Department of Mathematics, University of Utah

Introduction to Optimization
MATH 5770/6640, ME EN 6025 – Section 001 – Fall 2021
Homework 3 Solutions
Least squares and gradient descent

Due October 5, 2021

Text: *Introduction to Nonlinear Optimization*, Amir Beck,

Exercises:	#	3.1,
		3.2,
		4.3 (only the first 3 parts, ignore the *diagonally scaled* portions)
Extra:		P1,
		P2,
		P3,

3.1. Let $\boldsymbol{A} \in \mathbb{R}^{m\times n}$, $\boldsymbol{b} \in \mathbb{R}^n$, $\boldsymbol{L} \in \mathbb{R}^{p\times n}$, and $\lambda \in \mathbb{R}_{++}$. Consider the regularized least squares problem,

$$\text{(RLS)} \quad \min_{\boldsymbol{x}\in\mathbb{R}^n} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|^2 + \lambda\|\boldsymbol{L}\boldsymbol{x}\|^2.$$

Show that (RLS) has a unique solution if and only if $\text{Null}(\boldsymbol{A}) \cap \text{Null}(\boldsymbol{L}) = \{\boldsymbol{0}\}$, where here for a matrix $\boldsymbol{B}$, $\text{Null}(\boldsymbol{B})$ is the null space of $\boldsymbol{B}$ given by $\{\boldsymbol{x} : \boldsymbol{B}\boldsymbol{x} = \boldsymbol{0}\}$.

Solution: Since the objective function is quadratic, the minimization problem (RLS) has a unique solution if and only if the Hessian is positive definite. The Hessian of the objective is,

$$\frac{1}{2}\nabla^2 f = \boldsymbol{A}^T\boldsymbol{A} + \lambda\boldsymbol{L}^T\boldsymbol{L}, \qquad\qquad f(\boldsymbol{x}) = \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|^2 + \lambda\|\boldsymbol{L}\boldsymbol{x}\|^2.$$

Now let $\boldsymbol{x} \in \mathbb{R}^n\backslash\{\boldsymbol{0}\}$ be arbitrary. Defining $\boldsymbol{y} \coloneqq \boldsymbol{A}\boldsymbol{x}$ and $\boldsymbol{z} \coloneqq \boldsymbol{L}\boldsymbol{x}$, then we have,

$$\boldsymbol{x}^T\nabla^2 f\boldsymbol{x} = 2\boldsymbol{x}^T\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{x} + 2\lambda\boldsymbol{x}^T\boldsymbol{L}^T\boldsymbol{L}\boldsymbol{x} = 2\|\boldsymbol{y}\|^2 + 2\lambda\|\boldsymbol{z}\|^2.$$

We therefore conclude that $\boldsymbol{x}^T(\nabla^2 f)\boldsymbol{x} \geq 0$, and equals 0 if and only if $\|\boldsymbol{y}\| = \|\boldsymbol{z}\| = 0$ (since $\lambda > 0$). Since $\|\cdot\|$ is a norm, then $\|\boldsymbol{y}\| = \|\boldsymbol{z}\| = 0$ if and only if $\boldsymbol{y} = \boldsymbol{z} = \boldsymbol{0}$.
In summary, (RLS) has a unique solution if and only if either $\boldsymbol{y} \neq \boldsymbol{0}$ or $\boldsymbol{z} \neq \boldsymbol{0}$. This happens if and only if,

$$\boldsymbol{A}\boldsymbol{x} \neq \boldsymbol{0} \text{ or } \boldsymbol{L}\boldsymbol{x} \neq \boldsymbol{0}.$$

which means that we require $\boldsymbol{x} \neq \boldsymbol{0}$ to satisfy either $\boldsymbol{x} \notin \text{Null}(\boldsymbol{A})$ or $\boldsymbol{x} \notin \text{Null}(\boldsymbol{L})$. Therefore, (RLS) has a unique solution if and only if we cannot find any $\boldsymbol{x} \in \mathbb{R}^n\backslash\{\boldsymbol{0}\}$ that satisfies $\boldsymbol{x} \in \text{Null}(\boldsymbol{A}) \cap \text{Null}(\boldsymbol{L})$, i.e., if and only if $\text{Null}(\boldsymbol{A}) \cap \text{Null}(\boldsymbol{L}) = \{\boldsymbol{0}\}$.

3.2. Generate thirty points (x_i, y_i), $i = 1, 2, \ldots, 30$ by the Matlab code

```
randn('seed', 314);
x = linspace(0, 1, 30)';
y = 2*x.^2 - 3x+1+0.05randn(size(x));
```

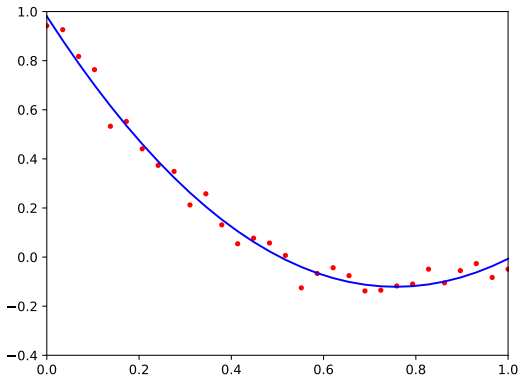

Figure 1: Plot associated to problem 3.2 from the text, showing a least squares quadratic polynomial fit (solid blue line) to noisy evaluations of a quadratic polynomial (red dots).

Find the quadratic function $y = ax^2 + bx + c$ that best fits the points in the least squares sense. Indicate what are the parameters a, b, c found by the least squares solution, and plot the points along with the derived quadratic function. The resulting plot should look like the one in Figure 3.5.

Solution: Python (not Matlab) code associated to this problem is available in the Git repo `https://github.com/akilnarayan/2021Fall-Optimization-homework3`, in particular the script `problem_3.2.py`. The textual output from running this script reads,

```
The least squares fitted polynomial has the form
1.9248 x^2 + -2.9126 x + 0.9808,
```

so that the coefficients are $a = 1.9248$, $b = -2.9126$, and $c = 0.9808$, which are close to the expected values of $(a, b, c) = (2, -3, 1)$. The deviation is caused by the noise added to the evaluations. The plot generated by this script is shown in Figure 1. Note that subsequent runs of this script will produce slightly different results due to the randomness of the noise.

4.3. Consider the quadratic minimization problem,

$$\min\{\boldsymbol{x}^T \boldsymbol{A}\boldsymbol{x} \ : \ \boldsymbol{x} \in \mathbb{R}^5\},$$

where $\boldsymbol{A}$ is the 5×5 Hilbert matrix defined by,

$$A_{i,j} = \frac{1}{i+j-1}, \qquad i, j = 1, 2, 3, 4, 5.$$

The matrix can be constructed via the Matla command `A = hilb(5)`. Run the following methods and compare the number of iterations required by each of the methods when the initial vector $\boldsymbol{x}_0 = (1, 2, 3, 4, 5)^T$ to obtain a solution $\boldsymbol{x}$ with $\|\nabla f(\boldsymbol{x})\| \leq 10^{-4}$:

- gradient method with backtracking stepsize rula nd parameters $\alpha = 0.5$, $\beta = 0.5$, $s = 1$;
- gradient method with backtracking stepsize rula nd parameters $\alpha = 0.1$, $\beta = 0.5$, $s = 1$;

- gradient method with exact line search;

Solution: Python (not Matlab) code associated to this problem is available in the Git repo `https://github.com/akilnarayan/2021Fall-Optimization-homework3`, in particular the script `problem_4.3.py`. The results are graphed in Figure 2. The matrix $\boldsymbol{A}$ is positive-definite, and so the exact solution to this problem is $\boldsymbol{x} = \boldsymbol{0}$ with function value 0. Note that all methods require a substantial number of iterations. The initial choice of (s, α, β) for backtracking $\alpha = 0.5$ requires more iterations than the alternative $\alpha = 0.1$, but not much improvement is gained. (Recall that decreasing α allows one to take more steps since this loosens the tolerance criterion regarding objective improvement relative to gradient value.) Using exact linesearch substantially improves the number of iterations, but still requires more than 1000 iterations. The lesson here is that simple first-order gradient-based methods can perform quite poorly, even in relatively small dimensions, $n = 5$. The reason for this behavior is that the matrix $\boldsymbol{A}$ is *poorly conditioned*.

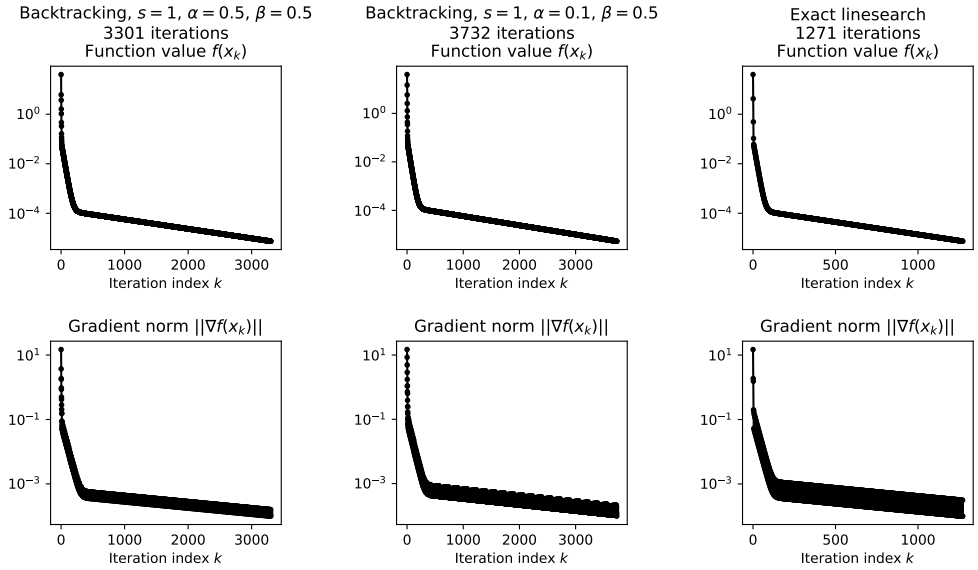

Figure 2: Plot associated to problem 4.3, showing the number of iterations required for each method, along with a log of the function value and gradient norm versus iteration count.

Additional problems:

P1. (Maximum likelihood estimation) Let $\{y_1, \ldots, y_M\} \subset \mathbb{R}$ denote M data points on the real line. The overall goal of this problem is to "fit" a probability distribution to these data points.

In particular, we assume that this data arose as (independent, identically distributed) samples from an unknown probability distribution with density $p(y)$. In order to find $p(y)$, we assume further that p corresponds to a normal distribution, i.e., a distribution having density

$$p(y; \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp(-(y-\mu)^2/(2\sigma^2)),$$

where μ and σ are the unknown mean and standard deviation of the distribution. We will choose the *parameters* (μ, σ) of this distribution as those parameters that maximize the "likelihood" of the data. In particular, given (μ, σ) and the data $\{y_m\}_{m=1}^M$, the likelihood is formally defined as

$$\mathcal{L}(\mu, \sigma) := \prod_{m=1}^{M} p(y_m; \mu, \sigma) = \prod_{m=1}^{M} \frac{1}{\sigma\sqrt{2\pi}} \exp(-(y_m - \mu)^2/(2\sigma^2)),$$

which is the probability of seeing independent data $\{y_m\}_{m=1}^M$ conditioned on their distribution having parameters (μ, σ). (It is not necessary for you to understand probability to complete this problem.)
The *maximum likelihood estimate* is the parameter choice that maximizes the likelihood:

$$(\mu_*, \sigma_*) = \operatorname*{argmax}_{\mu\in\mathbb{R}, \sigma\in\mathbb{R}_+ +} \mathcal{L}(\mu, \sigma).$$

Show that a strict global maximum of this optimization problem is given by

$$\mu_* = \frac{1}{M}\sum_{m=1}^{M} y_m, \qquad \sigma_*^2 = \frac{1}{M}\sum_{m=1}^{M} (y_m - \mu_*)^2.$$

(You may find it convenient to (i) use the logarithm function to monotonically transform the likelihood, (ii) convert the maximization problem into a minimization problem.)
6000-level students only: Simulate this result – with $M = 100$, choose some fixed value of μ, σ and generate data $\{y_m\}_{m=1}^{100}$ from a normal distribution with your prescribed (μ, σ). Compare a histogram of the data against the density $p(\cdot; \mu_*, \sigma_*)$ computed as the maximum likelihood estimate above.

Solution: We are attempting to maximize the function $\mathcal{L}$. Since log is a strictly monotone increasing function, then using a variant of the result in problem P3, we know that,

$$\operatorname*{argmax}_{\mu,\sigma} \mathcal{L}(\mu, \sigma) \overset{\text{P3}}{=} \operatorname*{argmax}_{\mu,\sigma} \log \mathcal{L}(\mu, \sigma) = \operatorname*{argmin}_{\mu,\sigma} -\mathcal{L}(\mu, \sigma),$$

where the second equality uses the fact that maxmimizing f is identical to minimizing $-f$. We will therefore compute

$$\operatorname*{argmin}_{\mu,\sigma} -\log \mathcal{L}(\mu, \sigma) = \operatorname*{argmin}_{\mu,\sigma} \left(\frac{M}{2}\log(2\pi) + M\log\sigma + \frac{1}{2\sigma^2}\sum_{m=1}^{M}(y_m - \mu)^2 \right).$$

Note that the domain we are minimizing over is $(\mu, \sigma) \in \mathbb{R} \times (0, \infty)$, since the likelihood is well-defined if and only if $\sigma > 0$. On this domain, the funciton $-\log \mathcal{L}$ is smooth (i.e., has continuous derivatives). We proceed to compute stationary points, which are given by solutions to

$$\nabla \left(-\log \mathcal{L}(\mu, \sigma)\right) = \mathbf{0}.$$

By direct computation,

$$\frac{\mathrm{d}}{\mathrm{d}\mu}(-\log \mathcal{L}) = \frac{1}{\sigma^2}\sum_{m=1}^{M}(\mu - y_m) = 0,$$

is solved by

$$\mu_* = \frac{1}{M}\sum_{m=1}^{M} y_m. \tag{1}$$

The second stationary point condition is,

$$\frac{\mathrm{d}}{\mathrm{d}\sigma}(-\log\mathcal{L}) = \frac{M}{\sigma} - \frac{1}{\sigma^3}\sum_{m=1}^{M}(y_m-\mu)^2 = 0,$$

where we must use $\mu = \mu_*$ to satisfy the first stationary point condition. Therefore, multiplying the above by σ^3 results in,

$$\sigma_*^2 = \frac{1}{M}\sum_{m=1}^{M}(y_m-\mu_*)^2. \tag{2}$$

We now assume that $\sigma_*^2 > 0$: if it equals 0, then this stationary point is outside the valid domain $\sigma \in (0,\infty)$, and this optimization problem has no solution. (This assumption is not strong since it implies only that the data $\{y_m\}_{m=1}^M$ is not constant.) To investigate second-order optimality of the stationary point (μ_*,σ_*), we compute the Hessian.

$$\nabla^2(-\log\mathcal{L}) = \frac{1}{\sigma^4}\begin{pmatrix} M\sigma^2 & 2\sigma S_1 \\ 2\sigma S_1 & -M\sigma^2 + 3S_2 \end{pmatrix},$$

where S_1 and S_2 are given by,

$$S_1 = \sum_{m=1}^{M}(y_m-\mu), \qquad S_2 = \sum_{m=1}^{M}(y_m-\mu)^2.$$

Note that, if $\mu = \mu_*$, then $S_1 = 0$, and $S_2 = M\sigma_*^2$ (see (1) and (2)). Therefore, at the stationary point (μ_*,σ_*), the Hessian takes the simplified form,

$$\nabla^2(-\log\mathcal{L})\big|_{(\mu_*,\sigma_*)} = \frac{1}{\sigma_*^4}\begin{pmatrix} M\sigma_*^2 & 0 \\ 0 & -M\sigma_*^2 + 3M\sigma_*^2 \end{pmatrix} = \frac{M}{\sigma_*^2}\begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix},$$

which is clearly positive-definite (the eigenvalue are $\lambda = M/\sigma_*^2, 2M/\sigma_*^2 > 0$).
Therefore, the stationary point $(\mu,\sigma) = (\mu_*,\sigma_*)$ is a strict local minimum of $-\log\mathcal{L}$. Note that this must also be a strict global minimum: $-\log\mathcal{L}$ is a smooth function everywhere in its domain, so if any other point (μ,σ) were a local minimum, then it must also be a stationary point. But we have determined the only stationary point; therefore our local minimum is actually a global one over the interior of the domain. Since the boundary of the domain ($\sigma = 0$) is not contained in the set, we conclude that our stationary point is the strict global minimum.

The simulation of this problem is again given in the repo located at `https://github.com/akilnarayan/2021Fall-Optimization-homework3`, in the script `problem_P1.py`. The results are shown in Figure 3, demonstrating that the density function defined by the maximum likelihood estimate is a reasonable fit to the data's empirical histogram.

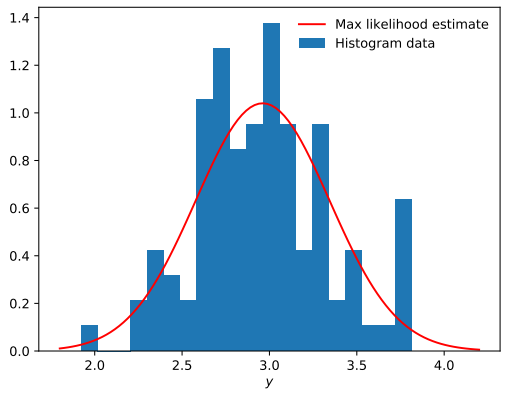

Figure 3: Plot associated to problem P1, showing histogram of data generated from a normal distribution compared to the maximum likelihood estimate defined by (μ_*, σ_*).

P2. (Maximum likelihood for coin flips) Suppose that you are given the result of 100 flips of a two-sided coin. Let H denote the number of heads observed, and T the number of tails (so that $H + T = 100$). Assume that $H, T > 0$. The coin may not be fair; it has probability $p \in [0, 1]$ that a heads is observed (and $1 - p$ for tails). The goal is determine the parameter p that maximizes the likelihood of having observed (H, T). Given that the likelihood is equal to

$$\mathcal{L}(p) = \begin{pmatrix} 100 \\ H \end{pmatrix} p^H (1-p)^T,$$

compute a maximum likelihood estimate for p. Is your computed value a global maximum?

Solution: First we note that for any $H, T > 0$, $\mathcal{L}(0) = \mathcal{L}(1) = 0$. Also, for any $p \in (0, 1)$, $\mathcal{L}(p) > 0$. Therefore, in order to maximize this likelihood, we need only consider the interior interval $p \in (0, 1)$, since both boundary points have strictly smaller likelihood than any point in the interior. In this interior, we again exercise the logarithm map,

$$\operatorname*{argmax}_{p \in (0,1)} \mathcal{L}(p) = \operatorname*{argmax}_{p \in (0,1)} \log \mathcal{L}(p),$$

and we proceed to optimize the log-likelihood. This is given by:

$$\log \mathcal{L}(p) = \log \begin{pmatrix} 100 \\ H \end{pmatrix} + H \log p + T \log(1-p),$$

which has gradient and Hessian,

$$\frac{\mathrm{d}}{\mathrm{d}p} \log \mathcal{L}(p) = \frac{H}{p} - \frac{T}{1-p}, \qquad \frac{\mathrm{d}^2}{\mathrm{d}p^2} \log \mathcal{L}(p) = -\frac{H}{p^2} - \frac{T}{(1-p)^2}.$$

The stationary point is then given by $p_* = \frac{H}{H+T} = \frac{H}{100}$. The Hessian for every $p \in (0, 1)$ is negative, so that $p = p_*$ is actually the strict global maximum of $\log \mathcal{L}$ for $p \in (0, 1)$,

and hence is also the strict global maximum for $\mathcal{L}$ for $p \in [0,1]$.

P3. (**6000-level students only**) Consider the optimization problem,

$$\min_{\boldsymbol{x} \in S \subset \mathbb{R}^n} f(\boldsymbol{x}),$$

where S is a given subset of $\mathbb{R}^n$ and $f : \mathbb{R}^n \to \mathbb{R}$ is a given function. Prove that if $g : \mathbb{R} \to \mathbb{R}$ is a strictly monotonic increasing function, then

$$\operatorname*{argmax}_{\boldsymbol{x} \in S \subset \mathbb{R}^n} f(\boldsymbol{x}) = \operatorname*{argmax}_{\boldsymbol{x} \in S \subset \mathbb{R}^n} g(f(\boldsymbol{x})), \qquad \operatorname*{argmin}_{\boldsymbol{x} \in S \subset \mathbb{R}^n} f(\boldsymbol{x}) = \operatorname*{argmin}_{\boldsymbol{x} \in S \subset \mathbb{R}^n} g(f(\boldsymbol{x})).$$

Solution: We first show the minimum property. Since g is a strictly monotonic increasing function, then $g(y) < g(z)$ when $y < z$. This is the key property we will use. Indeed, first define

$$S_- \coloneqq \operatorname*{argmin}_{\boldsymbol{x} \in S} f(\boldsymbol{x}),$$

which means by definition that for any $\boldsymbol{x}_* \in S_-$,

$$f(\boldsymbol{x}_*) = f(\boldsymbol{x}) \ \forall \boldsymbol{x} \in S_-, \tag{3}$$
$$f(\boldsymbol{x}_*) < f(\boldsymbol{x}) \ \forall \boldsymbol{x} \in S \backslash S_-. \tag{4}$$

First, we have

$$g(f(\boldsymbol{x}_*)) = g(f(\boldsymbol{x})) \ \forall \boldsymbol{x} \in S_-,$$

for any function g due to property (3). Due to the strict monotonicty of g and property (4), we also have

$$g(f(\boldsymbol{x}_*)) < g(f(\boldsymbol{x})) \ \forall \boldsymbol{x} \in S \backslash S_-.$$

Combining the two previous equations shows that

(a) $g(f(\boldsymbol{x}_*)) < g(f(\boldsymbol{x}))$ for any pair of points $\boldsymbol{x}_* \in S_-$ and $\boldsymbol{x} \in S \backslash S_-$
(b) $g(f(\boldsymbol{x}_*)) = g(f(\boldsymbol{x}))$ for any pair of points $\boldsymbol{x}_*, \boldsymbol{x} \in S_-$.

Thus, by definition $S_- = \operatorname{argmin}_{\boldsymbol{x} \in S} g(f(\boldsymbol{x}))$. We have therefore proven the minimum property.
The maximum property is a similar proof. Defining

$$S_+ \coloneqq \operatorname*{argmax}_{\boldsymbol{x} \in S} f(\boldsymbol{x}),$$

then by definition

$$f(\boldsymbol{x}_*) = f(\boldsymbol{x}) \ \forall \boldsymbol{x} \in S_+, \tag{5}$$
$$f(\boldsymbol{x}_*) > f(\boldsymbol{x}) \ \forall \boldsymbol{x} \in S \backslash S_+. \tag{6}$$

Therefore,

$$g(f(\boldsymbol{x}_*)) = g(f(\boldsymbol{x})) \ \forall \boldsymbol{x}_*, \boldsymbol{x} \in S_+,$$

for any function g, and since g is strictly monotone, then

$$g(f(\boldsymbol{x}_*)) > g(f(\boldsymbol{x})) \ \forall \boldsymbol{x}_* \in S_+, \ \ \boldsymbol{x} \in S \backslash S_+.$$

These two previous relations shows by definition that $S_+ = \operatorname{argmax}_{\boldsymbol{x} \in S} g(f(\boldsymbol{x}))$.