?

Ex. 4 — *(constrained optimization)*
Let $M \in \mathbb{R}^{p\times p}$ be a positive semi-definite matrix, and consider the minimization problem

$$\min_{x\in\mathbb{R}^p} \left(-\frac{1}{2}\langle Mx,\, x\rangle\right) \quad \text{s.t. } \|x\|^2 \leq 1, \tag{3}$$

where $\|\cdot\|$ is the usual Euclidean norm.
To fix notation, we write $A \overset{\text{def}}{=} \left\{x \in \mathbb{R}^p \;;\; \|x\|^2 \leq 1\right\}$ and $f\colon x \mapsto \left(-\frac{1}{2}\langle Mx,\, x\rangle\right)$.

1. a) Prove that there exists a solution to (3).
   b) Prove that the function $x \mapsto \left(\frac{1}{2}\langle Mx,\, x\rangle\right)$ is convex. Is the function f convex?
2. Let $x^\star \in A$ be a minimizer. Write the necessary optimality conditions and deduce the KKT conditions that $x^\star$ should satisfy.
   Justify your answer carefully! For the first part, you may distinguish between the cases $\|x\| < 1$ *and* $\|x\| = 1$.
3. Write the projected gradient descent algorithm (with fixed stepsize $\tau > 0$) for this problem. ***(Bonus:*** *Does this remind you of some other algorithm, possibly applied to a different matrix?)*
4. a) When is $x = 0$ a minimizer? Prove that, in that case, it is possible to find another minimizer $x^\star \in A$ such that $\|x^\star\| = 1$.
   b) If 0 is not a minimizer, prove that *every* minimizer $x^\star \in A$ satisfies $\|x^\star\| = 1$.
   c) Let $x^\star \in A$ be a minimizer, with $\|x^\star\| = 1$. Deduce from the KKT conditions that $x^\star$ is an eigenvector of M.

   From now on, we consider $(v_i)_{1\leq i\leq p}$, an orthonormal basis of eigenvectors of M, corresponding to the eigenvalues $(\lambda_i)_{1\leq i\leq p}$. We assume for simplicity that $0 < \lambda_1 < \ldots < \lambda_p$.
5. a) Let $x \in A$ be such that the KKT conditions hold at x. Is x necessarily a minimizer?
   Hint: *Consider the points* $x = \pm v_i$, $1 \leq i \leq p$. *Which ones satisfy the KKT conditions? Which ones are minimizers?*
   b) Fix an initial point $x_0 \in A$, and define $(x_t)_{t\in\mathbb{N}}$ by the projected gradient descent algorithm, with fixed stepsize $\tau > 0$. Do we always have

$$\left(\lim_{t\to\infty} f(x_t) = \min_{x\in A} f(x)\right)?$$

   Hint: *Consider again the points* $x = \pm v_i$, $1 \leq i \leq p$.

c) Assume that $\|x_0\| \geq 1$. Prove by induction that

$$\forall t \in \mathbb{N} \setminus \{0\}, \quad x_t = \frac{(\mathrm{Id} + \tau M)^t x_0}{\|(\mathrm{Id} + \tau M)^t x_0\|}. \tag{4}$$

Deduce that, if $x_0 = \sum_{i=1}^p \alpha_{0,i} v_i$ with $\alpha_{0,p} \neq 0$ and $\|x_0\| \geq 1$, then $x_t = \sum_{i=1}^p \alpha_{t,i} v_i$, with

$$\forall i \in \{1, \dots, p-1\}, \lim_{t \to \infty} \alpha_{t,i} = 0 \quad \text{and} \quad \lim_{t \to \infty} \alpha_{t,p} = 1. \tag{5}$$

d) Deduce that, for *almost every* $x_0 \in A$,

$$\lim_{t \to \infty} f(x_t) = \min_{x \in A} f(x). \tag{6}$$

Ex. 5 — *(proximal gradient and regularization)* In this exercise, we revisit the proximal operator and the proximal gradient method on a specific problem. Given a point $w \in \mathbb{R}^d$, we consider

$$\text{minimize}_{x \in \mathbb{R}^d} \ \|x\|_1 + \frac{1}{2\alpha}\|x - w\|_2^2, \tag{7}$$

where $\|x\|_1 = \sum_{i=1}^d |[x]_i|$, $\|x\|_2^2 = \sum_{i=1}^d [x]_i^2$ and $\alpha > 0$.

1. Using the properties of $x \mapsto \|x\|_1$, explain why the objective function of (7) cannot be optimized by gradient-type techniques.
2. Justify that problem (7) and

$$\text{minimize}_{x \in \mathbb{R}^d} \ \alpha\|x\|_1 + \frac{1}{2}\|x - w\|_2^2 \tag{8}$$

have the same solution set (argmin). Since both functions are strongly convex, what can be said about this solution set?

3. Write down an optimality condition for problem (7).
4. In this question, we view problem (7) as computing a proximal operator.
   a) Using the definition of the proximal operator, write down the solution of problem (7) as the value of a proximal operator of a certain function.
   b) By repeatedly solving instances of problem (7) using the last solution found as w, what algorithm do we obtain?
5. In this question, we view problem (8) in a composite form, where $\frac{1}{2}\|x - w\|_2^2$ is a data-fitting term and $\alpha\|x\|_1$ is a regularization term.
   a) What is the purpose of such a regularization term? Why is it computationally worth using?
   b) Write down an iteration of proximal gradient applied to this problem with $x_k = w$ and stepsize α. What do you observe then? Is it to be expected?

Ex. 6 — *(large-scale and distributed optimization)* In this exercise, we consider a regularized optimization problem of the form

$$\text{minimize}_{x \in \mathbb{R}^d} \ f(x) + \frac{\lambda}{2}\|Ax\|_2^2, \tag{9}$$

where $f : \mathbb{R}^d \to \mathbb{R}$ is continuously differentiable, $\|v\|_2^2 = \sum_{j=1}^d [v]_j^2$ for any $v \in \mathbb{R}^d$ and $A \in \mathbb{R}^{d \times d}$ is the second-difference matrix defined by

$$A_{ij} = \begin{cases} 1 & \text{if } j = i+1 \text{ or } j = i-1 \\ -2 & \text{if } j = i \\ 0 & \text{otherwise.} \end{cases}$$

Such problems arise when the vector x represents a discretization of a real-valued function. Using the proposed regularization promotes solutions whose components vary continuously.

1. In this question, we assume that $d \gg 1$ and that the function f is separable, in that it can be written as

$$f(x) = \sum_{j=1}^d f^j([x]_j),$$

where every $f^j : \mathbb{R} \to \mathbb{R}$ only depends on the jth coordinate of the output vector x.

a) Justify that the second term in the objective is partially separable (which implies that its gradient also is).

b) Suppose that we apply a basic coordinate descent method to problem (9). Justify how an iteration of such a method can be considered cheaper than an iteration of gradient descent.

c) Suggest a block variant of that method that makes use of the structure of the second term in the objective.

2. In this question, we suppose that f is strongly convex. We modify problem (9) by introducing an auxiliary variable $z \in \mathbb{R}^d$, leading to

$$\min_{x\in\mathbb{R}^d, z\in\mathbb{R}^d} f(x) + \frac{\lambda}{2}\|z\|_2^2 \qquad \text{s.t.} \qquad Ax - z = 0. \tag{10}$$

a) Write down the Lagrangian for problem (10). Using this function, how can we rewrite problem (10)?

b) What is the difference between a Lagrangian and an augmented Lagrangian?

c) How does the introduction of the variable z allows for applying ADMM to this problem? What is the advantage of such an approach here?

3. Finally, we suppose that the objective function f can be expressed as a finite sum $f(x) = \sum_{i=1}^n f_i(x)$, where every f_i is strongly convex and continuously differentiable. We consider that all f_i are spread across different agents, but that all agents know the regularization term.

a) Consider first the formulation (9). Rewrite this problem under the assumption that every agent has its own copy of the problem variable, and is using its own function f_i instead of f.

b) Using the same idea as for obtaining (10), explain how the problem from the previous question can be reformulated as a linearly-constrained problem in order to apply ADMM.

Ex. 7 — *(non-convex optimization)* Let us imagine that we want to solve the following system of polynomial equations:

$$(P)\left\{\begin{array}{rcl} x_1^2 + x_2^2 & = & 2; \\ x_1^2 - x_2^2 & = & 0; \\ 2x_1x_2 & = & 2. \end{array}\right.$$

We define

$$\begin{array}{rccc} L & : & \mathbb{R}^2 & \to & \mathbb{R} \\ & & (x_1, x_2) & \to & \left(x_1^2 + x_2^2 - 2\right)^2 + \left(x_1^2 - x_2^2\right)^2 + \left(2x_1x_2 - 2\right)^2. \end{array}$$

1. Show that, if (P) has at least one solution, then a pair $(x_1, x_2) \in \mathbb{R}^2$ is a solution of (P) if and only if it is a global minimizer of L.

Therefore, to solve (P), it is enough to find

$$(x_1, x_2) \in \underset{(x_1,x_2)\in\mathbb{R}^2}{\operatorname{argmin}} L(x_1, x_2).$$

[It is by far not the simplest way to solve (P), but let us pretend it is the only one we can think of.]

2. Denoting $e := (1, 1)$, Show that, for any $x = (x_1, x_2) \in \mathbb{R}^2$,

$$\nabla L(x_1, x_2) = 8(||x||^2 x - \langle x, e\rangle\, e).$$

3. Recall the definition of first-order and second-order critical points of L.

4. Compute the first and second-order critical points. You can use with no proof the following formula for the Hessian: for any $x, h \in \mathbb{R}^2$,

$$\nabla^2 L(x_1, x_2) \cdot (h, h) = 8\left(2\langle x, h\rangle^2 - \langle e, h\rangle^2 + ||h||^2||x||^2\right).$$

5. a) Imagine that we run gradient descent on L, with a small stepsize, starting from a point x_{init} chosen at random, uniformly in the unit ball of $\mathbb{R}^2$. What will, in your opinion, be the limit behavior of the sequence of iterates?

b) (Difficult) Prove that your opinion is correct.

Answer (Ex. 1) —

1. One has

$$f(x+\epsilon) = L(Bx+B\epsilon) = f(x) + \langle \nabla L(Bx),\, B\epsilon \rangle + o(\epsilon) = f(x) + \left\langle B^\top \nabla L(Bx),\, \epsilon \right\rangle + o(\epsilon)$$

so that $\nabla f(x) = B^\top \nabla L(Bx)$.

2. One has $\ell'(s) = e^s/(1+e^s)$. One has $\nabla L(z) = (\ell'(z_1), \ldots, \ell'(z_n))$.
3. One can use $B = -\mathrm{rows}(y_i a_i)$. We can use the previous question to compute ∇f.
4. One has

$$g(t+\epsilon) = f((x - t\nabla f(x)) - \epsilon \nabla f(x)) = g(t) - \epsilon \langle x - t\nabla f(x),\, \nabla f(x) \rangle + o(\epsilon)$$

so that $g'(t) = -\langle x - t\nabla f(x),\, \nabla f(x) \rangle$. One has $g'(0) = -\|\nabla f(x)\|^2$ which is negative if x is not stationary. It tells us that with a small enough step size, the energy f can only decrease during GD.

Answer (Ex. 2) —

1. Forward-mode differentiation has a better time complexity because there are more outputs than inputs.
2. - $a = x^2$
   - $b = a + 1$
   - $c = \sqrt{b}$
   - $d = \log b$
   - $y = c \times d$
3. - $\frac{\partial y}{\partial y} = 1$
   - $\frac{\partial y}{\partial d} = \frac{\partial y}{\partial y}\frac{\partial y}{\partial d} = c$
   - $\frac{\partial y}{\partial c} = \frac{\partial y}{\partial y}\frac{\partial y}{\partial c} = d$
   - $\frac{\partial y}{\partial b} = \frac{\partial y}{\partial c}\frac{\partial c}{\partial b} + \frac{\partial y}{\partial d}\frac{\partial d}{\partial b} = d \times \frac{1}{2\sqrt{b}} + c \times \frac{1}{b}$
   - $\frac{\partial y}{\partial a} = \frac{\partial y}{\partial b}\frac{\partial b}{\partial a} = \frac{\partial y}{\partial b} 2x$
4. Let us define $f(x,a) = \frac{1}{2}ax^2 + bx + c$. The derivative of f w.r.t. a is $\frac{1}{2}x^2$. From Danskin's thereom, we therefore have $h'(a) = \frac{1}{2}(x^\star(a))^2$. The derivative of f w.r.t. x is $ax + b$. Setting it to zero, we get $x^\star(a) = -\frac{b}{a}$. Plugging back into $f(x,a)$, we get $h(a) = -\frac{1}{2}\frac{b^2}{a} + c$. We thus obtain $h'(a) = \frac{1}{2}\frac{b^2}{a^2}$, which indeed matches our previous derivation.

Answer (Ex. 3) —

1. The kth iteration of a batch stochastic gradient applied to problem (1) can be written as

$$x_{k+1} = x_k - \frac{\alpha_k}{|\mathcal{S}_k|} \sum_{i \in \mathcal{S}_k} \nabla f_i(x_k),$$

where $\alpha_k > 0$ and $\mathcal{S}_k$ is a set of indices drawn randomly in $\{1, \ldots, n\}$. In order for that iteration to correspond to gradient descent, $\mathcal{S}_k$ must consist of n indices drawn without replacement from $\{1, \ldots, n\}$.

2. An epoch is a unit of cost corresponding to n accesses of an example in the data set. Since the kth iteration requires n_k such accesses, its cost is a fraction $\frac{n_k}{n}$ of an epoch.
3. Using a batch size is one possible way of reducing the variance in the stochastic gradient estimate. Indeed, if the variance associated with a single stochastic gradient (i.e. a single random index i) is σ^2, then using a set $\mathcal{S}$ of n_b indices drawn independently with the same distribution than i will lead to a variance in $\frac{\sigma^2}{n_b} < \sigma^2$ for $n_b > 1$:

$$\mathbb{E}_i\left[\|\nabla f_i(x)\|^2\right] \le \sigma^2 + \|\nabla f(x)\|^2 \quad \Leftarrow \quad \mathbb{E}_{\mathcal{S}}\left[\left\|\frac{1}{n_b}\sum_{i\in\mathcal{S}} \nabla f_i(x)\right\|^2\right] \le \frac{\sigma^2}{n_b} + \|\nabla f(x)\|^2.$$

More informally, whenever we increase the batch size, we incorporate more information on average, leading to stochastic gradient estimates that are less noisy than that of a vanilla gradient descent approach.

4. *Several possible answers* Using gradient aggregation methods, that combine current stochastic gradient evaluations with values at previous iterates, gives rise to several variance-reduction methods. Averaging the iterates computed by stochastic gradient is another way of reducing noise, in that it produces an alternate sequence with a typically less noisy behavior.

5. a) As long as the desired condition is not satisfied, we must have $\mathbb{E}\left[f(x_K) - f^*\right] > \epsilon$. Combining this with (2) gives

$$\epsilon < \mathbb{E}\left[f(x_K) - f^*\right] \leq c\tau^{-K} \quad \Leftarrow \quad K < \log_\tau\left(\tfrac{c}{\epsilon}\right) = \ell.$$

Therefore, the method will satisfy the desired condition after at most $\lceil \log_\tau\left(\frac{c}{\epsilon}\right)\rceil$ iterations.

b) Recall that the cost of the kth iteration of the dynamic batch stochastic gradient method is $\frac{n_k}{n}$ epochs. Since the method will take at most $\ell = \lceil \log_\tau\left(\frac{c}{\epsilon}\right)\rceil$ iterations to reach ϵ accuracy in expectation, the cost is at most

$$\begin{aligned}\sum_{k=0}^{\ell} \frac{n_k}{n} &= \frac{1}{n}\sum_{k=0}^{\ell} \lceil \tau^k \rceil \\ &\leq \frac{2}{n}\sum_{k=0}^{\ell} \tau^k \\ &\leq \frac{2}{n}\frac{\tau^{\ell+1}-1}{\tau-1} \\ &\leq \frac{2\tau}{\tau-1}\frac{\tau^\ell}{n}.\end{aligned}$$

Using that $\tau^\ell = \frac{c}{\epsilon}$, we obtain

$$\sum_{k=0}^{\ell} \frac{n_k}{n} \leq \frac{2\tau c}{\tau-1}\frac{1}{n\epsilon},$$

proving the desired result.

c) Gradient descent reaches ϵ accuracy in at most $\mathcal{O}(\log(1/\epsilon))$ iterations, but every iteration has a cost of 1 epoch. As a result, this method reaches ϵ accuracy after at most $\mathcal{O}(\log(1/\epsilon))$ epochs. This latter bound can be much higher than the $\mathcal{O}\left(\frac{1}{n\epsilon}\right)$ bound for dynamic batch stochastic gradient method for large n.

d) If the τ parameter is set very large (e.g. $\tau = n$, or τ being a small fractional power of n), then the cost of a dynamic batch stochastic gradient iteration will quickly exceed that of a gradient descent iteration, leading to one iteration costing more than an epoch. In that case, the epoch complexity results above will either be comparable to or in favor of gradient descent.

6. Advanced methods like AdaGrad, RMSProp or Adam use a scaling of the stepsizes along each coordinate of the stochastic gradient estimates. Any of those methods is therefore a good suggestion for a try, as they will produce stepsizes that adaptively adjust to the magnitude along every coordinate. On a finer level, if the stochastic gradients are not sparse, using the averaging technique of RMSProp or Adam is likely a better choice than using AdaGrad.

Answer (Ex. 4) — 1. a) The function f is continuous, and the set A is nonempty compact (closed and bounded in $\mathbb{R}^p$). Therefore f has a minimizer on A.

b) *(There are many different possible answers)* Let $g\colon x \mapsto \left(\frac{1}{2}\langle Mx,\, x\rangle\right)$. We know that g is C^∞, its gradient is $\nabla g(x) = Mx$, and its Hessian is $H_g(x) = M$. Since $H_g(x) = M \succeq 0$ by assumption, we deduce that g is convex.

In general $f = -g$ is not convex (it is concave). The only case where f is convex is when both $M \succeq 0$ and $-M \succeq 0$. That is, $M = 0$ and f is constant.

2. As $f \in C^1(\mathbb{R}^p)$, the optimality conditions are $-\nabla f(x^\star) \in \mathcal{N}_A(x)$.

We note that A satisfies Slater's conditions: letting $h(x) = \|x\|^2 - 1$ denote the inequality constraint, h is convex C^1, and there exists $x_0 = 0$ such that $h(x_0) < 0$. As a result the constraints are qualified:

$$\mathcal{N}_A(x) = \{\mu\nabla h(x)\ ;\ \mu \geq 0\} = \{2\mu x\ ;\ \mu \geq 0\} \quad \text{if } \|x\| = 1,$$

or otherwise

$$\mathcal{N}_A(x) = \{0\} \text{ if } \|x\| < 1$$

(depending on whether the constraint $h(x) \leq 0$ is active or not).
Since the constraints are qualified, we can reformulate the optimality conditions using the KKT conditions. Let $\mathcal{L}(x, \mu) = f(x) + \mu h(x)$. Then the minimizer $x^\star$ should satisfy:

$$\left\{\begin{array}{rl} \nabla_x \mathcal{L}(x^\star, \mu) & = 0, \\ h(x^\star) & \leq 0, \\ \mu & \geq 0, \\ \mu h(x) & = 0. \end{array}\right., \text{ that is } \left\{\begin{array}{rl} Mx^\star & = 2\mu x^\star, \\ \|x^\star\| & \leq 1, \\ \mu & \geq 0, \\ \mu\left(\|x^\star\|^2 - 1\right) & = 0. \end{array}\right.$$

3. The projected gradient descent algorithm is given by

$$\forall t \geq 1, \quad x_{t+1} = P_A\left(x_t - \tau \nabla f(x)\right).$$

In our case, it amounts to

$$\forall t \geq 1, \quad x_{t+1} = \frac{(I + \tau M)x_t}{\max\left(1, \|(I + \tau M)x_t\|\right)}.$$

This is similar to the power iteration method applied to $B = I + \tau M$,

$$x_{t+1} = \frac{Bx_t}{\|Bx_t\|},$$

which is used to compute a leading eigenvector of a matrix B. In fact, we show below that $\max\left(1, \|(I + \tau M)x_t\|\right) = \|(I + \tau M)x_t\|$, so that there is really an equivalence.

4. a) If $x = 0$ is a minimizer, it means that for all $x \in A$,

$$0 = -\frac{1}{2}\langle M0,\, 0\rangle \leq -\frac{1}{2}\langle Mx,\, x\rangle \leq 0$$

so that every $x \in A$ is a minimizer. Note that f is constant in a neighborhood of 0, so that $M = H_f(x) = 0$.
Conversely, if $M = 0$, then every $x \in A$ is a minimizer.

b) If 0 is not a minimizer, then for every minimizer $x^\star \in A$,

$$-\frac{1}{2}\langle Mx^\star,\, x^\star\rangle < -\frac{1}{2}\langle M0,\, 0\rangle = 0.$$

Assume by contradiction that $t \overset{\text{def}}{=} \|x^\star\| < 1$, then $x/t \in A$ and

$$-\frac{1}{2}\left\langle M\left(\frac{1}{t}x^\star\right),\, \frac{1}{t}x^\star\right\rangle = \left(\frac{1}{t^2}\right)\left(-\frac{1}{2}\langle Mx^\star,\, x^\star\rangle\right) < -\frac{1}{2}\langle Mx^\star,\, x^\star\rangle,$$

which contradicts the optimality of $x^\star$. Hence $\|x^\star\| = 1$.
Remark: *the solution is not unique either. If x is a minimizer, then so is $-x$.*

c) The KKT conditions imply that $Mx^\star = 2\mu x^\star$. Since $x^\star \neq 0$, that means that $x^\star$ is an eigenvector of M, with eigenvalue 2μ.

5. a) Let $x = \pm v_i$ for $1 \leq i \leq p$. We see that

$$\left\{\begin{array}{rl} Mx & = 2\left(\frac{\lambda_i}{2}\right)x, \\ \|x\| & \leq 1, \\ \lambda_i/2 & \geq 0, \\ \frac{\lambda_i}{2}\left(\|x\|^2 - 1\right) & = 0. \end{array}\right.$$

Therefore all the $\pm v_i$'s satisfy the KKT conditions. Yet, they have value

$$f(\pm v_i) = -\frac{1}{2}\langle Mv_i,\, v_i\rangle = -\frac{\lambda_i}{2}, \tag{11}$$

so that for $1 \leq i \leq p-1$, $f(\pm v_p) < f(\pm v_i)$ and the point $\pm v_i$ is not a minimizer. Since we know that there is a minimizer $x^\star$ with unit norm which is an eigenvector, we must have $x^\star = \pm v_p$. As a result, $\operatorname{argmin}_A f = \{v_p, -v_p\}$.

b) Let $x_0 = \pm v_i$. We have $(I+\tau M)x_0 = (1+\tau\lambda_i)x_0$ with $1+\tau\lambda_i > 1$, so that

$$x_1 = \frac{(I+\tau M)x_0}{\max\left(1, \|(I+\tau M)x_0\|\right)} = x_0.$$

By induction, we see that $x_t = x_0$ for all $t \geq 1$. Thus, for $1 \leq i \leq p-1$,

$$\lim_{t\to\infty} f(x_t) = -\lambda_i > \min_A f.$$

c) Let $t \in \mathbb{N}$, and assume that $\|x_t\| \geq 1$. Since M is positive semi-definite and $\|x_t\| \geq 1$, we have

$$\|(I+\tau M)x_t\|^2 = \|x_t\|^2 + \underbrace{\tau^2 \|Mx_t\|^2}_{\geq 0} + \underbrace{2\tau \langle Mx_t,\, x_t\rangle}_{\geq 0} \geq 1.$$

As a result, $\max\left(1, \|(I+\tau M)x_t\|\right) = \|(I+\tau M)x_t\|$, and

$$x_{t+1} = P_A((I+\tau M)x_t) = \frac{(I+\tau M)x_t}{\|(I+\tau M)x_t\|}.$$

By induction, we deduce that for all $t \in \mathbb{N}\setminus\{0\}$, $\|x_t\| = 1$ and

$$x_t = \frac{(I+\tau M)^t x_0}{\|(I+\tau M)^t x_0\|}.$$

As a result, for all $t \geq 1$, $x_t = \sum_{i=1}^p \alpha_{t,i} v_i$ with

$$\alpha_{t,i} = \frac{(1+\tau\lambda_i)^t \alpha_{0,i}}{\left\|\sum_{j=1}^p (1+\tau\lambda_j)^t \alpha_{0,j} v_j\right\|}.$$

Since $(v_i)_{1\leq i\leq p}$ is an orthonormal basis, we have

$$\left\|\sum_{j=1}^p (1+\tau\lambda_j)^t \alpha_{0,j} v_j\right\| = \sqrt{\sum_{j=1}^p (1+\tau\lambda_j)^{2t} |\alpha_{0,j}|^2} \geq (1+\tau\lambda_p)^t |\alpha_{0,p}|,$$

hence, for $1 \leq i \leq p-1$,

$$|\alpha_{t,i}| \leq \left(\frac{1+\tau\lambda_i}{1+\tau\lambda_p}\right)^t \frac{|\alpha_{0,i}|}{|\alpha_{0,p}|} \longrightarrow 0$$

as $t \to +\infty$. On the other hand,

$$\begin{aligned}\left\|\sum_{j=1}^p (1+\tau\lambda_j)\alpha_{t,j} v_j\right\| = \sqrt{\sum_{j=1}^p (1+\tau\lambda_j)^2 |\alpha_{t,j}|^2} &= (1+\tau\lambda_p)^t |\alpha_{0,p}| \sqrt{1 + \sum_{i=1}^{p-1} \left(\frac{1+\tau\lambda_j}{1+\tau\lambda_p}\right)^{2t} \frac{|\alpha_{0,j}|^2}{|\alpha_{0,p}|^2}} \\ &= (1+\tau\lambda_p)^t |\alpha_{p,0}| (1+o(1)),\end{aligned}$$

so that, for $t \to +\infty$,

$$\alpha_{t,p} = 1 + o(1).$$

d) Let $x_0 \in \mathbb{R}^p$ such that $\alpha_{0,p} \neq 0$. If $\|x_0\| \geq 1$, we may apply the above result: for all $t \geq 1$, $x_t \in A$ and

$$\min_A f \leq f(x_t) = -\frac{1}{2}\langle Mx_t,\, x_t\rangle \leq -\frac{1}{2}\lambda_p (\alpha_{t,p})^2 \longrightarrow -\frac{1}{2}\lambda_p = \min_A f.$$

so that the algorithm yields a minimizing sequence.
If $\|x_0\| < 1$, for a few iterations, the algorithms yields iterates of the form

$$x_t = (I+\tau M)^t x_0$$

hence $\|x_t\| \geq (1+\tau\lambda_1)^t \|x_0\|$. Therefore, there is some finite $T \in \mathbb{N}$ such that

$$\|(I+\tau M)x_T\| \geq 1.$$

At that point, we have $\alpha_{T+1,p} \neq 0$, $\|x_{T+1}\| \geq 1$, and the algorithm switches to the regime studied above: $\lim_{t\to\infty} f(x_t) = \min_A f$.
As a result, except possibly on the hyperplane defined by $\alpha_{0,p} = 0$, $\lim_{t\to\infty} f(x_t) = \min_A f$.
Remark: *Let us partition the space* $\mathbb{R}^p$ *as*

$$\mathbb{R}^p = \bigcup_{i=0}^{p} E_i, \quad \textit{where } E_i = \{x \in \mathbb{R}^p : x_p = \ldots = x_{i+1} = 0, x_i \neq 0\}, E_0 = \{0\}.$$

Applying the above result with a starting point $x_0 \in E_i$*, we see that for* $1 \leq i \leq p$*,*

$$\forall x_0 \in E_i, \quad \lim_{t\to\infty} f(x_t) = -\frac{1}{2}\lambda_i,$$

while for $x_0 = 0$*, we have* $f(x_t) = 0$ *for all* t*.*

Answer (Ex. 5) — 1. The function $x \mapsto \|x\|_1$ is nonsmooth in the sense that its gradient is not defined at any point that has a zero coefficient. This property precludes from applying gradient-type techniques to optimize this function and, consequently, to the objective function of problem (7).

2. Multiplying the objective function by a positive constant does not modify the set of solutions of an optimization problem. Since $\alpha > 0$, this justifies that

$$\mathrm{argmin}_{x\in\mathbb{R}^d} \left\{\alpha\|x\|_1 + \frac{1}{2}\|x-w\|_2^2\right\} = \mathrm{argmin}_{x\in\mathbb{R}^d} \left\{\|x\|_1 + \frac{1}{2\alpha}\|x-w\|_2^2\right\}.$$

Both functions are continuous and strongly convex, therefore this set contains a single element.

3. An optimality condition for problem (7) can be formulated using the subdifferential of the objective. More precisely, the unique solution of the problem, call it x^*, is the unique vector such that

$$0_{\mathbb{R}^d} \in \partial\left(\|\cdot\|_1 + \frac{1}{2\alpha}\|\cdot - w\|_2^2\right)(x^*).$$

Extra: Since only the ℓ_1 *norm is not necessarily differentiable at* x^* *while* $\nabla\left(\frac{1}{2\alpha}\|\cdot - w\|_2^2\right)(x^*) = \frac{1}{\alpha}(x^* - w)$*, this condition is equivalent to*

$$\frac{1}{\alpha}(w - x^*) \in \partial\left(\|\cdot\|_1\right)(x^*).$$

4. a) By definition, the proximal operator of a function h is

$$\mathrm{prox}_h(x) = \mathrm{argmin}_{u\in\mathbb{R}^d}\left\{h(u) + \frac{1}{2}\|u-x\|_2^2\right\}.$$

The objective of problem (8) has the structure appearing in the definition of the proximal operator. As a result, we can write the solution (which is unique by Question 2) as

$$x^* = \mathrm{prox}_{\alpha\|\cdot\|_1}(w).$$

b) Repeating the iteration

$$w \leftarrow \mathrm{prox}_{\alpha\|\cdot\|_1}(w)$$

corresponds to performing iterations of the proximal point algorithm.

5. In this question, we view problem (7) in a composite form, where $\frac{1}{2\alpha}\|x-w\|_2^2$ is a data-fitting term and $|x\|_1$ is a regularization term.

a) The purpose of an ℓ_1 regularization term is to promote sparsity of the solution. Its computational interest is that it is a convex, continuous relaxation term (unlike the ℓ_0 norm which is both nonconvex and discontinuous) for which the proximal subproblems can be solved explicitly.

b) The proximal gradient iteration for problem 7 can be written as

$$x_{k+1} = \mathrm{argmin}_{x \in \mathbb{R}^d} \left\{ \frac{1}{2}\|x_k - w\|_2^2 + \nabla \| \cdot \|_2^2(x_k)^{\mathrm{T}}(x - x_k) + \frac{1}{2\alpha}\|x - x_k\|_2^2 + \|x\|_1 \right\}.$$

Since $x_k = w$, we have

$$\nabla \| \cdot \|_2^2(x_k) = \tfrac{1}{\alpha}(x_k - w) = 0_{\mathbb{R}^d},$$

and thus the first two terms vanish, resulting in the expression

$$x_{k+1} = \mathrm{argmin}_{x \in \mathbb{R}^d} \left\{ \frac{1}{2\alpha}\|x - w\|_2^2 + \|x\|_1 \right\}.$$

This iteration of proximal gradient is therefore equivalent to solving the original problem. This could have been expected because the original problem is a quadratic problem with an ℓ_1 regularization term, and thus the proximal subproblems have the same form.

Answer (Ex. 6) — 1. a) The second term of the objective can be written as $\ell(x) = \sum_{j=1}^d \ell^j(x)$, where

$$\ell^j(x) = \frac{\lambda}{2} \times \begin{cases} (-2[x]_1 + [x]_2)^2 & \text{if } j = 1 \\ ([x]_{i-1} - 2[x]_i + [x]_{i+1})^2 & \text{if } 1 < j < d \\ ([x]_{d-1} - 2[x]_d)^2 & \text{if } j = n. \end{cases}$$

As a result, every function ℓ_j depends on two or three coordinates of the output vector. The sum of ℓ_js is thus partially separable.

b) An iteration of standard coordinate descent for problem (9) can be written as

$$[x_{k+1}]_{j_k} = [x_k]_{j_k} - \alpha_k \left(\nabla_{j_k} f(x_k) e_{j_k} + \nabla_{j_k} \ell(x_k) e_{j_k} \right),$$

where $\alpha_k > 0$ and $j_k \in \{1, \dots, d\}$. Because the function f is assumed to be separable, we have that

$$\nabla_{j_k} f(x_k) e_{j_k} = \nabla f^{j_k}([x_k]_{j_k}) e_{j_k},$$

hence this part of the calculation only requires to access one coordinate of the iterate. Thanks to the previous question, we also know that

$$\nabla_{j_k} \ell(x_k) = \sum_{j=1}^{d} \nabla_{j_k} \ell^j(x_k) = \sum_{\max\{1, j_k - 1\}}^{\min\{n, j_k + 1\}} \nabla_{j_k} \ell^j(x_k),$$

with at most five coordinates being involved in the latter calculation. As a result, every iteration of coordinate descent requires at most access to five coordinates of the iterate. When such access represents the computational bottleneck of the algorithm, coordinate descent iterations can be performed at a much lower cost than gradient descent iterations.

c) *The answer of this question is relatively open, but the answer should lead to cheaper updates than gradient descent.* Given that every coordinate appears in at most three terms in the second part of the objective in a very controlled pattern, one possiblity consists in using subsets of three consecutive coordinates as blocks (drawn either at random or in a cyclic order). A refined version of this consists in distinguishing the two "corner cases" where only indices $(1, 2)$ and $(d - 1, d)$ are to be used. Alternatively, one can draw an index $j_k \in \{1, \dots, d\}$, then perform updates according to all coordinates with which the j_kth coordinate is linked through the regularization term. The corresponding block size will range between 3 and 5 and require at most 9 coordinate accesses, so it will remain cheaper than gradient descent.

2. a) The Lagrangian for problem (10) is given by

$$\mathcal{L}(x, z, y) = f(x) + \frac{\lambda}{2}\|z\|_2^2 + y^{\mathrm{T}}(Ax - z),$$

where $y \in \mathbb{R}^d$ is the dual variable associated with the constraint $Lx - z = 0$. Using this function, problem (10) can be rewritten as

$$\mathrm{minimize}_{\substack{x \in \mathbb{R}^d \\ z \in \mathbb{R}^d}} \ \mathrm{maximize}_{y \in \mathbb{R}^d} \mathcal{L}(x, z, y).$$

N.B. The feasible set of problem (10) *is not empty here since it contains the pair* ($x = 0_{\mathbb{R}^d}, z = 0_{\mathbb{R}^d}$)*. This justification is not expected, but could be worthy of a bonus.*

b) The augmented Lagrangian is a regularized version of the Lagrangian in which a penalty term is added in order to penalize points that do not satisfy the constraints of the original problem. Mathematically, an augmented Lagrangian for our problem of interest would be

$$\mathcal{L}(x,z,y;\mu) = f(x) + \frac{\lambda}{2}\|z\|_2^2 + y^{\mathrm{T}}(Ax-z) + \frac{\mu}{2}\|Ax-z\|_2^2,$$

for some $\mu > 0$.

c) The objective function of problem (10) is partially separable in x and z, in that it has the form $f(x)+g(z)$. Problem (10) consists in minimizing this sum under linear constraints that are also separable in x and z. This is the typical setup in which ADMM is applicable, and it results in iterations where an augmented Lagrangian will be minimized first with respect to either x or z, then with respect to the other variable. Assuming the method starts by updating x, at the kth iteration, the update with respect to z will be

$$z_{k+1} = \mathrm{argmin}_{z\in\mathbb{R}^d}\,\mathcal{L}(x_{k+1},z,y_k;\mu),$$

where the augmented Lagrangian is a quadratic function of z with simple quadratic terms. It is thus quite easy to minimize exactly, which shows the advantage of ADMM in this setting.

3. a) Assuming that every agent has a copy of the problem variable, we obtain the following problem

$$\mathrm{minimize}_{\substack{x^1\in\mathbb{R}^d\\ \vdots\\ x^n\in\mathbb{R}^d}}\ f_i(x^i) + \frac{\lambda}{2}\|Lx^i\|_2^2.$$

b) By adding an extra variable to account for the second term in each objective, we obtain the following formulation:

$$\begin{aligned}\mathrm{minimize}_{\substack{x^1\in\mathbb{R}^d\\ \vdots\\ x^n\in\mathbb{R}^d}} \quad & f_i(x^i) + \tfrac{\lambda}{2}\|z\|_2^2\\ \text{s.t.} \quad & Ax^i - z = 0.\end{aligned}$$

We therefore obtain a partially separable problem in which the functions have independent terms in $x^1,\dots,x^n,z$. We can thus apply ADMM to this setting to perform updates with respect to all x^i by their respective agents, then perform updates with respect to z and the dual variables by a single agent.

Answer (Ex. 7) — 1. Let us assume that (P) has at least one solution, which we denote (x_1^*,x_2^*). Then $\min_{x\in\mathbb{R}^2} L(x) = 0$. Indeed, $L(x)\geq 0$ for any $x\in\mathbb{R}^2$ (a sum of square is always nonnegative), and $L(x_1^*,x_2^*) = 0^2+0^2+0^2 = 0$.

Consequently, a pair (x_1,x_2) is a global minimizer of L if and only if

$$L(x_1,x_2) = 0,$$

which happens if and only if each square in the definition of L is zero, that is

$$\begin{aligned} x_1^2+x_2^2-2 &= 0;\\ x_1^2-x_2^2 &= 0;\\ 2x_1x_2-2 &= 0,\end{aligned}$$

which is equivalent to (x_1,x_2) being a solution of (P).

2. For any $x=(x_1,x_2)\in\mathbb{R}^2$,

$$\begin{aligned}\frac{\partial L}{\partial x_1}(x) &= 2(2x_1)(x_1^2+x_2^2-2) + 2(2x_1)(x_1^2-x_2^2) + 2(2x_2)(2x_1x_2-2)\\ &= 8x_1(x_1^2+x_2^2) - 8(x_1+x_2);\\ \frac{\partial L}{\partial x_2}(x) &= 2(2x_2)(x_1^2+x_2^2-2) + 2(-2x_2)(x_1^2-x_2^2) + 2(2x_1)(2x_1x_2-2)\\ &= 8x_2(x_1^2+x_2^2) - 8(x_1+x_2).\end{aligned}$$

Hence

$$\nabla L(x) = \begin{pmatrix} \frac{\partial L}{\partial x_1}(x) \\ \frac{\partial L}{\partial x_2}(x) \end{pmatrix} = 8(x_1^2 + x_2^2) \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} - 8(x_1 + x_2) \begin{pmatrix} 1 \\ 1 \end{pmatrix} = 8(||x||^2 x - \langle x, e \rangle e).$$

3. An element $x \in \mathbb{R}^2$ is a first-order critical point of L if and only if $\nabla L(x) = 0$.
It is a second-order critical point if and only if $\nabla f(x) = 0$ and $\nabla^2 f(x) \succeq 0$.

4. We start with the first-order critical points. Let $x = (x_1, x_2)$ be an element of $\mathbb{R}^2$. It is a first-order critical point if and only if

$$||x||^2 x - \langle x, e \rangle e = 0.$$

This equality is true if $x = 0$. If $x \neq 0$, it is equivalent to x being colinear to e ($x = \lambda e$ for some $\lambda \in \mathbb{R}$), with the colinearity factor λ satisfying

$$\begin{aligned} & ||\lambda e||^2 \lambda e - \langle \lambda e, e \rangle e = 0; \\ \iff \quad & (2\lambda^3 - 2\lambda) e = 0; \\ \iff \quad & \lambda = -1, 0 \text{ or } 1. \end{aligned}$$

In other words, x is a first-order critical point of L if and only if $x = 0, x = -e$ or $x = e$.
Let us now compute the second-order critical points. A second-order critical point is a first-order critical point at which the Hessian of L is positive semidefinite. Let us thus compute the Hessian at $0, -e, e$.
For any h,

$$\nabla^2 L(0) \cdot (h, h) = -8 \langle e, h \rangle^2 .$$

In particular, $\nabla^2 L(0) \cdot (e, e) = -8||e||^4 < 0$, hence $\nabla^2 L(0) \not\succeq 0$ and 0 is not a second-order critical point.
For any h,

$$\nabla^2 L(-e) \cdot (h, h) = \nabla^2 L(e) \cdot (h, h) = 8 \left(\langle e, h \rangle^2 + ||e||^2 ||h||^2 \right) \geq 0.$$

Therefore, $\nabla^2 L(\pm e) \succeq 0$: the second-order critical points of L are e and $-e$.

5. a) With probability 1, the sequence of iterates converges to e or $-e$, that is towards a solution of (P) (one can check that $L(-e) = L(e) = 0$).

b) To prove the above statement, we use a theorem seen during the lecture on non-convex optimization (Theorem 2 in the lecture notes). This theorem applies to functions which are ℓ-smooth for some $\ell > 0$, have bounded sublevel sets and a finite number of first-order critical points. Our function L has a finite number of first-order critical points. One can also show that it has bounded sublevel sets. However, it is not ℓ-smooth. To be fully rigorous, we will therefore show that the gradient descent iterates stay inside a bounded domain, then construct a function $\tilde{L}$ which coincides with L over this bounded domain, and is ℓ-smooth for some $\ell > 0$. We will apply the theorem to $\tilde{L}$, and it will be enough to prove that the gradient descent iterates of L converge, since, in the bounded domain, L and $\tilde{L}$ yield the same sequence of iterates.
Let us assume that we run gradient descent with stepsize α smaller than $\frac{1}{96}$. We denote $(x_n)_{n \in \mathbb{N}}$ the sequence of iterates generated from an initial point $x_0 = x_{init}$ in the unit ball. First, we show by iteration over n that, if $||x_{init}|| \leq 1$, then $||x_n|| \leq 3$ for any $n \geq 0$.

- For $n = 0$, it is true: $||x_0|| = ||x_{init}|| \leq 1 \leq 3$.
- Let us assume that $||x_n|| \leq 3$ for some $n \in \mathbb{N}$. If $||x_n|| \leq 2$, then

$$\begin{aligned} ||\nabla L(x_n)|| &\leq 8(||x_n||^3 + |\langle x_n, e \rangle| \, ||e||) \\ &\leq 8(||x_n||^3 + ||x_n|| \, ||e||^2) \\ &= 8(||x_n||^3 + 2||x_n||) \\ &\leq 8(2^3 + 2.2) \\ &= 96. \end{aligned}$$

In this case,

$$||x_{n+1}|| = ||x_n - \alpha \nabla L(x_n)|| \leq ||x_n|| + 96\alpha \leq 2 + 96\frac{1}{96} \leq 3.$$

Now, if $||x_n|| > 2$,

$$\begin{aligned}||\nabla L(x_n)||^2 &= 64(||x_n||^4 - 2(||x_n||^2 - 1)\left\langle x_n, e\right\rangle^2)\\ &\leq 64||x_n||^4\\ &\leq 64 \times 9||x_n||^2 \quad (\text{since } ||x_n|| \leq 3).\end{aligned}$$

$$\begin{aligned}\left\langle \nabla L(x_n), x_n\right\rangle &= 8(||x_n||^4 - \left\langle x_n, e\right\rangle^2)\\ &\geq 8||x_n||^2(||x_n||^2 - ||e||^2)\\ &= 8||x_n||^2(||x_n||^2 - 2)\\ &\geq 16||x_n||^2.\end{aligned}$$

Therefore, if $\alpha \leq \frac{1}{96} < \frac{1}{18}$,

$$\begin{aligned}||x_{n+1}||^2 &= ||x_n||^2 - 2\alpha\left\langle x_n, \nabla L(x_n)\right\rangle + \alpha^2||\nabla L(x_n)||^2\\ &\leq ||x_n||^2 - 32\alpha||x_n||^2 + 64 \times 9\alpha^2||x_n||^2\\ &= ||x_n||^2(1 - 32\alpha(1 - 18\alpha))\\ &\leq ||x_n||^2,\end{aligned}$$

so $||x_{n+1}|| \leq ||x_n|| \leq 3$.

We have shown that whatever the initial point x_{init} in the unit ball, the iterates of gradient descent stay in $\overline{B}(0,3)$ (the closed ball centered at 0, with radius 3).

We define $M = \sup_{x\in\overline{B}(0,3)} L(x)$. Let $\phi : \mathbb{R} \to \mathbb{R}$ be any C^∞ function such that

- $\phi(t) = t$ for all $t \leq M$;
- $\phi(t) = \sqrt{t}$ for all t large enough;
- $\phi'(t) > 0$ for all $t \in \mathbb{R}$.

We set $\tilde{L} = \phi \circ L$.

This function is ℓ-smooth for some $\ell > 0$. Indeed, $\tilde{L}(x) = \sqrt{L(x)}$ for all x such that $||x||$ is large enough. Hence, if $||x||$ is large,

$$\nabla\tilde{L}(x) = \frac{\nabla L(x)}{2\sqrt{L(x)}};$$

$$\nabla^2\tilde{L}(x)\cdot(h,h) = \frac{\nabla^2 L(x)\cdot(h,h)}{2\sqrt{L(x)}} - \frac{\left\langle\nabla L(x), h\right\rangle^2}{4L(x)^{3/2}}.$$

As $L(x) \geq \frac{||x||^4}{2}$ if $||x||$ is large enough, $||\nabla L(x)|| = O(||x||^3)$ and $|||\nabla^2 L(x)||| = O(||x||^2)$, we have

$$|||\nabla^2\tilde{L}(x)||| = O(1) \quad \text{when } ||x|| \to +\infty.$$

As a consequence, $|||\nabla^2\tilde{L}|||$ is bounded over $\mathbb{R}^2$, hence $\nabla\tilde{L}$ is Lipschitz, that is $\tilde{L}$ is smooth.

In addition, $\tilde{L}$ has the same first-order critical points as L, hence a finite number, and it has bounded sublevel sets because $\tilde{L}(x) \to +\infty$ when $||x|| \to +\infty$.

We can therefore apply the theorem: for almost any initial point x_{init}, the sequence of iterates $(\tilde{x}_n)_{n\in\mathbb{N}}$ obtained by running gradient descent over $\tilde{L}$ from x_{init}, with a sufficiently small stepsize, converges to a second-order critical point of $\tilde{L}$, that is towards $\pm e$ (the second-order critical points of L and $\tilde{L}$ are the same). And since $\tilde{L} = L$ over $\overline{B}(0,3)$, the same result holds for gradient descent over L starting from an initial point in the unit ball.